

Test Automation

Lecture 11 –

Getting Started with Selenium IDE



IT Learning &
Outsourcing Center

Lector: Milen Strahinski
Skype: strahinski
E-mail: milen.strahinski@pragmatic.bg
Facebook: <http://www.facebook.com/LamerMan>
LinkedIn: <http://www.linkedin.com/pub/milen-strahinski/a/553/615>

www.pragmatic.bg

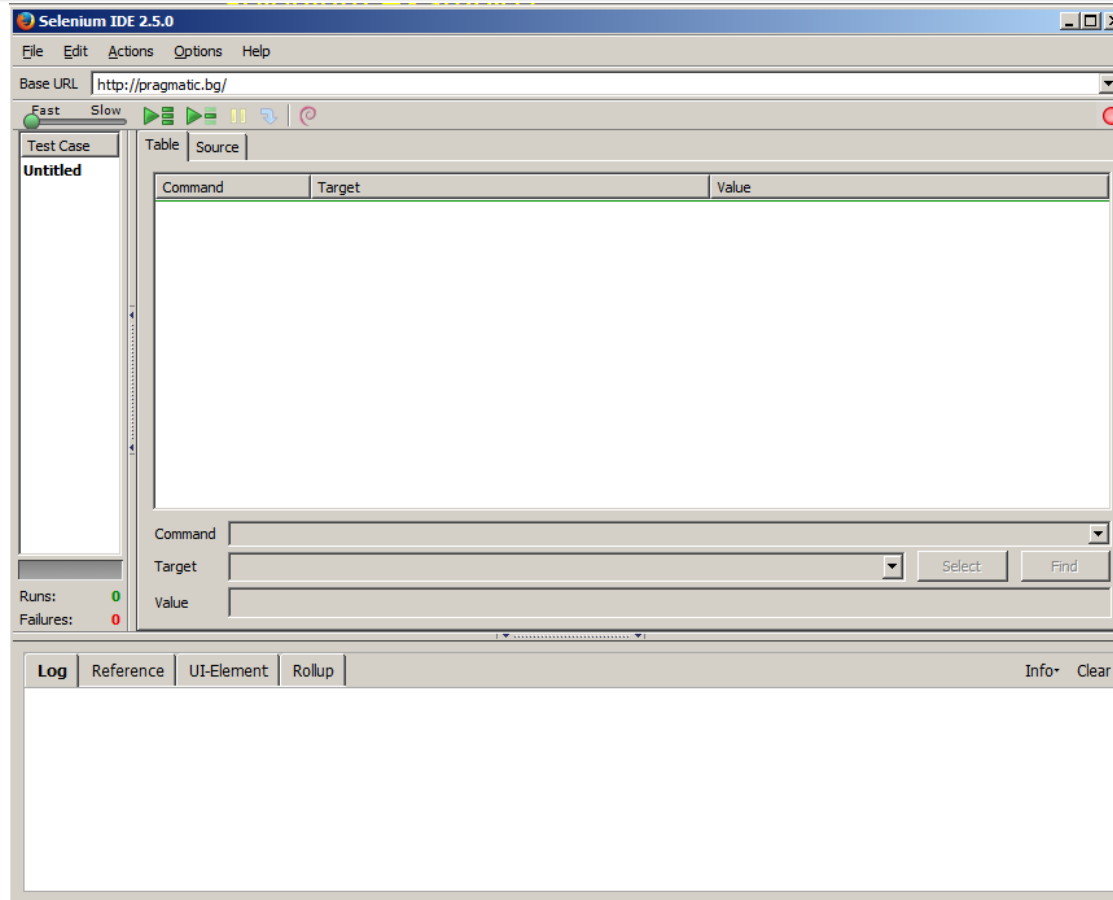


Summary - overall

- What is Selenium IDE
- Recording our first test
- Updating tests to work with AJAX sites
- Using variables in our tests
- Debugging tests
- Saving tests to be used later
- Creating and saving test suites
- Working with multiple windows
- Using one of the most famous extensions



Selenium IDE



- <http://release.seleniumhq.org/selenium-core/1.0/reference.html>



Rules for automation

- Tests should always have a known starting point. In the context of Selenium, this could mean opening a certain page to start a workflow.
- Tests should not have to rely on any other tests to run. If a test is going to add something, do not have a separate test to delete it. This is to ensure that if something goes wrong in one test, it will not mean you have a lot of unnecessary failures to check.
- Tests should only test one thing at a time.
- Tests should clean up after themselves.

