

excelenium - User Guide

Prabhu R

Table of Contents

1. Introduction and Features	1
2. Features	2
3. Prerequisites	3
4. Building and Running excelenium.	4
4.1. Building excelenium	4
4.2. Running excelenium	4
5. Creating and Running the Suite	5
5.1. Creating the Suite	5
5.2. Creating Scripts	7
5.3. Running the Suite	16
6. Save and Load Suite	17
6.1. Save the Suite	17
6.2. Load the Suite	17

Ch. 1. Introduction and Features

This is the manual about **exce^lenium**, an automation tool that enables automation testing of web applications using spreadsheet. It uses the popular Selenium WebDriver for automation. Unlike the regular Selenium Scripts which are run as JUnit test cases, this script is prepared using a spreadsheet in a web-based UI. **exce^lenium** provides an array of features that would speed up automation script writing as there is no need for any knowledge in programming or JUnit. **exce^lenium** can execute automation tests on Google Chrome, Mozilla Firefox, Microsoft Edge, Opera and Safari browsers.

Ch. 2. Features

The tool has the following major features

1. Automation Script using spreadsheets in a web-based UI
2. Single executable jar file
3. No external dependencies, except the browser drivers and the browsers themselves.
4. Intuitive Report and log file
5. Concurrently run tests on the browsers
6. Supported Browsers - Mozilla Firefox, Google Chrome, Microsoft Edge, Opera, and Safari (Safari - only on macOS)
7. Ability to set custom user agent and test mobile browser test scenarios

Ch. 3. Prerequisites

The tool is an executable jar with all the dependencies included and can be directly executed from the command-line; however, there are some basic prerequisites that are required for running this tool.

The prerequisites are

1. Java JDK 8 or higher
2. Apache Maven – 3.6.0 or higher (required only if building from source)
3. Mozilla Firefox, Google Chrome, Microsoft Edge, Opera and Safari Browsers (Install only the required browsers)
4. Browser WebDrivers - Can be downloaded from the following locations
 - a. Mozilla Firefox - [Gecko Driver](https://github.com/mozilla/geckodriver/releases) [https://github.com/mozilla/geckodriver/releases]
 - b. Google Chrome - [Chrome Driver](https://sites.google.com/a/chromium.org/chromedriver/downloads) [https://sites.google.com/a/chromium.org/chromedriver/downloads]
 - c. Microsoft Edge Driver - [Edge Driver](https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/#downloads) [https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/#downloads]
 - d. Opera - [Opera Driver](https://github.com/operasoftware/operachromiumdriver/releases) [https://github.com/operasoftware/operachromiumdriver/releases]
 - e. Safari - In-built (Supported only on macOS)
5. Git – to clone the repository (required only if building from source)



Microsoft Edge driver is applicable for version above 18 that use the Chromium Engine. For others please read the notes in the link above for Microsoft Edge

