

M09 - Chapters 4 and 5

Selenium WebDriver Book

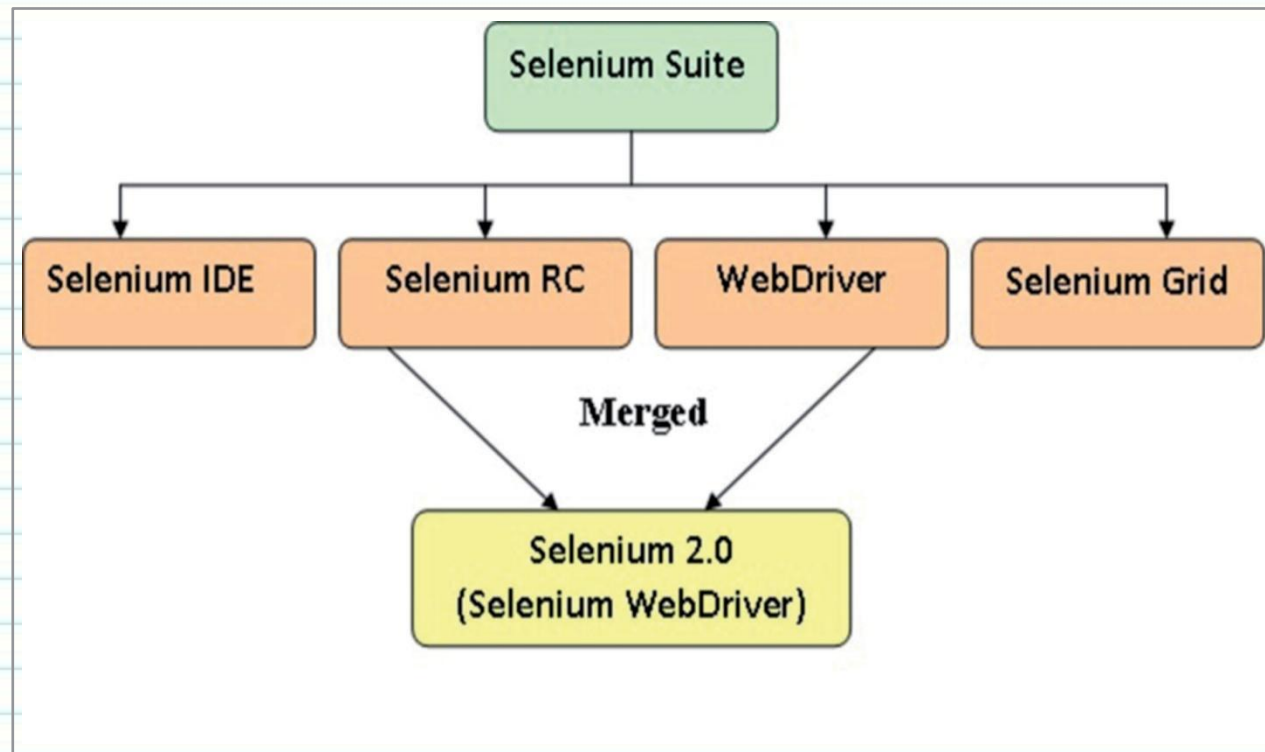
Dr. John H Robb, PMP, IEEE SEMC
UTA Computer Science and Engineering

Introduction to Selenium

- Selenium is an Open Source tool for automating browser-based applications.
 - It is a set of different software tools, each with a different approach to support test automation.
 - The tests can be written as HTML tables or coded in a number of popular programming languages and can be run directly in most modern Web browsers.
 - It can be deployed on Windows, Linux, and Macintosh and many OS for mobile applications like iOS, Windows Mobile, and Android.
- Among all Open Source tools, Selenium functional testing tool is considered to be a highly portable software testing framework and one of the **best tools available** in the current market for automation of Web applications.

The Selenium Tool Suite

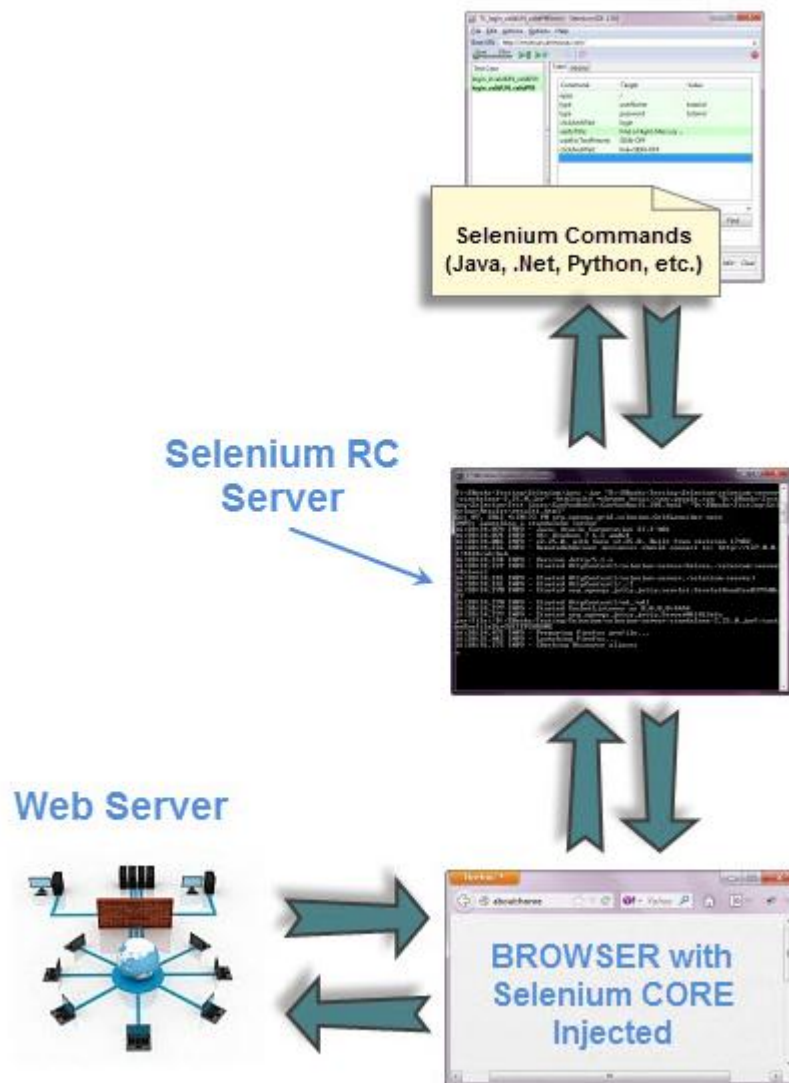
- Selenium is not just a single tool but a suite of software, each catering to different testing needs of an organization. **It has four components.**



Selenium's Tool Suite (cont.)

