

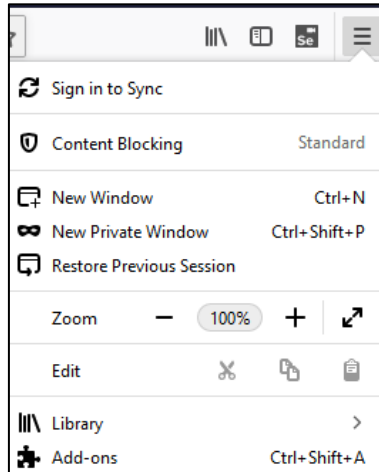
UECS2354 Software Testing

Lab 09: Introduction to Selenium

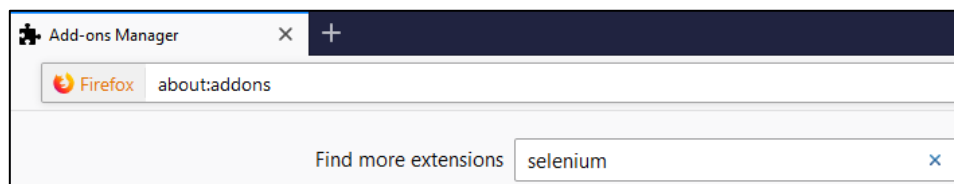
Selenium IDE allows testers and developers to record their actions based on the workflow so that the actions can be repeated.

Setup – Firefox Quantum 66.0.1

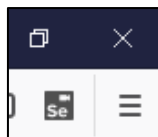
1. Click on the menu button on the top-right corner. Then click on **Add-ons**.



2. Search for **selenium**.



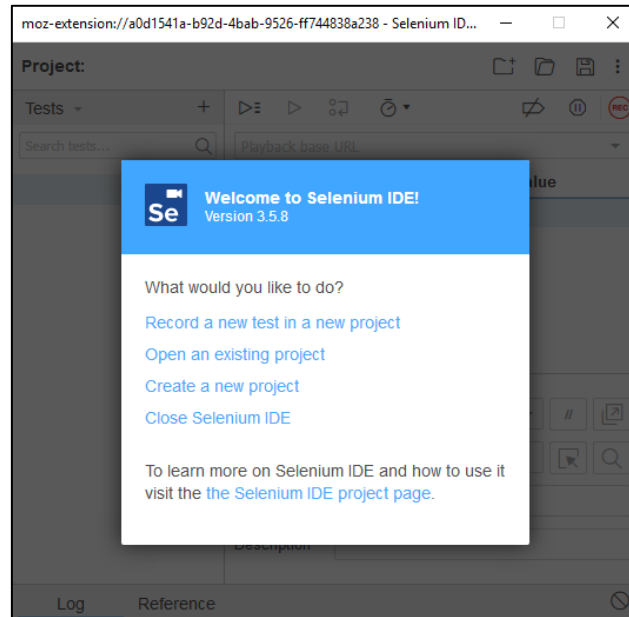
3. Add **Selenium IDE** to Firefox. A button should be added next to the menu button.



4. Click the Selenium IDE button to launch it. We can choose to record the actions we perform and we can replay it in a later time. That means we can automate the execution of the actions.

UECS2354 Software Testing

Lab 09: Introduction to Selenium



5. Selenium is also available in Google Chrome.

Try It

1. Select **Record a new test in a new project**. Enter a name for the project.

A screenshot of a dialog box titled 'Name your new project'. It contains the text 'Please provide a name for your new project.' followed by a text input field labeled 'PROJECT NAME'. Below the input field, it says 'You can change the name of your project at any time by clicking it and entering a new name.' At the bottom right, there are two buttons: 'OK' and 'CANCEL'.