Ch. 4. Building and Running excelenium

4.1. Building excelenium

To build **excelenium** from source follow these steps

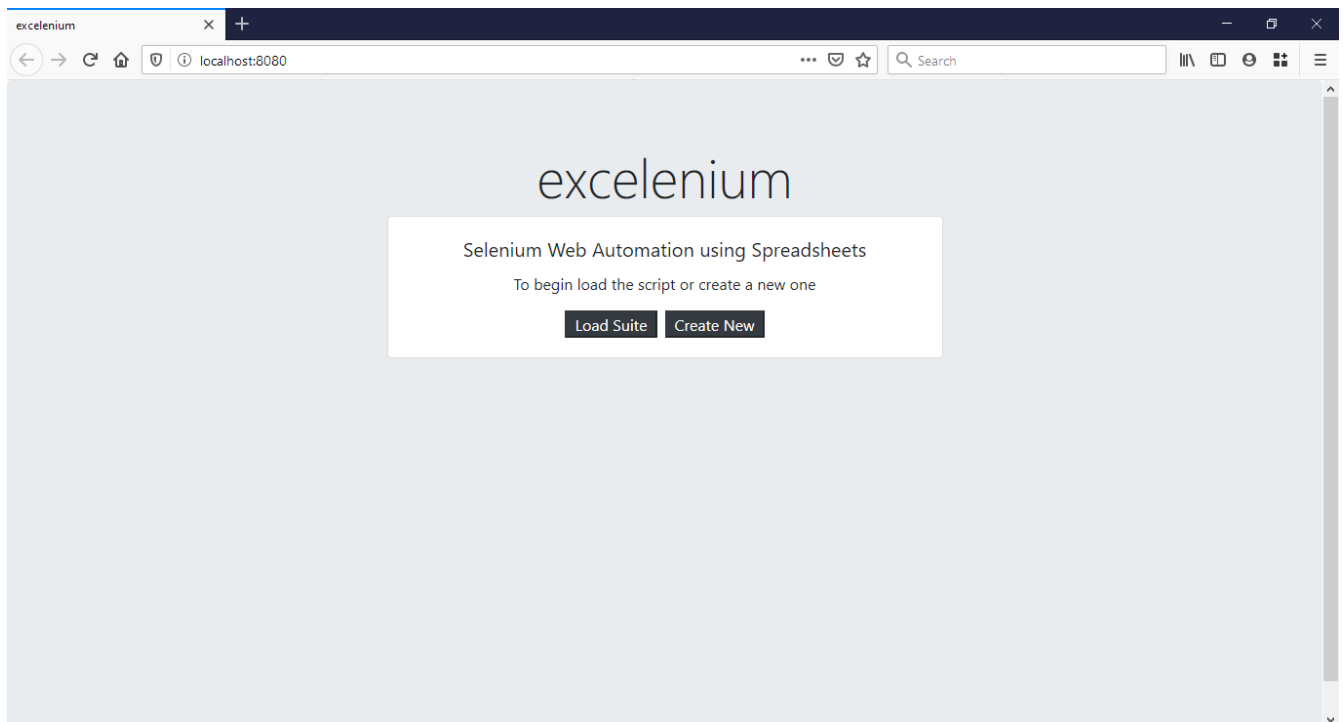
1. Clone the repository
2. Make sure you have installed Java and Maven and set the **JAVA_HOME** and **MVN_HOME** environment variables and both are in the system **PATH** variable
3. Move the directory where the **excelenium** repository has been checked out, open a command line in that directory
4. Type **mvn clean package**
5. This will generate **excelenium-0.0.1-SNAPSHOT.jar** inside the target directory
6. The generated JAR is a fat jar, which means, it is a self-contained jar that has all the required dependencies.

4.2. Running excelenium

To run **excelenium** type the following command in the command line

```
java -jar excelenium-0.0.1-SNAPSHOT.jar
```