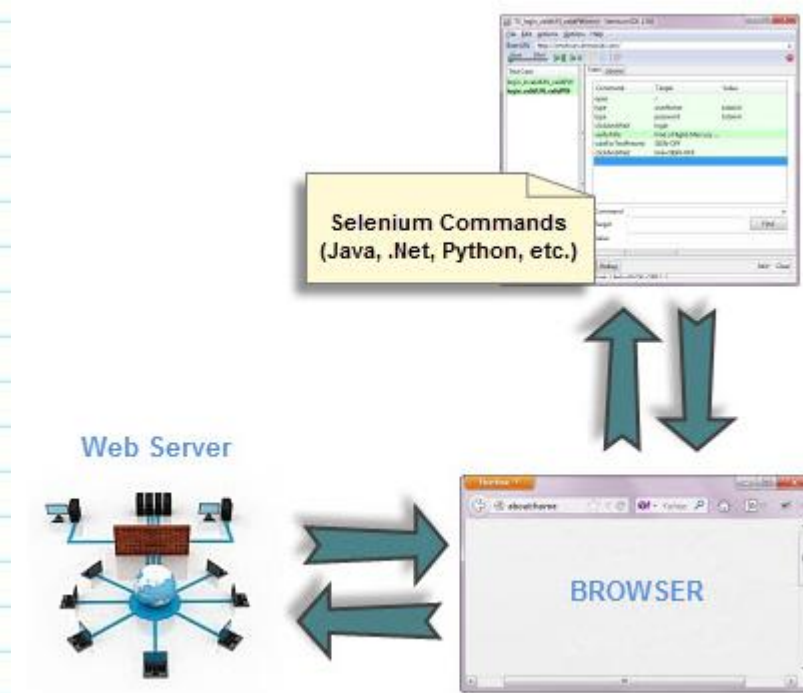
- **Selenium IDE (Integrated Development Environment)**
 - It is a prototyping tool for building test scripts. It comes as a Firefox and Chrome plug-in and provides an easy-to-use interface for developing automated tests.
 - It has a recording feature, which records user actions as they are performed and then exports them as a reusable script in one of many programming languages for execution later.
- **It is intended to be a rapid prototyping tool.**
 - It has a “Save” feature that allows users to keep the tests in a table-based format for later import and execution.
 - It doesn't provide iteration or conditional statements for test scripts.
 - Use It for basic automation.
 - Selenium developers usually recommend Selenium WebDriver to be used for serious, robust test automation.

Selenium's Tool Suite (cont.)



- **Selenium Commands sent to the RC server**
- **The RC server passes the Selenium command to the browser using Selenium-Core JavaScript commands.**
- **The browser, using its JavaScript interpreter, executes the Selenium command.**
- **Browser requests are sent to the Web Server in typical fashion.**

Selenium's Tool Suite (cont.)



- Code developed in one of many specific languages drive OS libraries to interact with the browser directly.
- It utilizes conditional operations and loops
- Several programming languages are supported: Java, .Net, PHP, Python, Perl, Ruby.
- All browsers are supported.
- All you need are your programming language's IDE (which contains your Selenium commands) and a browser.

Selenium's Tool Suite (cont.)

- **Selenium Grid**

- Selenium Grid is a tool used together with Selenium RC and WebDriver to run parallel tests across different machines and different browsers all at the same time. Parallel execution means running multiple tests at once.
- If you have a large test suite, or a slow-running test suite, you can boost its performance substantially by using Selenium Grid to divide your test suite to run different tests at the same time which will result in a significant time savings.
- Also, if you must run your test suite on multiple environments you can have different remote machines supporting and running your tests in them at the same time.
 - Run your tests against different browsers, operating systems, and machines all at the same time.

How to Choose the Right Selenium Tool

Tool	Why Choose?
Selenium IDE	<p>To learn about concepts on automated testing and Selenium, including: Selenese commands such as type, open, clickAndWait, assert, verify, etc.</p> <p>Locators such as id, name, xpath, css selector, etc.</p> <p>Executing customized JavaScript code using runScript</p> <p>Exporting test cases in various formats.</p> <p>To create tests with little or no prior knowledge in programming.</p> <p>To create simple test cases and test suites that you can export later to RC or WebDriver.</p> <p>To test a web application against Firefox only.</p>
WebDriver	<p>To use a certain programming language in designing your test case.</p> <p>To test applications that are rich in AJAX-based functionalities.</p> <p>To execute tests on the HtmlUnit browser.</p> <p>To create customized test results.</p>
Selenium Grid	<p>To run your Selenium RC scripts in multiple browsers and operating systems simultaneously.</p> <p>To run a huge test suite, that need to complete in soonest time possible.</p>

Advantages/Limitations of Selenium Tools

- **Advantages of Selenium**

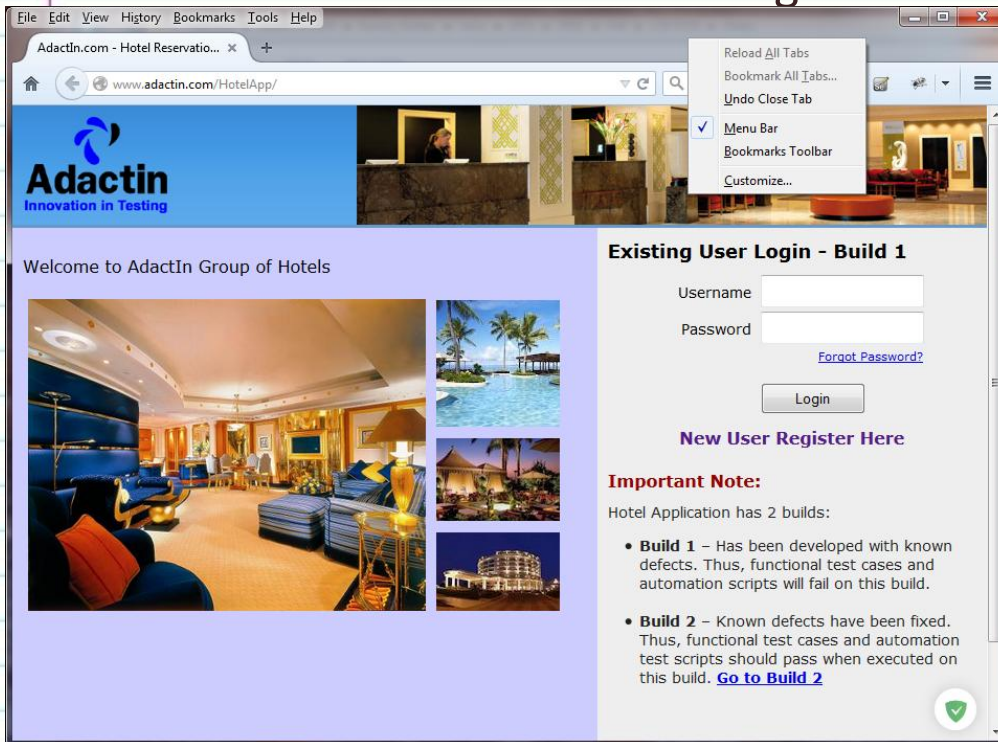
- Open Source tool.
- Supports all browsers like Internet Explorer, Firefox, Safari or Opera.
- Runs on all operating systems - Windows, Mac OS and Linux.
- Supports various languages like Java, .NET, Ruby, Perl, PHP, etc.
- Runs multiple tests at a time.
- Provides the option of using wide range of IDEs such as Eclipse, Netbeans, Visual Studio, etc., depending on the choice of development language.
- Primary source of UI testing for Android apps

- **Limitations of Selenium**

- Supports only browser based applications, not desktop/windows applications.
- Does not support file upload from local machine.
- Requires high technical skills to meet its full potential.
- Being an open source, Selenium has no official technical support.