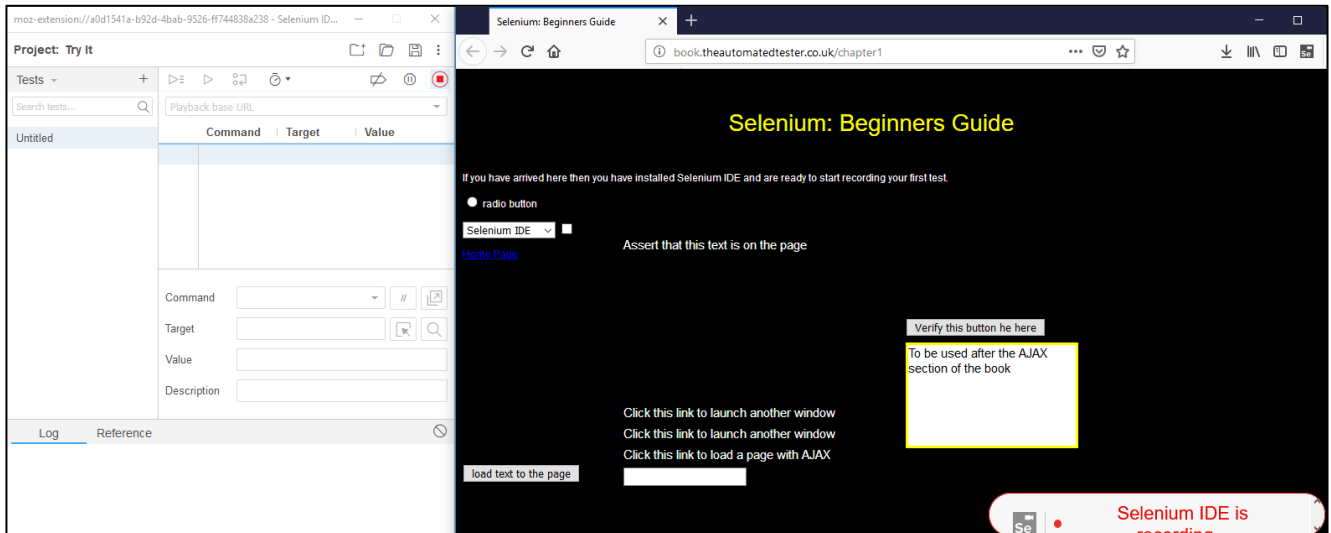
2. Enter the base URL. Use the following URL and click on **Start Recording**.

`http://book.theautomatedtester.co.uk/chapter1`

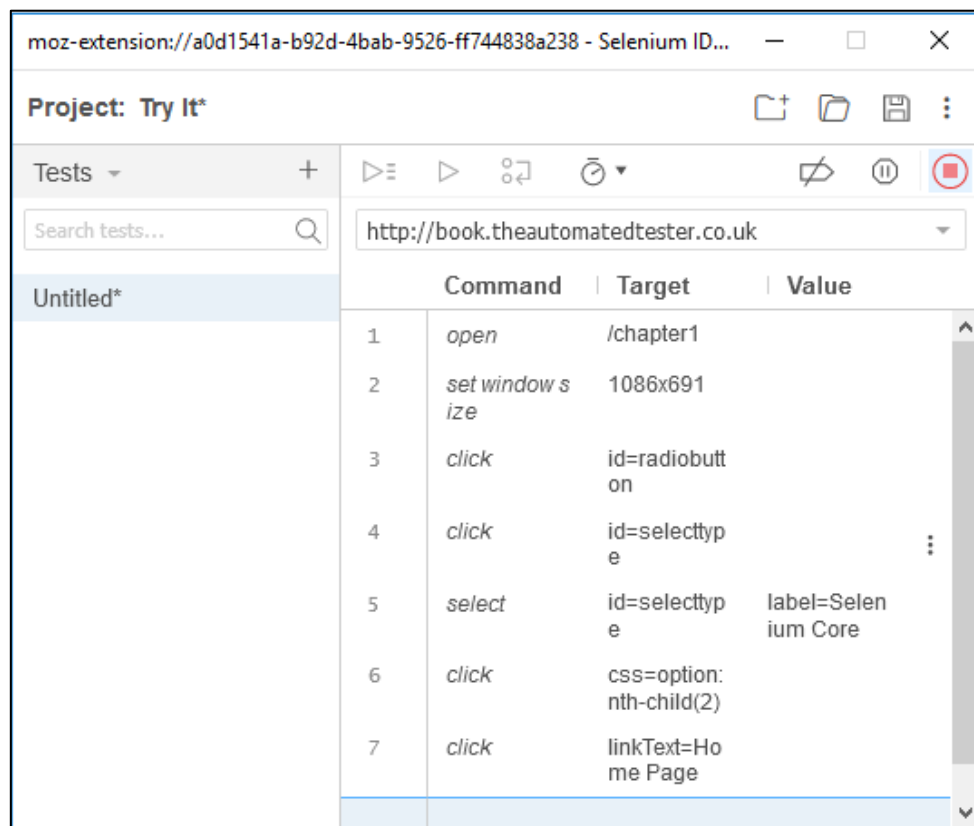
A screenshot of a dialog box titled 'Set your project's base URL'. It contains the text 'Before you can start recording, you must specify a valid base URL for your project. Your tests will start by navigating to this URL.' followed by a text input field labeled 'BASE URL'. The input field contains the URL 'http://book.theautomatedtester.co.uk/chapter1'. At the bottom, there are two buttons: 'START RECORDING' and 'CANCEL'.