This will open the browser with the following UI



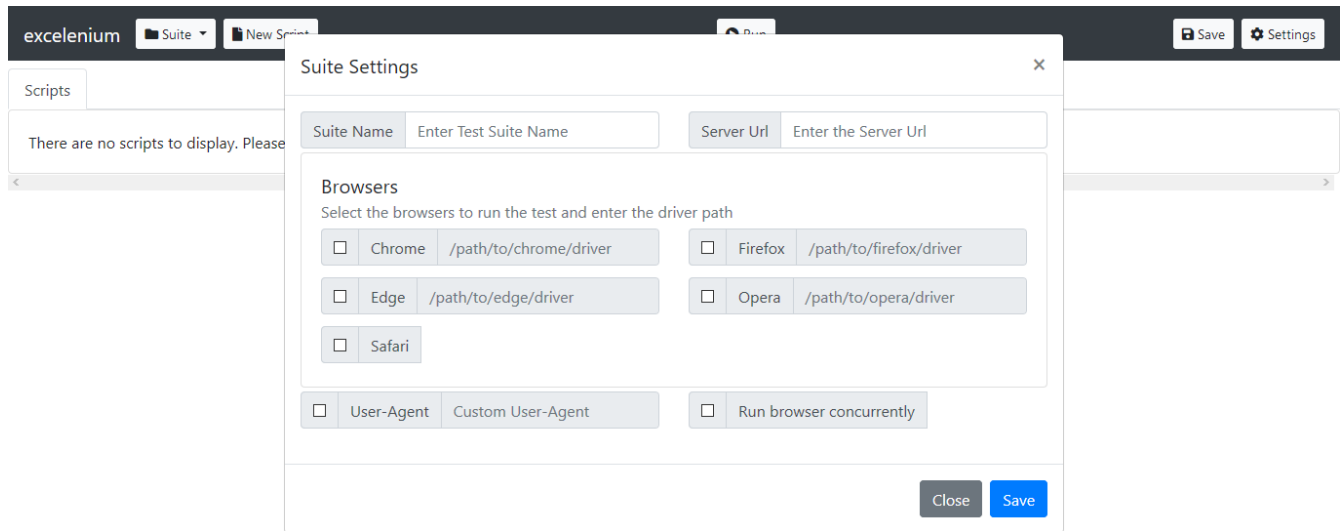
Ch. 5. Creating and Running the Suite

5.1. Creating the Suite

In the landing page, there are two options **Load Suite** and **Create New**.

We will look at saving and loading the suite in subsequent sections.

Clicking on the **Create New** navigates to the new suite page



The **New Suite Page** displays the **Suite Settings** dialog

Suite Settings

Suite Name

Enter Test Suite Name

Server Url

Enter the Server Url

Browsers

Select the browsers to run the test and enter the driver path

☐

Chrome

/path/to/chrome/driver

☐

Firefox

/path/to/firefox/driver

☐

Edge

/path/to/edge/driver

☐

Opera

/path/to/opera/driver

☐

Safari

☐

User-Agent

Custom User-Agent

☐

Run Browsers Concurrently

Close

Save

5.1.1. The Suite Settings Dialog

Fill the settings in the **Suite Settings** dialog, the description of which are as follows

- **Suite Name** - Provide the name for the suite
- **Server URL** - The base url to start the test with

The **Browsers** section displays the browsers on which the suite should be executed and their corresponding webdriver executables path, if selected.



At least one browser should be selected for running the suite. Also, **Safari** browser is support only on the Apple macOS

- **User-Agent** - The user agent string to spoof the browser user agent, useful to automate mobile websites.



The user-agent field cannot be set in Edge and Safari. Mostly works only with Mozilla Firefox and Google Chrome

- **Run Browsers Concurrently** - If checked, the browsers will be run concurrently, else they will be run in sequence.

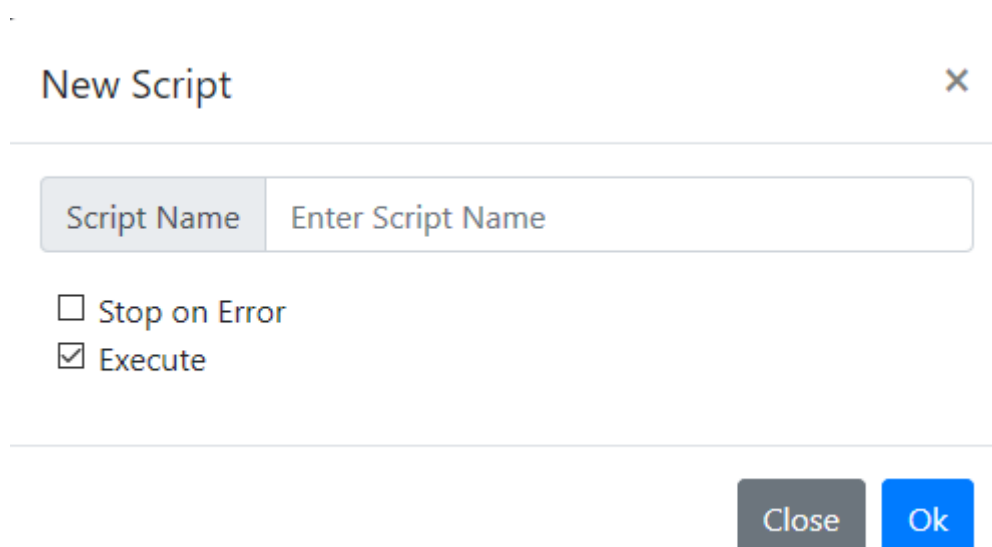


There could be unexpected behaviour if all the browsers are run concurrently because of memory, cpu wait, network latency etc. Though the webdrivers are run in separate threads.

5.2. Creating Scripts

Once the settings have been input and saved. We can start creating scripts.

Click on the  **New Script** button or **Ctrl+M** to open the **New Script** dialog.