Install Firefox Browser

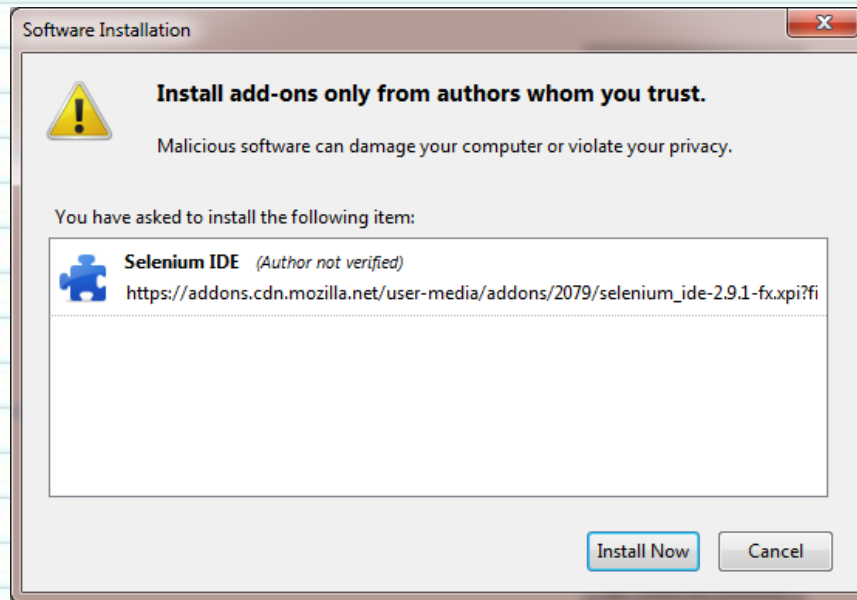
- Install Firefox browser (we need v39 to work with Selenium IDE v2)
 - <https://ftp.mozilla.org/pub/firefox/releases/39.0/>
 - Choose the right processor for you from this site
 - Disable updates
 - Options > Advanced > select “Never check for updates”
 - Uncheck “Use a background service to install updates”



- Turn on “menu bar” - right click on an empty space in above and check Menu Bar
- Set Homepage to adactin.com/HotelApp - Options > General - Startup section - “Home Page”

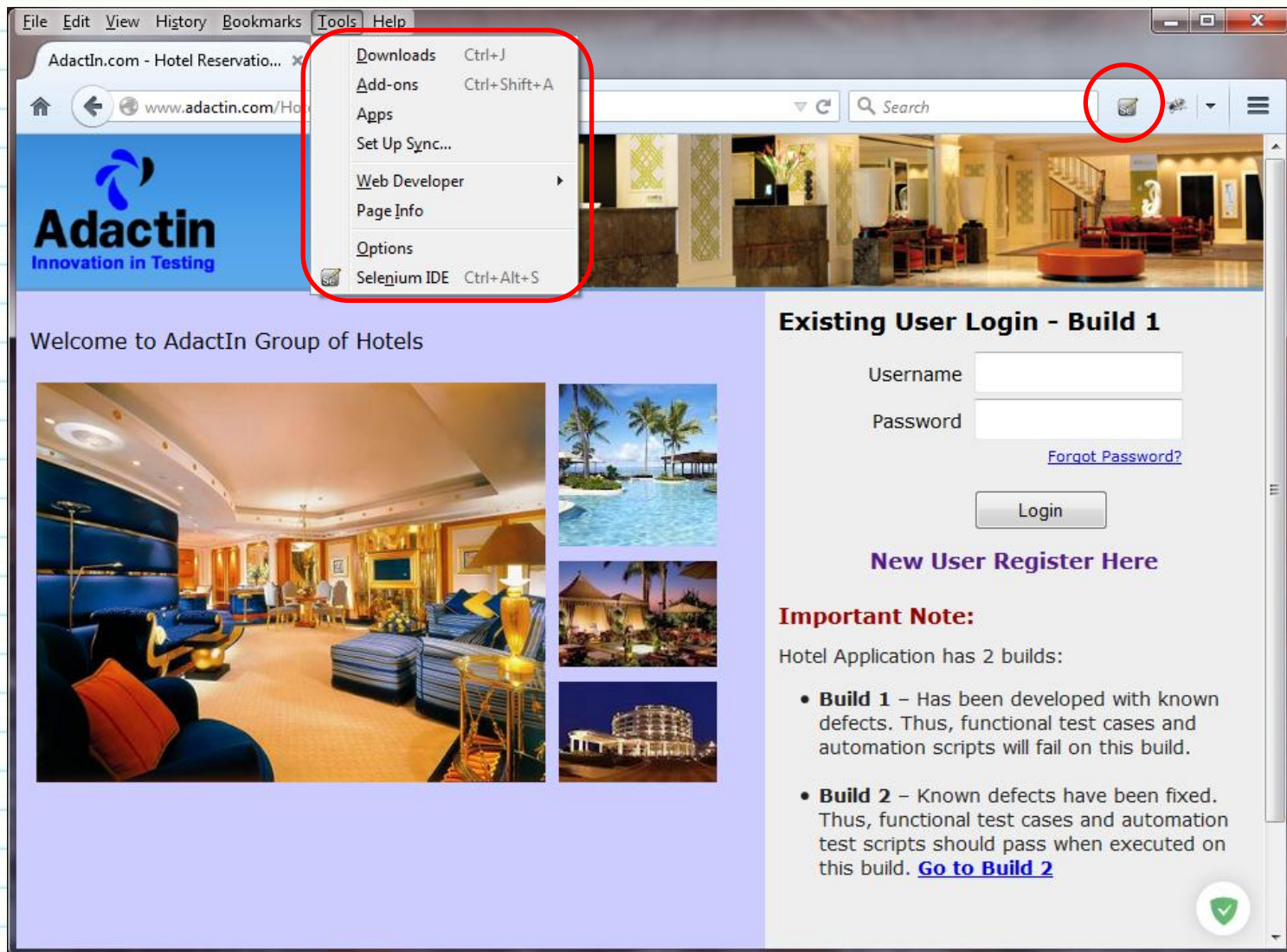
Install Selenium IDE

- Install Selenium IDE - get the file from Blackboard zip file
- Drag and drop onto a running Firefox browser