UECS2354 Software Testing

Lab 09: Introduction to Selenium



3. Click the **radio button** on the page.
4. Select a value from the **drop-down list**.
5. Click on the **Home Page** link.
6. Notice that the 3 actions that we performed have been recorded.



UECS2354 Software Testing

Lab 09: Introduction to Selenium


7. Click on the **Stop** button on the top-right corner to stop the recording. Give a name to the test.

×

Name your new test





Please provide a name for your new test.




TEST NAME

You can change it at any time by clicking the  icon next to its name in the tests panel.

OK

LATER

8. Click on the **Save** button to save the project.
9. We can choose to run all recorded tests  or current test .
10. Play current test and observe the actions. If it is too fast, we can adjust the execution speed .
11. It is possible to delete recorded actions. Click on the  icon for options.

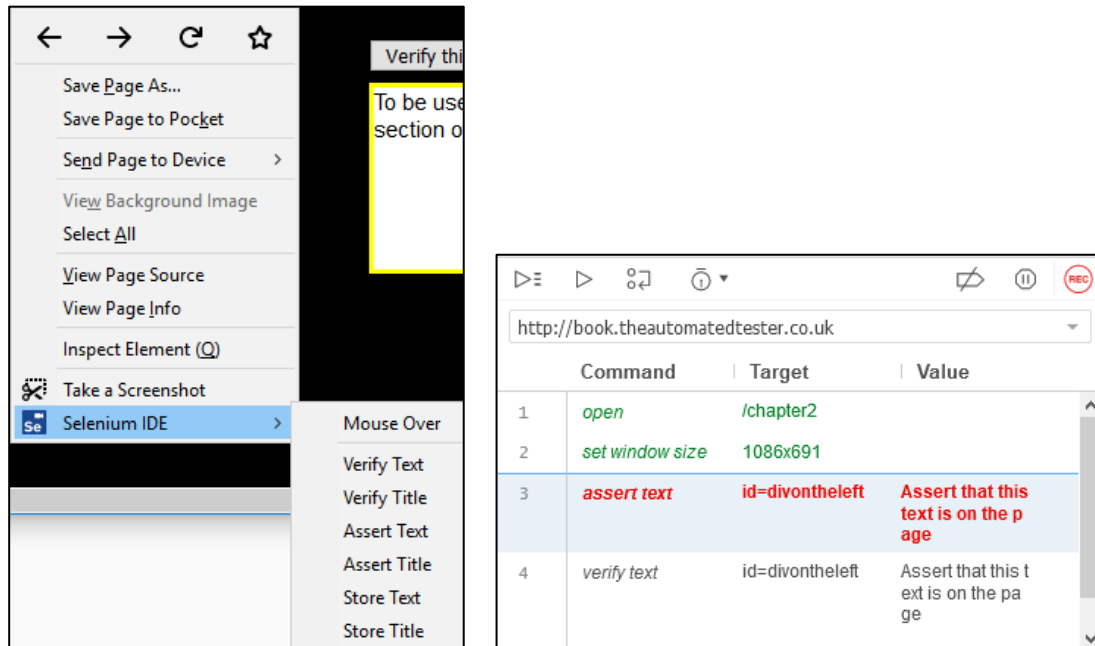
8	verify text	css=html	Selenium: Beginners Guide\nIf you have arrived here then you have installed Selenium IDE and are ready to start recording y
9	click	id=divontheleft	<div><div>CutCtrl+X</div><div>CopyCtrl+C</div><div>PasteCtrl+V</div><div>DeleteDel</div></div>
10	click	id=divontheleft	
11	verify text	id=divontheleft	
<div><div>Command</div><div>verify text</div><div>//</div><div></div></div> <div><div>Target</div><div>css=html</div><div></div><div></div></div>			

UECS2354 Software Testing

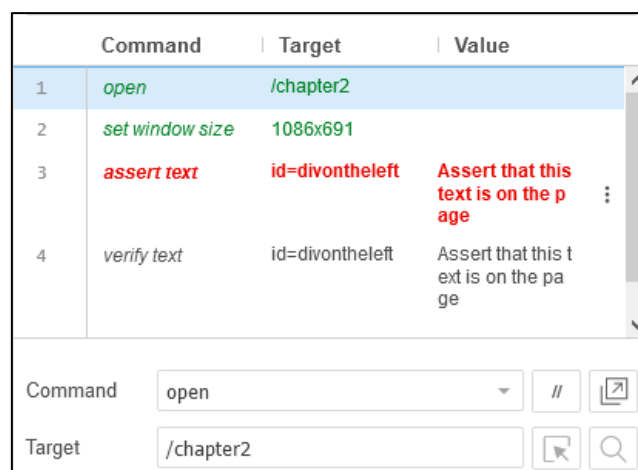
Lab 09: Introduction to Selenium

Verify Components

1. Selenium also allows us to validate the existence of certain components on the page by using **assert** or **verify**.
2. Using the same URL, start the recording.
3. Right-click on the text **Assert that this text in on the page** and choose **Assert Text**. Notice that the **assert text** has been recorded. If the text doesn't exist on the page, the test will fail and the execution will stop.