The 'New Script' dialog box has a title bar with a close button (X). Below the title bar is a text input field with a placeholder 'Enter Script Name' and a label 'Script Name'. Below the input field are two checkboxes: 'Stop on Error' (unchecked) and 'Execute' (checked). At the bottom right are two buttons: 'Close' (grey) and 'Ok' (blue).

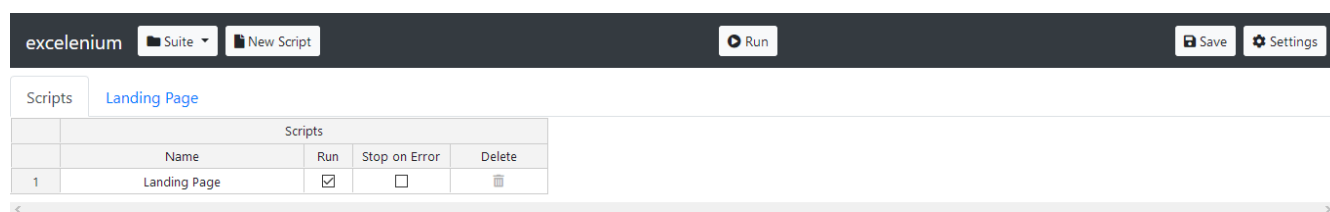
To create a script the following fields have to be filled in the **New Script** dialog

- **Script Name** - Name for the script
- **Stop On Error** - Stop executing the script if checked, default is unchecked
- **Run** - Run the script, if checked.


Scripts in the suite can be chosen to run or not based on the **Run** being checked.


Once the details have been entered, a new script is created in the suite and a tab with the script name is created. In addition, there is also another tab names **Scripts** that lists the scripts in the suite

First let's see about the **Scripts** tab, before we see the details of actual script. The following image shows the **Scripts** tab



The screenshot shows the 'excelenium' interface with a 'Suite' dropdown and a 'New Script' button. The 'Scripts' tab is active, showing a table with the following data:

Scripts				
	Name	Run	Stop on Error	Delete
1	Landing Page	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

The columns should be self-explanatory. There is a delete column that has  icon. Clicking on it deletes the script.



Script once deleted cannot be undone.

Individual script tabs can be closed by clicking on the close button in the tab. Closed tabs can be

reopened by clicking on the script name in the **Scripts** tab.

Individual scripts can be switched on/off from executing by selecting the **Run** checkbox

The script name can be changed by clicking on the Name cell of the script in the **Scripts** tab

Now let's see the Script Sheet Tab

Scripts shown in the **Scripts** tab will be executed in sequence based on the **Run** flag.

excelenium

Suite

New Script

Run

Errors

Save

Settings

Scripts

Home Page

+1

	Actions								Preprocess		
	Run	Stop on Error	Name	Type	Element Id/XPath	Element Val	Attr Name	Attr Val	Action	Attr Name	Attr Val
1	<input type="checkbox"/>	<input type="checkbox"/>									
2	<input type="checkbox"/>	<input type="checkbox"/>									
3	<input type="checkbox"/>	<input type="checkbox"/>									
4	<input type="checkbox"/>	<input type="checkbox"/>									
5	<input type="checkbox"/>	<input type="checkbox"/>									
6	<input type="checkbox"/>	<input type="checkbox"/>									
7	<input type="checkbox"/>	<input type="checkbox"/>									
8	<input type="checkbox"/>	<input type="checkbox"/>									
9	<input type="checkbox"/>	<input type="checkbox"/>									
10	<input type="checkbox"/>	<input type="checkbox"/>									
11	<input type="checkbox"/>	<input type="checkbox"/>									
12	<input type="checkbox"/>	<input type="checkbox"/>									
13	<input type="checkbox"/>	<input type="checkbox"/>									
14	<input type="checkbox"/>	<input type="checkbox"/>									
15	<input type="checkbox"/>	<input type="checkbox"/>									
16	<input type="checkbox"/>	<input type="checkbox"/>									
17	<input type="checkbox"/>	<input type="checkbox"/>									
18	<input type="checkbox"/>	<input type="checkbox"/>									
19	<input type="checkbox"/>	<input type="checkbox"/>									
20	<input type="checkbox"/>	<input type="checkbox"/>									

A script contains a sequence of actions that will be executed by the WebDriver on the web page rendered by the browser.