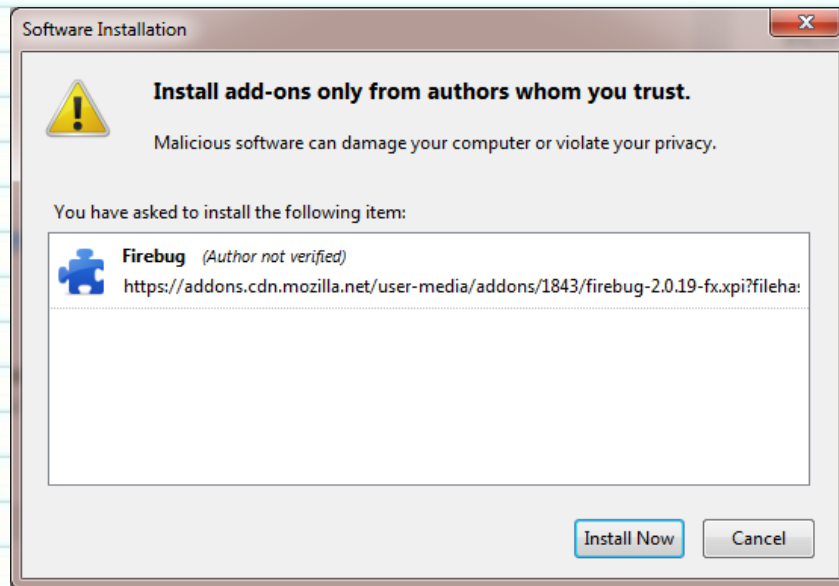
- Close and reboot Firefox
- After Firefox reboots you will find the Selenium-IDE icon on the upper right side of the browser window and also listed under the Firefox Tools menu. Go to **Tools → Selenium IDE**

Install Selenium IDE



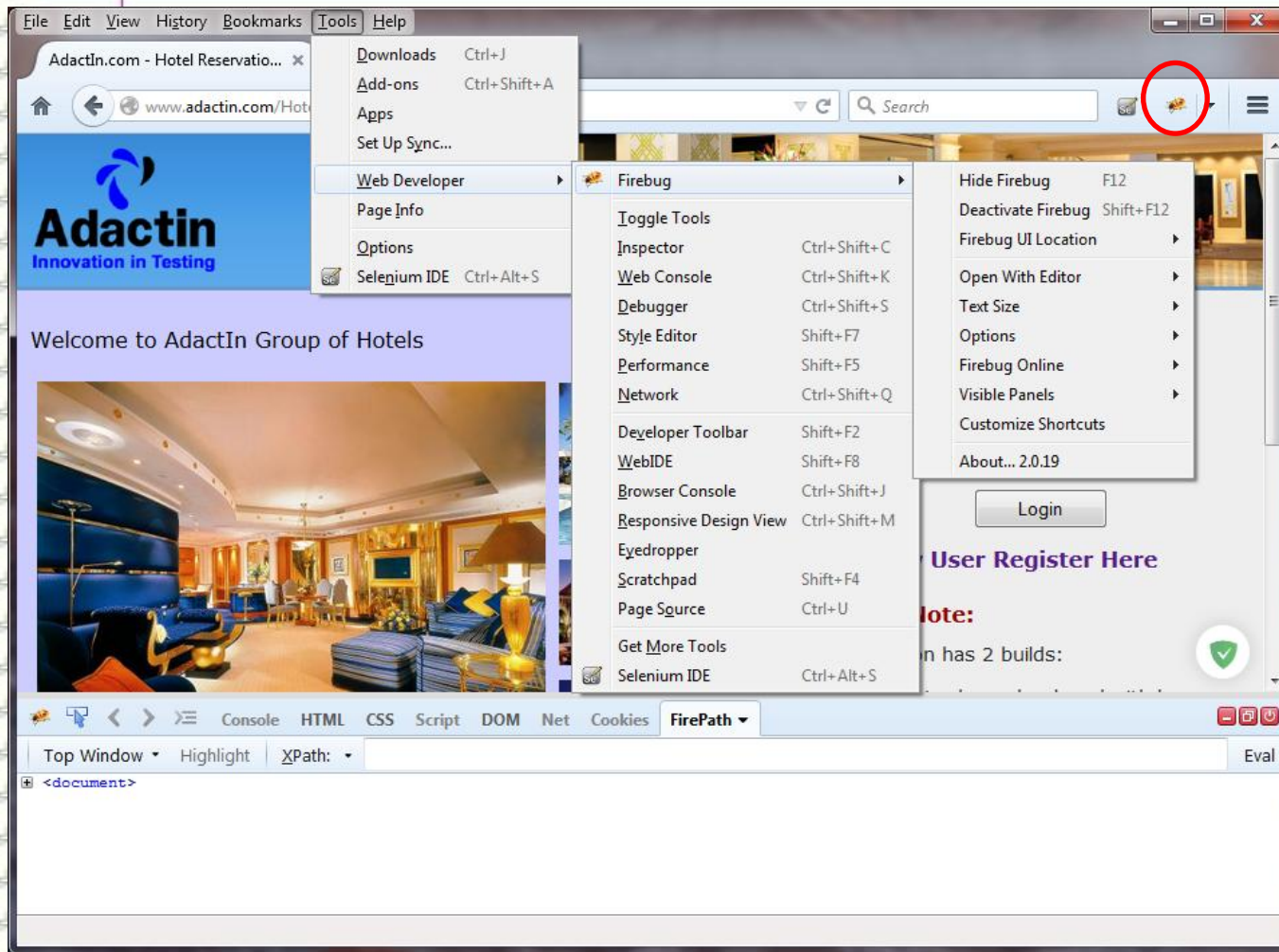
Install Firebug Plug-in

- Firebug integrates with Firefox to give access to Web development tools to edit, debug, and monitor CSS, HTML, and JavaScript live in any Web page.
- In Selenium, Firebug helps in inspecting UI elements and finding its associated properties and values.
- From the Blackboard files drag and drop the Firebug xpi onto a running Firefox browser



- Restart Firefox

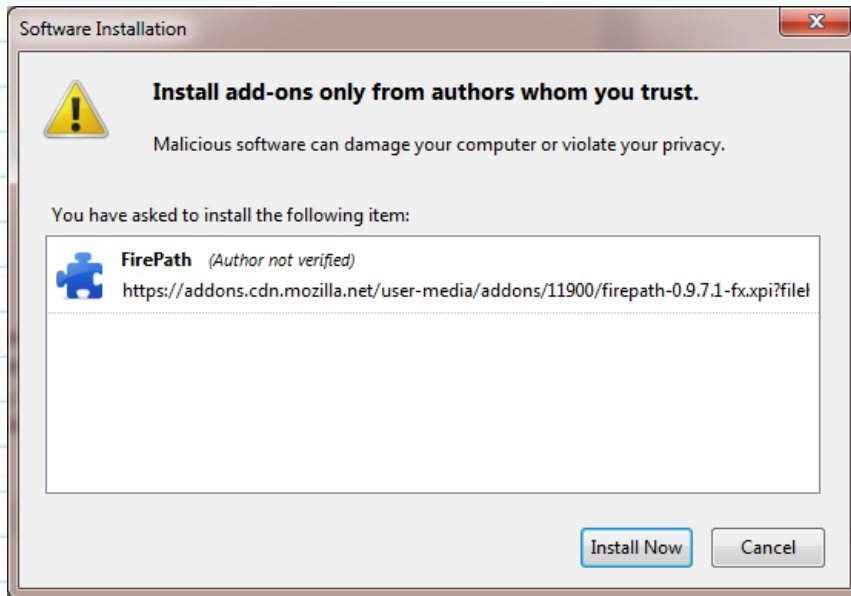
Install Firebug Plug-in (cont.)