Recording our first test



- Click on the red circle and open

<http://pragmatic.bg/automation/example1.html>



Assert / Verify (part 1)

- VerifyElementPresent
- assertElementPresent
- VerifyElementNotPresent
- assertElementNotPresent
- VerifyText
- assertText
- VerifyAttribute
- assertAttribute
- VerifyChecked
- assertChecked
- VerifyAlert
- assertAlert
- VerifyTitle
- assertTitle

```
open /chapter1
```

```
verifyTextPresent Assert that this text is on the page
```

```
verifyValue
```

```
storeValue
```

```
verifyText divontheleft Assert that this text is on the page
```

```
verifyElementPresent divontheleft
```

```
waitForElementPresent divontheleft
```

```
Show All Available Commands
```



Assert / Verify (part 2)

work-with-multiple-popup-windows.html - Selenium IDE 2.5.0

File Edit Actions Options Help

Base URL

Fast Slow

Test Case

work-with-m...

Table Source

Command	Target	Value
open	/automation	
clickAndWait	link=Example 1	
click	id=multiplewindow	
waitForPopUp	popupwindow	30000
selectWindow	name=popupwindow	
assertText	id=popuptext	Text within the pop up window
selectWindow	null	
click	css=div.multiplewindow2	
waitForPopUp	popupwindow2	30000
selectWindow	name=popupwindow2	
verifyTextPresent	Text within the pop up windowww	
selectWindow	name=popupwindow	
click	id=dosepopup	
selectWindow	name=popupwindow2	
click	id=dosepopup	

Command

Target

Value

Runs: 1

Failures: 1



Insert comments

C <u>u</u> t	Ctrl+X
C <u>o</u> py	Ctrl+C
P <u>a</u> ste	Ctrl+V
D <u>e</u> lete	Delete
Insert New Command	
Insert New Co <u>m</u> ment	
Clear <u>A</u> ll	
T <u>o</u> ggle B <u>r</u> eakpoint	B
S <u>e</u> t / Clear Start Point	S
E <u>x</u> ecute this command	X

store-variable-example.html - Selenium IDE 2.5.0

File Edit Actions Options Help

Base URL

Fast Slow

Table Source

Command	Target	Value
open	/automation/example1.html	
Starting main part of the test		
storeText	id=divontheleft	pragmaticVar
Man over board :)		
type	id=storeinput	\${pragmaticVar}

Command

Target

Value

Working with multiple windows (part 1)



- 1. Open up Selenium IDE in record mode and go to the **Example1** page on the site.
<http://pragmatic.bg/automation/example1.html>
- 2. Click on one of the elements on the page that has the text **Click this link to launch another window**. This will cause a small window to appear.
- 3. Verify the text in the popup by right-clicking and selecting VerifyText id=popup text within the popup window.
- 4. Once the window has loaded, click on the **Close the Window** text inside it.
- 5. Add a verify command for an element on the main page.

Working with multiple windows (part 2)



- `<div id='multiplewindow' class='multiplewindow' onClick='window.open("windowpopup.html","popupwindow","width=300,height=200");'>Click this link to launch another window</div>`
- **selectWindow(windowID)**
 - **title=My Special Window:** Finds the window using the text that appears in the title bar. Be careful; two windows can share the same title. If that happens, this locator will just pick one.
 - **name=myWindow:** Finds the window using its internal JavaScript "name" property. This is the second parameter "windowName" passed to the JavaScript method `window.open(url, windowName, windowFeatures, replaceFlag)` (which Selenium intercepts).
 - **var=variableName:** Some pop-up windows are unnamed (anonymous), but are associated with a JavaScript variable name in the current application window, e.g. `"window.foo = window.open(url);"`. In those cases, you can open the window using `"var=foo"`.

Working with multiple windows (part 3)



- Demonstration of a more complex example with 2 popup windows and a main window. Refer to the code examples in the lecture:

work-with-multiple-popup-windows.html

Selenium IDE tests against AJAX applications (part 1)

- 1. Start up Selenium IDE and make sure that the Record button is pressed.
- 2. Navigate to <http://pragmatic.bg/automation/example1.html>
- 3. Click on the text that says ***Click this link to load a page with AJAX.***
- 4. Verify the text that appears on your screen.
- Refer to the code examples in the lecture:
[example-WITHOUT-fix-for-the-ajax.html](#)
[example-WITH-fix-for-the-ajax.html](#)

Selenium IDE tests against AJAX applications (part 2)



- ~~W~~aitForAlertNotPresent
- ~~W~~aitForAlertPresent
- ~~W~~aitForElementPresent
- ~~W~~aitForElementNotPresent
- ~~W~~aitForTextPresent
- ~~W~~aitForTextNotPresent
- ~~W~~aitForPageToLoad
- ~~W~~aitForFrameToLoad
- and much more waitFor* in Selenium IDE