By default, there are 20 rows added to the scripts sheet. To add more right click on the last row and add more rows as needed. Click the **+1** button in the toolbar for adding one row at the end of the spreadsheet. To add 10 rows at the end click the **≡** button.

The script can be grouped in the two sections

- The Actions Section
- The Preprocess Section

The first two columns are the **Run** and **Stop On Error**

Column	Description
Run	Executes the action if checked
Stop On Error	Stops the entire script if checked . Default unchecked

5.2.1. Action Section

The actions section has the columns that describe what the action does. The following explains the columns

Column	Description
Name	The name of the action
Type	One of the permissible values in the dropdown as shown in the table below Action Types
Element Id/XPath	Represents a web element – xpath/id . Or specified values depending on actions. For e.g. Action Type NAVIGATE can take one of the values – back , forward , refresh or a URL. This column is optional in some cases
Element Val	Represents the value of the web element, optional in some cases
Attribute Name	Represents the attribute name of the element on which an action has to be performed. Optional in some cases
Attribute Val	Represents the attribute value of the element on which an action is performed. Optional in some cases

5.2.2. Preprocess Section

The Preprocess Section is executed prior to the action specified. It is normally used to set the attribute of element in consideration. The following table explains the columns

Column	Description
Action Type	One of permissible values - ADD_ATTRIBUTE – to add an attribute before performing the main action - REMOVE_ATTRIBUTE – to remove an attribute before performing the main action - SET_ATTRIBUTE – to set an attribute value before performing the main action
Attribute Name	Name of the attribute
Attribute Value	Value of the attribute

The following table explains the action types that was mentioned above in the [Action Section](#), the **Type** column in the table above

5.2.3. Action Types

The following action types are available in **excelemium** and are described in the table below

Action Type	Description
FILL	Fills the value in for the element selected, usually, text field or text area. Optionally can also do a submit in case of search boxes etc.
CLEAR	Clears the content of the element selected, usually, text field or text area.
CLICK	Generates a mouse click for the element selected.
RIGHT_CLICK	Generates a mouse right click for the element selected
CHECK	Toggles the check box.
SELECT	Selects the option in the drop down as specified in the Element Value column. A special variable <code><#random></code> can be used to randomly select one of the values in the dropdown.
VERIFY_TEXT	Verifies the text content of the element.
VERIFY_PRESENT	Verifies if an element is present in the DOM use IS_VISIBLE to check if the element is displayed.
IS_VISIBLE	Verifies if an element is displayed and not hidden.
IS_HIDDEN	Verifies if an element is hidden and not displayed.
CHECK_ATTRIBUTE	Checks if the attribute is with the value specified is set for the element.
ACCEPT_POPUP	Selects OK/Yes when the browser popup is displayed. Popup here refers to the native popup and not the modals generated using JavaScript
DISMISS_POPUP	Selects Cancel/No when the browser popup is displayed. Popup here refers to the native popup and not the modals generated using JavaScript
SWITCH_TO_IFRAME	Switch to iframe specified.
SWITCH_TO_PARENT	Switch to parent from iframe.
CAPTURE_SCREEN	Captures the currently rendered screen in the browser and saves as a png image.
NAVIGATE	Navigates back/forward/to a particular URL.
SET_VARIABLE	Sets a variable that can be later substituted in the script.
UNSET_VARIABLE	Unsets a variable that has already been set.
CLEAR_COOKIES	Clears all the cookies that are currently present
DELETE_COOKIE	Deletes the specified cookie
ADD_COOKIE	Adds the specified cookie with the given value
SWITCH_TO_WINDOW	Switches to the specified window/tab - based on index from 0 to n.
EXECUTE_JAVASCRIPT	Executes the provided JavaScript file.
WAIT_MSECS	Wait for the specified milliseconds before performing the next action. This is in addition to the implicit waits provided in Selenium.