Start by either clicking on the insect icon in the upper right or by the tool menu as shown

Install FirePath

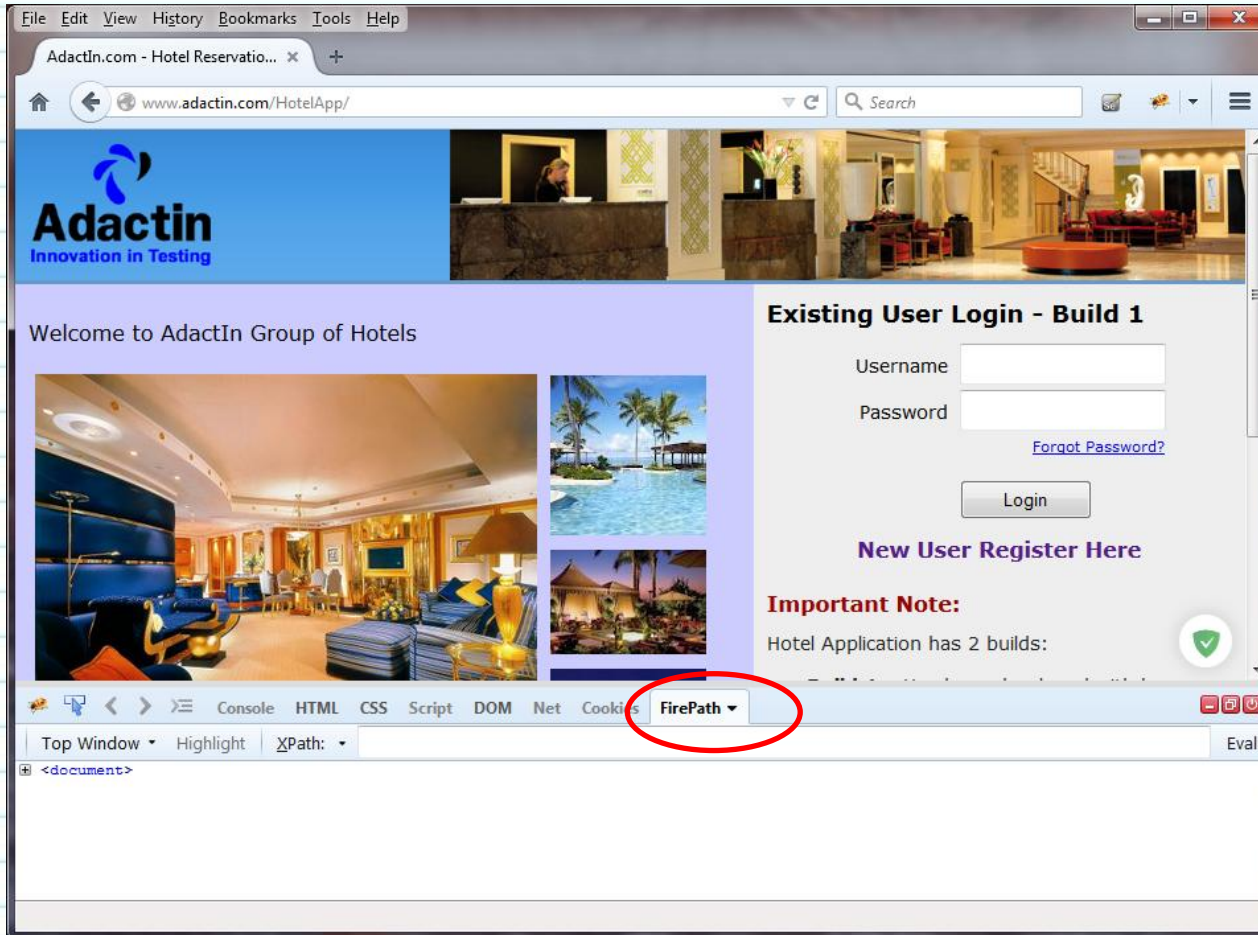
- FirePath is a Firebug extension that adds a development tool to edit, edit and retrieve the XPath from the UI extension.
- The XPath describes a way to locate an item in XML documents by using a tree representation of the document.
- From the Blackboard files drag and drop the FirePath xpi onto a running Firefox browser



- Restart Firefox

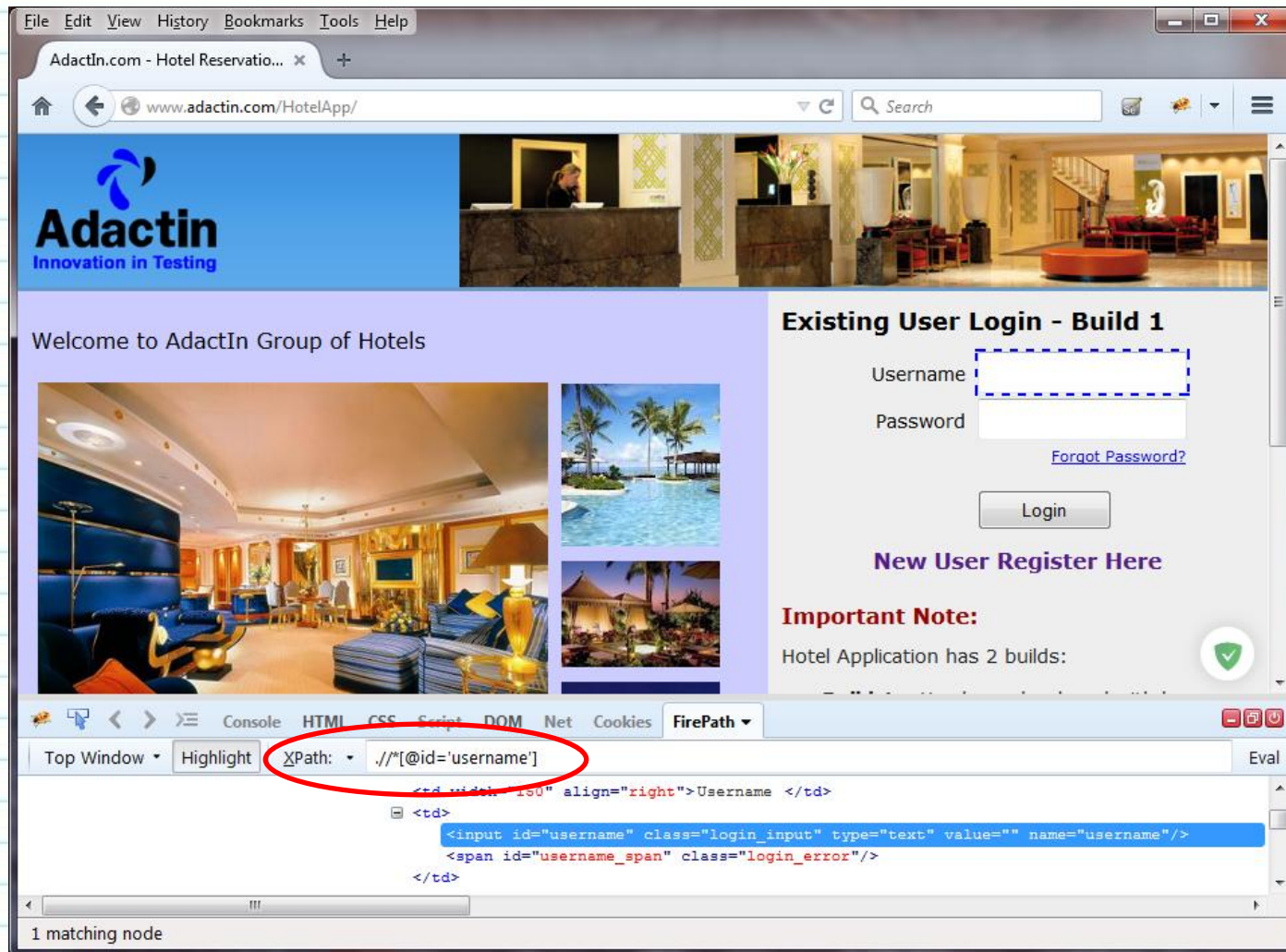
Install FirePath (cont.)