4. Change the **Target** for the **open** command.



5. Remove other actions (click radio button, select from list and click on homepage link) or move the assert text to the top (click on the action and drag to reposition the action in the list).
6. Run the test. The **assert text** command should fail.

UECS2354 Software Testing

Lab 09: Introduction to Selenium

Simple test*

http://book.theautomatedtester.co.uk

	Command	Target	Value
1	open	/chapter2	
2	set window size	1086x691	
3	assert text	id=divontheleft	Assert that this text is on the page
4	verify text	id=divontheleft	Assert that this text is on the page

Runs: 1 Failures: 1

Log Reference

1. open on /chapter2 OK

2. setWindowSize on 1086x691 OK

3. assertText on id=divontheleft with value Assert that this text is on the page Failed:
Actual value "" did not match "Assert that this text is on the page"

'Simple test' ended with 1 error(s)

- Right-click on the text **Assert that this text in on the page** and choose **Verify Text**. Move the verify text action to the top of assert text action and run the test. Notice that when **Verify Text** fails, the test continues to execute.

Simple test*

http://book.theautomatedtester.co.uk

	Command	Target	Value
1	open	/chapter2	
2	set window size	1086x691	
3	verify text	id=divontheleft	Assert that this text is on the page
4	assert text	id=divontheleft	Assert that this text is on the page

Runs: 1 Failures: 1

Log Reference

3. verifyText on id=divontheleft with value Assert that this text is on the page Failed:
Actual value "" did not match "Assert that this text is on the page"

4. assertText on id=divontheleft with value Assert that this text is on the page Failed:
Actual value "" did not match "Assert that this text is on the page"


'Simple test' ended with 2 error(s)

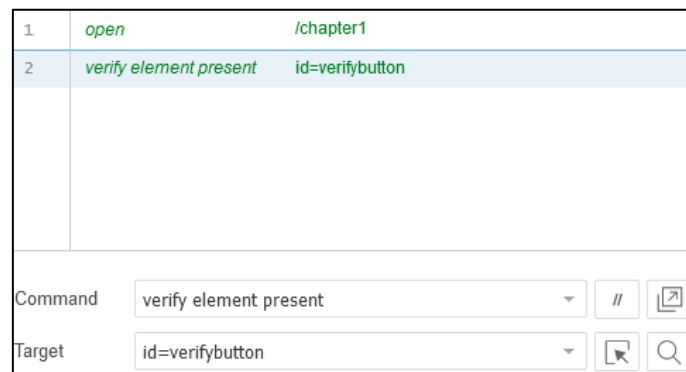
- We can add commands other than the verify, assert commands in the right-click menu. For example, to verify the existence of an element, we use the **verify element present** command. We can choose the command from the drop-down list.

UECS2354 Software Testing

Lab 09: Introduction to Selenium



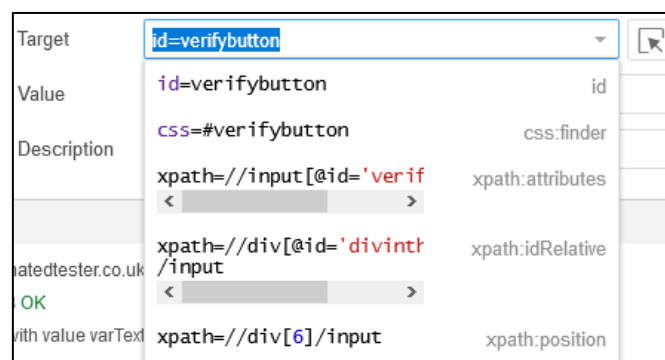
9. Next, we need to specify the target element, click on  to select the button **Verify this button be here**.



10. Run the test case and it should be passed.
11. When **verify** command is used in the test case, the IDE will throw an Error if the component is not found, and then continue with the rest of the test. However, if **assert** command is used, the test will not continue if item is not found.