Action Type	Description
IS_ENABLED	Checks if the specified element is enabled
IS_DISABLED	Checks if the specified element is disabled
SET_WINDOW_SIZE	
COMPARE_URL	Compares the current browser url with the provided url considering the options - starts_with and full_url.
RUN_SCRIPT	<p>Runs the provided beanshell script or groovy script.</p> <p>For beanshell, Refer: www.beanshell.org [https://www.beanshell.org].</p> <p>For Groovy script. Refer: groovy-lang.org [https://groovy-lang.org].</p> <p>The script is supplied with all the variables created using SET_VARIABLE and a few other internal variables in a map named inputMap that can be referenced in the script.</p> <p>The logger object is also available in the name logger that can be used to print debug messages.</p> <p>The script will have to create a HashMap in the name result and store all the results that it wants printed in the log after execution.</p> <p>Also, the result hashmap object should have an entry status which is either true or false. The value true indicates the script executed successfully and false if there were errors in the expected output.</p>
GET_DOM	Gets the dom of the specified element and stores in the variable name specified.
SCROLL_WINDOW_BY	Scrolls the window by specified x and y pixels
SCROLL_TO_ELEMENT	Scrolls the window to the specified web element
MAKE_REQUEST	Makes a GET request and stores the response in the variable name specified.
HOVER	Hovers the mouse pointer on the specified element
DRAG_AND_DROP	Drag and drop the source element to target element
GET_CURRENT_URL	Gets the current url of the current focused window and saves it to the variable specified
HAS_CSS_CLASS	Checks if the specified element has the css classes applied
CHECK_CSS_ATTRIBUTE	Checks if the css attribute of the element has the specified value

As mentioned earlier, the element, element value, attribute name and the attribute value columns are optional for some of the action types. The following table describes those

The following table explains the possible values in the columns of the [Action Section](#) for each of the [Action Types](#) described above

Action-Type	Element	Element Value	Attribute Name	Attribute Value
FILL	id or xpath of the element	Value to be filled in that element	True/False to indicate if the field needs to be submitted i.e., Enter key pressed. For example, "Search. Default "FALSE"	None
CLEAR	id or xpath of the element	None	None	None
CLICK	id or xpath of the element	None	None	None
RIGHT_CLICK	id or xpath of the element	0 based index indicating the option to be chosen in case of a native browser context menu, -ve value in case of a Javascript generated context menu	None	None
CHECK	id or xpath of the element	None	None	None
SELECT	id or xpath of the element	Text value of the item to be selected or <#random> to randomly select one of the items in the dropdown	None	None
VERIFY_TEXT	id or xpath of the element	Text value to be verified	<p>One of following options starts_with: compares if the text of the element starts with the provided text.</p> <p>ends_with: compares if the text element ends with the provided text.</p> <p>contains: checks if the text of the element contains the provided text</p> <p>full_text: compares the text of the element to the provided text</p>	None

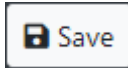
Action-Type	Element	Element Value	Attribute Name	Attribute Value
VERIFY_PRESENT	id or xpath of the element	None	None	None
IS_VISIBLE	id or xpath of the element	None	None	None
IS_HIDDEN	id or xpath of the element	None	None	None
CHECK_ATTRIBUTE	id or xpath of the element	None	Name of the attribute	Value of the attribute
ACCEPT_POPUP	None	None	None	None
DISMISS_POPUP	None	None	None	None
SWITCH_TO_IFRAME	id or xpath of the element	None	None	None
SWITCH_TO_PARENT	None	None	None	None
CAPTURE_SCREEN	Optional filename without extension	None	None	None
NAVIGATE	back, forward, refresh or url	None	None	None
SET_VARIABLE	xpath of the element or variable name	variable name in case xpath set as element or variable value otherwise	None	None
UNSET_VARIABLE	variable name to be unset	None	None	None
CLEAR_COOKIES	None	None	None	None
DELETE_COOKIE	Cookie name	None	None	None
ADD_COOKIE	Cookie name	Cookie Value	None	None

Action-Type	Element	Element Value	Attribute Name	Attribute Value
SWITCH_TO_WINDOW	0 based index, where 0 always indicates the main window and subsequent windows opened are numbers sequentially	None	None	None
EXECUTE_JAVASCRIPT	absolute path to the JavaScript file	None	None	None
WAIT_MSECS	Time to wait in milliseconds before executing next action	None	None	None
IS_ENABLED	id or xpath of the element	None	None	None
IS_DISABLED	id or xpath of the element	None	None	None
SET_WINDOW_SIZE	Positive integer value specifying the width	Positive integer value specifying the height	None	None