- Once in Firefox open Firebug
- You should now see the FirePath tab



Install FirePath (cont.)

- Select FirePath tab and view XPath field



Other Installations

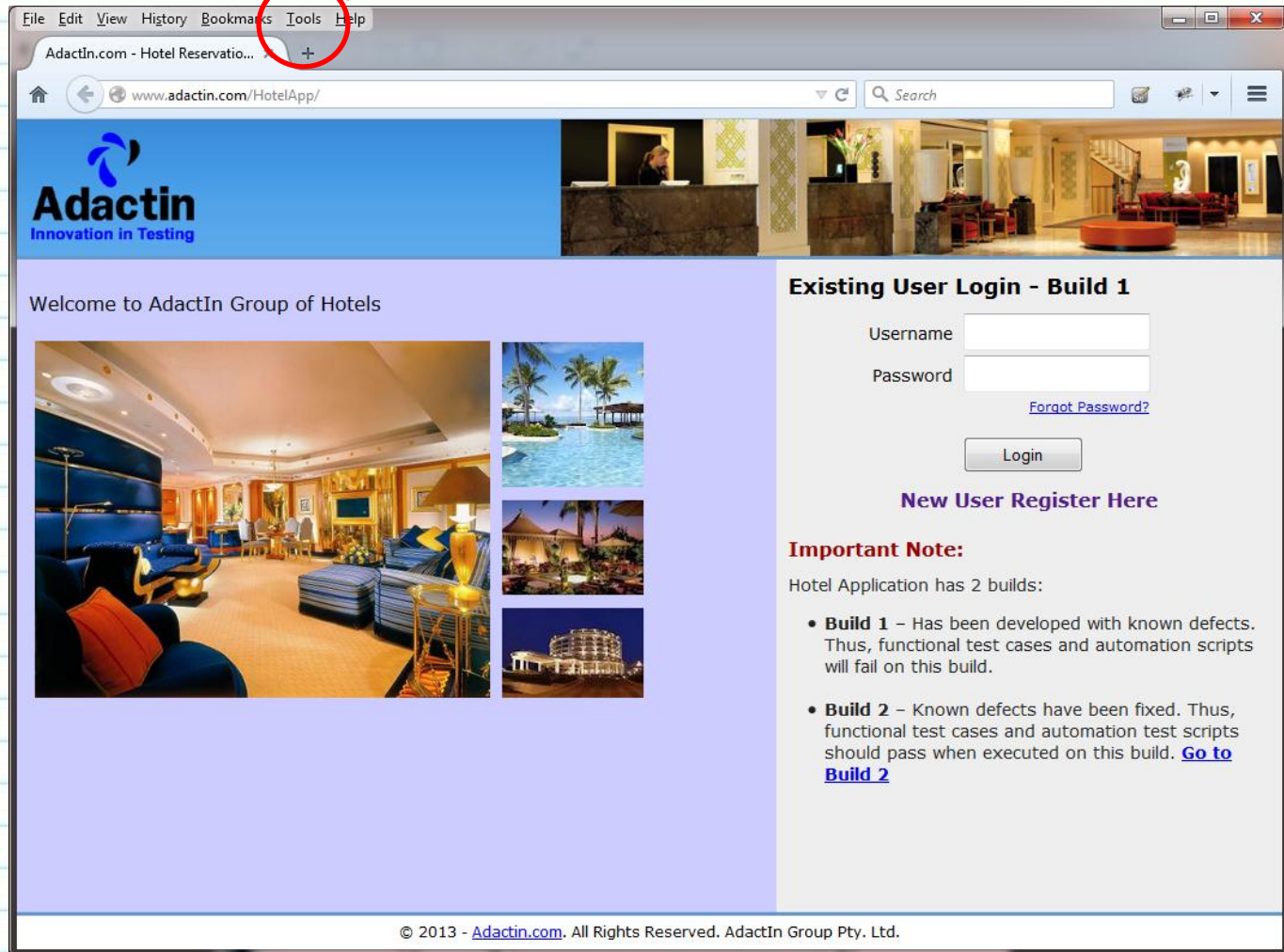
- Install WinANT v7(MacUsers see below)
- Windows users use file from Blackboard
- WinANT for Mac OS - <https://github.com/Benjol/SE-AutoReviewComments/wiki/Building#mac-os-x> replace link for step 3. with: <http://macappstore.org/ant-contrib/>

Introduction to Selenium IDE

- Many times functional testers believe that we need to write programs to automate applications. This is not entirely true.
- **Most** of the automation tools come with **record/replay features** which allow you to record user actions and replay those actions back **without writing a single line of code**.
- In Selenium we can perform record/replay and automate test cases using **Selenium IDE**.
- Selenium IDE is an easy-to-use Firefox plug-in and is generally a quick and efficient way to develop test cases.
- It also contains a context menu that allows you to first select a UI element from the browser's currently displayed page and then select from a list of Selenium commands with parameters pre-defined according to the context of the selected UI element.
- This lecture is all about understanding features of Selenium IDE and how to use it effectively.