Multiple Locators

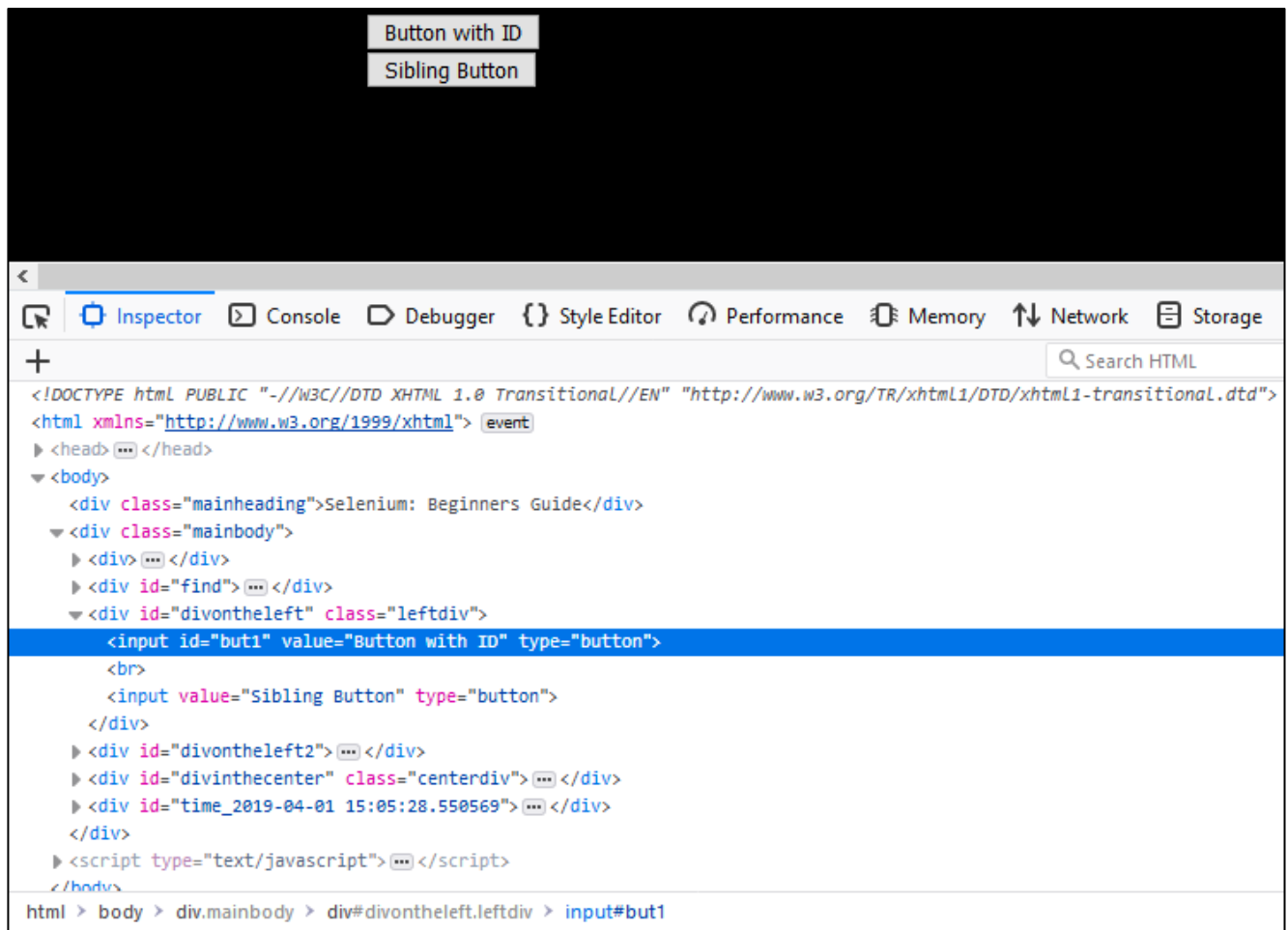
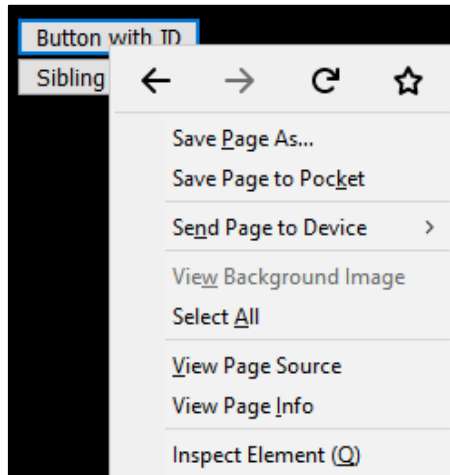
Selenium IDE will store multiple locators for an element (in Target) which is useful during the playback to the test. If one locator fails, the others will be tried until one is successful.