Action-Type	Element	Element Value	Attribute Name	Attribute Value
COMPARE_URL	Url to compare against the current browser url	<p>One of following options - starts_with: compares if the current browser url starts with the provided url. The url might contain additional query parameters that might need to be ignored.</p> <p>ends_with: compares if the current browser url ends with the provided url.</p> <p>contains: checks if the browser url contains the provided text</p> <p>full_url: compares the entire url</p>	None	None
RUN_SCRIPT	absolute path base directory that contains the bsh or groovy script files	file name with extension (.bsh or .groovy)	None	None
GET_DOM	id or xpath of the element	Variable name to store the DOM	None	None
SCROLL_WINDOW_BY	number of x pixels to scroll	number of y pixels to scroll	None	None
SCROLL_TO_ELEMENT	id or xpath of element to scroll to	None	None	None
MAKE_REQUEST	url to request	Variable name to store response	None	None
HOVER	id or xpath of element	None	None	None
DRAW_AND_DROP	id or xpath of source element	id or xpath of target element	None	None

Action-Type	Element	Element Value	Attribute Name	Attribute Value
GET_CURRENT_URL	Variable name to store the url	None	None	None
HAS_CSS_CLASS	id or xpath of element	Comma separated list of css classes	None	None
CHECK_CSS_ATTRIBUTE	id or xpath of element	Css attribute name	Css attribute value	None

5.3. Running the Suite

Once the scripts in the suite have been prepared, clicking on the  Save saves the suite to the server memory. In addition, **exelenium** also validates the suite and if there any errors, it displays a dialog similar to the one below

Errors / Warnings

ERROR: Browser driver path for firefox is invalid - D:/Progs/selenium1/webdrivers/geckodriver/geckodriver.exe - does not exist

ERROR: fillSuite - Attribute Name has to be either true or false.

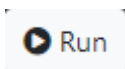
WARNING: fillSuite - Attribute Value fields will be ignored.

ERROR: fillUrl - Attribute Name has to be either true or false.

WARNING: fillUrl - Attribute Value fields will be ignored.

Close

The errors dialog can be viewed again by clicking on the .

Clicking on the  Run button, saves and validates the suite, and if the suite is valid it executes the suite on the browsers selected in the settings dialog.

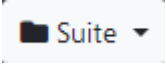
Each browser will have a separate log file in its name along with the timestamp.

Also, the screenshots captured during the execution of the suite will be stored in the screens folder in the directory where the **exelenium** jar file is executed from.

Moreover the screenshots will be named based on the browser it is executed. For example, if the suite is executed on Mozilla Firefox, the name firefox will be appended along with the timestamp to the screenshot file name. :imagesdir: ./images :experimental: true :icons: font

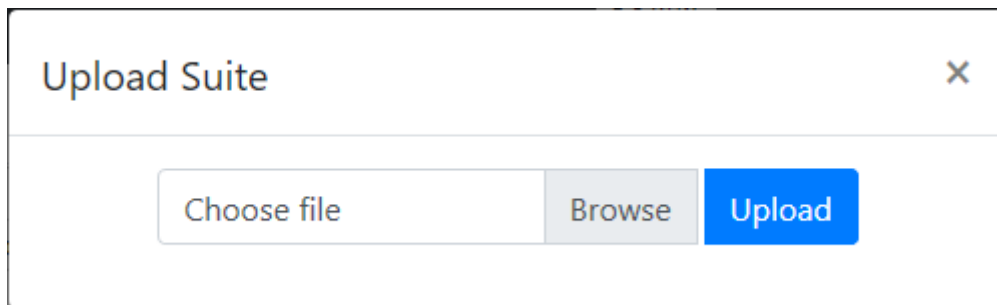
Ch. 6. Save and Load Suite

6.1. Save the Suite

To save the current suite, click on the  dropdown and click on **Export** option or press **Ctrl+E** which will export the suite file in json format.

6.2. Load the Suite

To load the suite, click on the  dropdown and click **Load** option or press **Ctrl+L** which will display the **Upload Suite** dialog



Choose the suite file saved earlier and click upload to load the suite file.