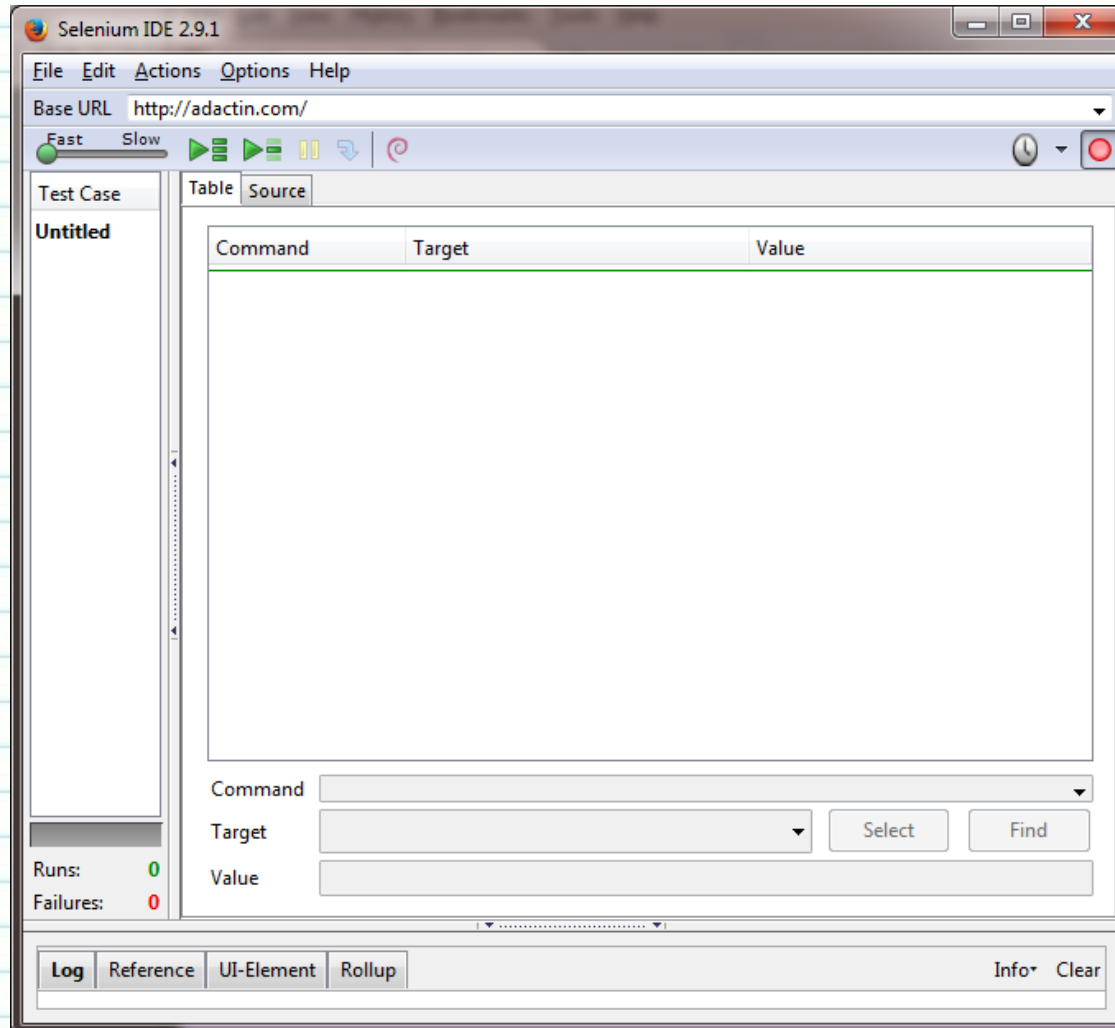
Selenium IDE Interface

- To launch the Selenium-IDE, select “Tools” then Selenium IDE



Selenium IDE Interface (cont.)

- It opens with an empty script-editing window and a menu for loading, or creating new test cases



Selenium IDE Interface (cont.)

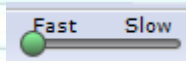
- **Menu Bar**

- The **File** menu has options for Test Case and Test Suite (suite of Test Cases). Using these you can add a new Test Case, open a Test Case, save a Test Case and export Test Case in a language of your choice. You can also open the recent Test Case. All these options are also available for Test Suite.
- The **Edit** menu allows copy, paste, delete, undo, and select all operations for editing the commands in your test case.
- The **Options** menu allows the changing of settings. You can set the timeout value for certain commands, add user-defined user extensions to the base set of Selenium commands, and specify the format (language) used when saving your test cases.
- The **Help** menu is the standard Firefox Help menu; only one item on this menu -> UI-Element Documentation pertains to Selenium-IDE.

Selenium IDE Interface (cont.)

- **Toolbar**

- The toolbar contains buttons for controlling the execution of your test cases.



- **Speed Control:** controls how fast your test case runs.



- **Run All:** Runs the entire test suite when a test suite with multiple test cases is loaded.



- **Run:** Runs the currently selected test. When only a single test is loaded this button and the Run All button have the same effect.



- **Pause/Resume:** Allows stopping and re-starting of a running test case.



- **Step:** Allows you to "step" through a test case by running it one command at a time. Use for debugging test cases.



- **Apply Rollup Rules:** allows repetitive sequences of Selenium commands to be grouped into a single action.

Selenium IDE Interface (cont.)

- **Toolbar**

- The toolbar contains buttons for controlling the execution of your test cases.



- Scheduler: Turns the scheduler on or off.



- Record: Records the user's browser actions. Note that this is enabled when IDE is started.

Recording Using Selenium IDE

- Note: Many first-time users begin by recording a test case from their interactions with a website.
- When Selenium-IDE is first opened the record button is ON by default.
- If you do not want Selenium-IDE to begin recording automatically you can turn this off by going under Options > Options... and deselecting **"Start recording immediately on open"**

Recording Using Selenium IDE (cont.)

- Recording your First Test Case
 1. Open Firefox browser and enter the following URL want test.
`http://www.adactin.com/HotelApp/`
 2. Click on Selenium IDE icon in upper right corner. Note that Selenium IDE is recording
 3. Assuming the application is open in Firefox browser perform the following steps:
 - a. Login (Use the username/password with which you have registered earlier).
 - b. Search for Hotel.
 - i. Select a location, e.g., Sydney
 - ii. Select number of rooms, e.g., 2-Two
 - iii. Select adults per rooms, e.g., 2-Two
 - iv. Click on Search button

Recording Using Selenium IDE (cont.)

c. Select a Hotel.

- i. Select one of the Hotel Radio buttons, e.g., select radio button next to Hotel Creek.

d. Book a Hotel.