To see information about an element, right-click on the element and choose **Inspect Element**.

UECS2354 Software Testing

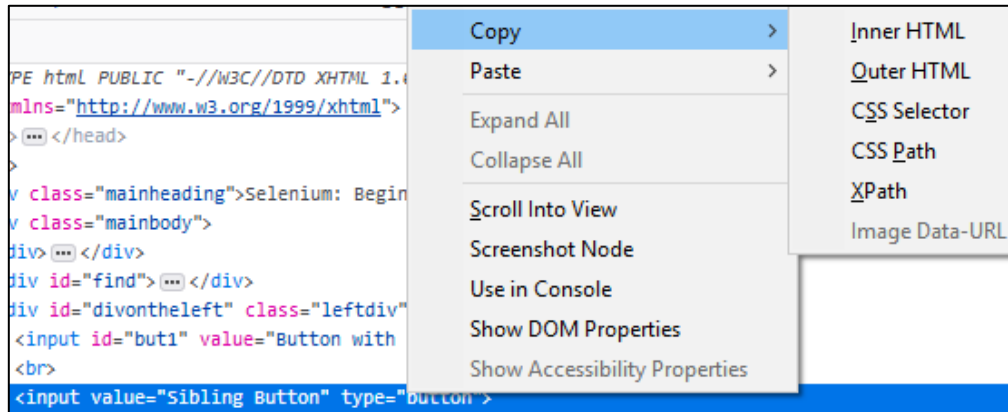
Lab 09: Introduction to Selenium



CSS Selector, XPath and other locators for the element can be copied by right-click on the source and choose the appropriate option.

UECS2354 Software Testing

Lab 09: Introduction to Selenium



For a complete CSS Selector reference, you may refer to following or other resources.

https://www.w3schools.com/cssref/css_selectors.asp

In general, XPath has the following syntax:

`xpath=//tagname[@attribute='value']`

where

//	Current node
tagname	Node's tagname (e.g. div, input)
@	Select attribute
Attribute	Attribute name
Value	Attribute value

Example: `xpath=//input[@value='Sibling Button']`

UECS2354 Software Testing

Lab 09: Introduction to Selenium

Test on Multiple Windows

Web application might create a new pop-up windows, the testing might involve multiple windows.

1. Create a new test, set the URL to <http://book.theautomatedtester.co.uk/chapter1> and start recording.
2. Click any of the [**Click this link to launch another window**] and a small window will appear. The browser might block the pop-up. Allow the pop-up window manually.
3. In the pop-up window, **verify** [Text within the pop up window].
4. Close the pop-up window.
5. Verify [**Assert that this text in on the page**]. The Selenium IDE should look like the following screenshot.

1	<i>open</i>	http://book.theautomatedtester.co.uk/chapter1	
2	<i>set window size</i>	998x752	
3	<i>click</i>	id=multiplewindow	
4	<i>store window handle</i>	root	
5	<i>select window</i>	handle=\${win7584}	
6	<i>verify text</i>	id=popuptext	Text within the pop up window
7	<i>close</i>		
8	<i>select window</i>	handle=\${root}	
9	<i>verify text</i>	id=divontheleft	Assert that this text is on the page
10	<i>close</i>		

6. Before the pop-up window is selected by the command **select window**, the parent window is stored by the **store window handle** command.
7. When the pop-up window is closed, it will return to the parent window, done by the **select window** command with target **handle=\${root}**.

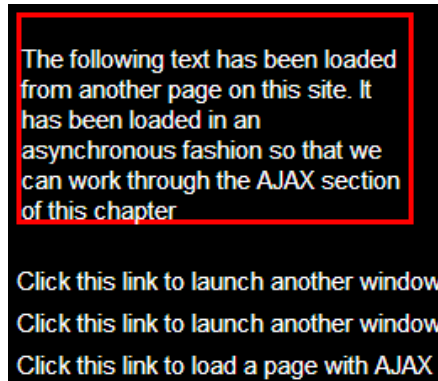
Now Try This:

1. Create a new test, set the URL to <http://book.theautomatedtester.co.uk/chapter1> and start recording.
2. Click the first [**Click this link to launch another window**]. **Verify** [Text within the pop up window] in the pop-up window.
3. Back to the parent window (<http://...../chapter1>) and click the second [**Click this link to launch another window**]. **Assert** [Text within the pop up window] in the pop-up window.
4. Close the first pop-up window by clicking the close button.
5. Close the second pop-up window.
6. **Verify** an element / text in the parent window and stop recording.
7. Run the test and observe the actions.