Storing information from the page in the test



- Let's try to store something into some variables
- Using the saved vars is done the following way:
`storedVars['variableName']` **or** `${variableName}`
- Refer to the code examples in the lecture:
[store-variable-example.html](#)



Debugging tests

- Using the pause and next step buttons
- Using *echo* command, it's equivalent to *Console.log* in *JavaScript*. For example, *echo | \${variableName}*.
- Using breakpoints

Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Delete
Insert New Command	
Insert New Comment	
Clear All	
Toggle Breakpoint	B
Set / Clear Start Point	S
Execute this command	X



Test Suites

- Creating new test suite
- Saving test suite
 - One thing to note is that saving a test suite does not save the test case. Make sure that you save the test case every time you make a change and not just the test suite.
- Play entire test suite



What you cannot record

- Silverlight and Flex/Flash applications
- HTML 5 - is not fully supported with Selenium IDE.
A good example of this is elements that have the *contentEditable=true* attribute.
- Selenium IDE cannot do file uploads.

Exporting Selenium IDE tests and results



- Exporting test cases
- Exporting test suites
- Exporting test case results...
- Exporting test suite results...

goto, gotoIF, while loop in Selenium IDE (part 1)



- <https://github.com/darrenderidder/sideflow>

The screenshot shows the Selenium IDE 1.0.12 interface. The Base URL is set to `https://github.com/`. The main table contains the following commands:

Command	Target	Value
store	10	x
gotolabel	target1	
runScript	alert("This should not appear")	
label	target1	
store	14	y
gotoIf	<code>\${y} > \${x}</code>	target2
runScript	alert("This also should not appear");	
label	target2	
while	<code>\${x} <= \${y}</code>	
store	<code>new Number(\${x} + 1)</code>	x
endWhile		
runScript	alert("Finished with x="+\${x}+" and y="+\${y});	

Below the table, there are input fields for Command, Target, and Value, along with a Find button.

The Log panel at the bottom shows the execution flow:

```
[info] Executing: |endwhile | |
[info] Executing: |while | ${x} <= ${y} | |
[info] Executing: |store | new Number(${x} + 1) | x |
[info] Executing: |endWhile | |
[info] Executing: |while | ${x} <= ${y} | |
[info] Executing: |store | new Number(${x} + 1) | x |
[info] Executing: |endWhile | |
[info] Executing: |while | ${x} <= ${y} | |
[info] Executing: |runScript | alert("Finished with x="+${x}+" and y="+${y}); | |
```

goto, gotoF, while loop in Selenium IDE (part 2)



- **label | mylabel** - creates a label called "mylabel" (a goto target)
- **goto | mylabel** - goto "mylabel"
- **gotoLabel | mylabel** - synonym for goto
- **gotof | expression** - jump to specified label if expression is true
- **while | expression** - loop while expression is true
- **endWhile** - indicate the end of a while loop
- **push | value | arrayName** - push value onto an array, creating array if necessary

goto, gotoIF, while loop in Selenium IDE (part 3)



- Lets have a look on the following examples:

[goto-gotoIF-whileloop-example.html](#)

[foothill-college-example-data-driven-data.html](#)

Useful Selenium IDE plugins (part 1)



- Favorites
 - <https://addons.mozilla.org/en-US/firefox/addon/favorites-selenium-ide/versions/>
- File Logging
 - <https://addons.mozilla.org/en-US/firefox/addon/file-logging-selenium-ide/versions/>
- Highlight Elements
 - <https://addons.mozilla.org/en-US/firefox/addon/highlight-elements-selenium-id/versions/>
- Log Search Bar
 - <https://addons.mozilla.org/en-US/firefox/addon/log-search-bar-selenium-ide/versions/>
- Page Coverage
 - <https://addons.mozilla.org/en-US/firefox/addon/page-coverage-selenium-ide/versions/>
- Power Debugger
 - <https://addons.mozilla.org/en-US/firefox/addon/power-debugger-selenium-ide/versions/>

Useful Selenium IDE plugins (part 2)



- Screenshot on Fail
 - <https://addons.mozilla.org/en-US/firefox/addon/screenshot-on-fail-selenium/versions/>
- Selenium Expert
 - <https://addons.mozilla.org/en-US/firefox/addon/selenium-expert-selenium-ide/versions/>
- Stored Variables Viewer
 - <https://addons.mozilla.org/en-US/firefox/addon/stored-variables-viewer-seleni/versions/>
- Test Results
 - <https://addons.mozilla.org/en-US/firefox/addon/test-results-selenium-ide/versions/>
- Test Suite Batch Converter
 - <https://addons.mozilla.org/en-US/firefox/addon/test-suite-batch-converter-sel/versions/>

Practice & Questions?



Practice exercises and Questions?