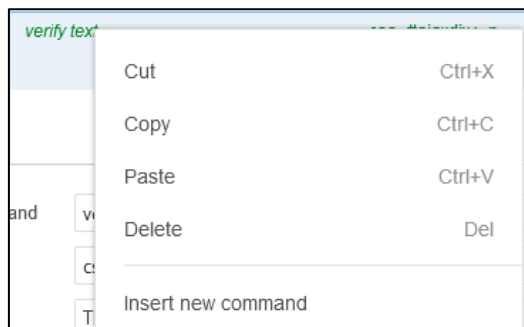
UECS2354 Software Testing
Lab 09: Introduction to Selenium

AJAX

1. Create a new test, set the URL to <http://book.theautomatedtester.co.uk/chapter1> and start recording.
2. Click on [Click this link to load a page with AJAX]. The page should be filled with some texts as the diagram below.



3. Verify the text [The following text....] and stop recording.
4. Run the recorded test.
5. If the test failed at **verify text** (in older Selenium IDE), it means the component contains the text is not loaded yet. New Selenium will wait automatically.
6. In order to wait for a component to be loaded / ready, we need to insert a command before **verify text**. Right-click on **verify text** and choose [Insert new command].



7. Choose [wait for element present] from the command dropdown list.

Command	wait for element present
Target	css=#ajaxdiv > p
Value	5000

8. As for the **Target**, use the same target in **verify text**.
9. **Value** is the timeout duration in millisecond.

UECS2354 Software Testing

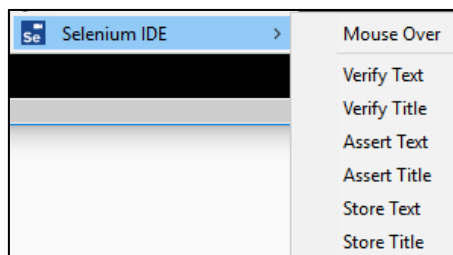
Lab 09: Introduction to Selenium

10. The command list in the test should be similar to the following diagram.

	Command	Target	Value
1	<i>open</i>	<i>http://book.theautomatedtester.co.uk/chapter1</i>	
2	// <i>set window size</i>	<i>998x753</i>	
3	<i>click</i>	<i>id=loadajax</i>	
4	<i>wait for element present</i>	<i>css=#ajaxdiv > p</i>	<i>5000</i>
5	<i>verify text</i>	<i>css=#ajaxdiv > p</i>	<i>The following text has been loaded from another page on this site. It has been loaded in an asynchronous fashion so tha</i>

Stored Value for Later Use

1. Value from a page can be stored for later use.
2. Create a new test, set the URL to <http://book.theautomatedtester.co.uk/chapter1> and start recording.
3. Right-click on the text [**Assert that this text in on the page**] and choose **Store Text**.



4. If Step 3 doesn't work, click on a blank line to enter a new command. Enter **store text** in the command input box or choose from the drop-down list.

	Command	Target	Value
1	<i>open</i>	<i>http://book.theautomatedtester.co.uk/chapter1</i>	
2	<i>set window size</i>	<i>1231x888</i>	
3	<i>store</i>		

Command

//

Target

Value

Description


5. Select the target text to be stored. Click on the button. Click on the text [**Assert that this text in on the page**].

UECS2354 Software Testing
Lab 09: Introduction to Selenium

6. Enter the variable name in the **Value** textbox, which can be accessed later by `${variable_name}`.

	Command	Target	Value
1	<i>open</i>	http://book.theautomatedtester.co.uk/chapter1	
2	<i>set window size</i>	1231x888	
3	<i>store text</i>	id=divontheleft	varText

Command	<input type="text" value="store text"/>	<input type="button" value="//"/>	<input type="button" value="🔍"/>
Target	<input type="text" value="id=divontheleft"/>	<input type="button" value="🔍"/>	<input type="button" value="🔍"/>
Value	<input type="text" value="varText"/>		

7. To write the stored value to a target, use the **type** command. Click the  button to select the target. Example, the textbox under the [Click this link to load a page with AJAX].

	Command	Target	Value
1	<i>open</i>	http://book.theautomatedtester.co.uk/chapter1	
2	<i>set window size</i>	1231x888	
3	<i>store text</i>	id=divontheleft	varText
4	<i>type</i>	id=storeinput	<code>\${varText}</code>

Command	<input type="text" value="type"/>	<input type="button" value="//"/>	<input type="button" value="🔍"/>
Target	<input type="text" value="id=storeinput"/>	<input type="button" value="🔍"/>	<input type="button" value="🔍"/>
Value	<input type="text" value="\${varText}"/>		

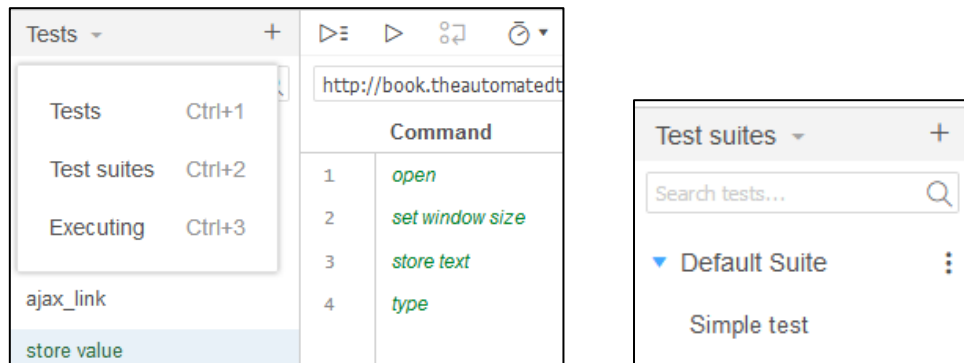
8. Run the test and observe the actions.

UECS2354 Software Testing

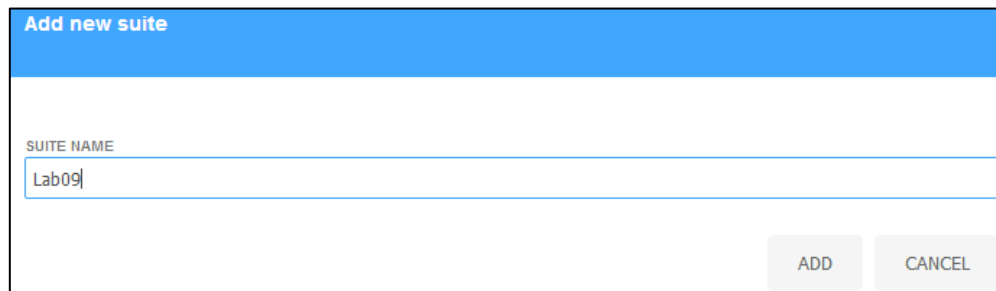
Lab 09: Introduction to Selenium


Combine all Tests in a Test Suite

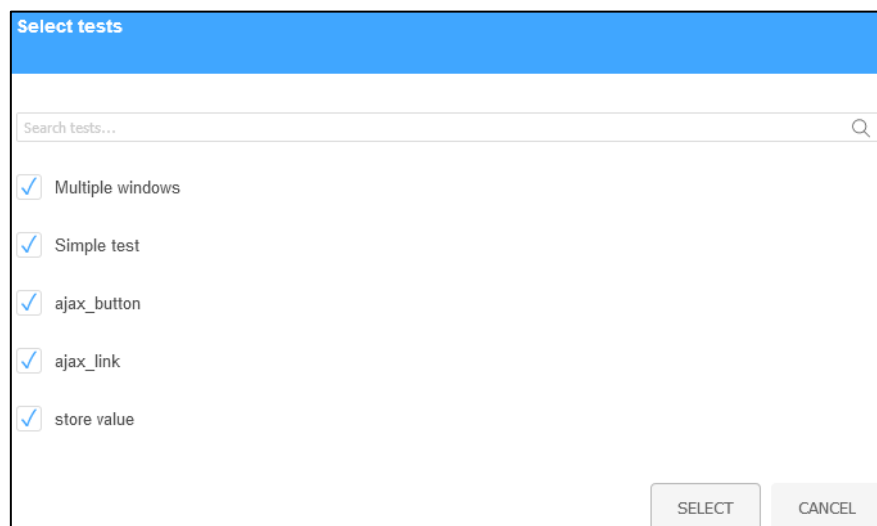
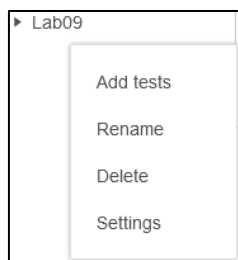
1. All the tests create are listed in the left panel. In order to switch to test suite, click on [Tests] on top of the panel and choose [Test Suite] or press **Ctrl+2**.



2. Click on the + button to add a new test suite.



3. Click on the  button next to test suite to add tests to the suite.



UECS2354 Software Testing
Lab 09: Introduction to Selenium

Exercise

Try to repeat all the above in Google Chrome.