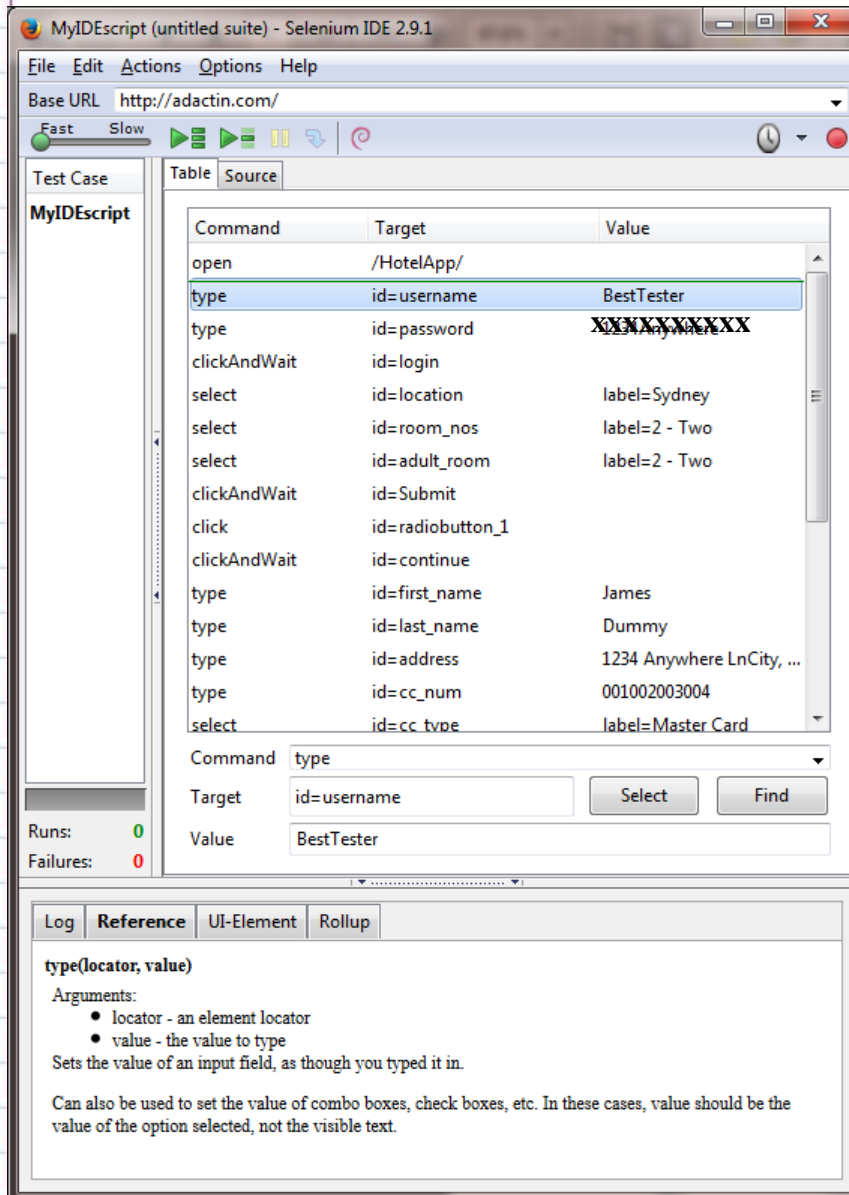
- i. Enter First Name - **Your first name**
- ii. Enter Last Name - **Your last name**
- iii. Enter Address - **Your address**
- iv. Enter 16-digit Credit Card no: - **Your UTA Id ending with 000000**
- v. Enter Credit Card type - **Amex**
- vi. Enter Expiry Month - **December**
- vii. Enter Expiry Year - **2020**
- viii. Enter CVV number - **Last 4 digits of your student ID**
- ix. Click on Book Now

e. After you see the Booking confirmation page, click on Logout link in the top right corner

Recording Using Selenium IDE (cont.)

- f. Click on "Click here to Login again" link to go back to Home page.
- 4. Stop recording by clicking on Stop Recording button in record toolbar .
- 5. Verify the steps that are recorded in Selenium IDE.
- 6. Save this as MyFirstSeleniumIDETest.side - **make sure to type the extension as .side**
- 7. **Please notice the very important principle that**
 - a. **each test starts from the main application page before login**
 - b. **each test ends at the main application page before login**
 - c. **each test leaves the application in a clean state**
- 8. Execute your test to make sure that it meets the previous criteria - if it does not delete any unnecessary instructions

Recording Using Selenium IDE (cont.)



The screenshot displays the Selenium IDE 2.9.1 interface. The main window shows a test suite named "MyIDEScript" with a table of recorded commands. The table has three columns: "Command", "Target", and "Value". The commands include "open", "type", "clickAndWait", "select", and "click". The "type" command is highlighted, showing its target as "id=username" and its value as "BestTester". Below the table, there are input fields for "Command", "Target", and "Value", along with "Select" and "Find" buttons. At the bottom, there is a "Log" tab showing the details of the selected command, including its arguments and a description of its functionality.

Command	Target	Value
open	/HotelApp/	
type	id=username	BestTester
type	id=password	XXXXXXXXXX
clickAndWait	id=login	
select	id=location	label=Sydney
select	id=room_nos	label=2 - Two
select	id=adult_room	label=2 - Two
clickAndWait	id=Submit	
click	id=radiobutton_1	
clickAndWait	id=continue	
type	id=first_name	James
type	id=last_name	Dummy
type	id=address	1234 Anywhere LnCity, ...
type	id=cc_num	001002003004
select	id=cc_type	label=Master Card

Command: type
Target: id=username
Value: BestTester

Runs: 0
Failures: 0

Log Reference UI-Element Rollup

type(locator, value)
Arguments:

- locator - an element locator
- value - the value to type

Sets the value of an input field, as though you typed it in.

Can also be used to set the value of combo boxes, check boxes, etc. In these cases, value should be the value of the option selected, not the visible text.

First column “Command”
represents the operation or method performed on user-controls.

Second column - “Target”
represents the user controls of the application on which actions are done.

Third column “Value” represents all the input data that has been entered into the application for testing.

Recording Using Selenium IDE (cont.)

- Note: note that all actions which we performed are captured as separate steps in the IDE script.
- During recording, Selenium IDE will automatically insert commands into your test case based on your actions. Typically this will include:
 - clicking a Web element like a button or link - *click* or *clickAndWait* commands
 - entering values - *type* command
 - selecting options from a drop-down list box - *select* command
 - clicking checkboxes or radio buttons - *click* command

Save and Replay the Script Using IDE

- **Saving the Test case**
- There are Save and Open commands under the File menu.
- Note: Selenium distinguishes between test cases and test suites.
- To save your Selenium-IDE tests for later use, you can either save the individual test cases or save the test suite.
- If the test cases of your test suite have not been saved, you'll be prompted to save them before saving the test suite.
 1. Select **File - Save Test Case**
 2. Save these test cases from this and the next few IDE lectures in your C:\Selenium Folder

Recording Using Selenium IDE (cont.)

- Select step of where location selection is Sydney

MyIDEscript (untitled suite) - Selenium IDE 2.9.1

File Edit Actions Options Help

Base URL <http://adactin.com/>

Fast Slow

Test Case: MyIDEscript

Command	Target	Value
open	/HotelApp/	
type	id=username	BestTester
type	id=password	XXXXXXXXXX
clickAndWait	id=login	
select	id=location	label=Sydney
select	id=room_nos	label=2 - Two
select	id=adult_room	label=2 - Two
clickAndWait	id=Submit	
click	id=radiobutton_1	
clickAndWait	id=continue	
type	id=first_name	James
type	id=last_name	Dummy
type	id=address	1234 Anywhere LnCity, ...
type	id=cc_num	001002003004
select	id=cc_type	label=Master Card

Command: select

Target: id=location

Value: label=Sydney

Runs: 0

Failures: 0

Note that in the bottom pane you can select values for Command, change Target and Data values.

Running Test Cases

- **Run a Test Case** - Click the Run button to run the currently displayed test case.
- **Run a Test Suite** - Click the Run All button to run all the test cases in the currently loaded test suite.
- **Stop and Start** - The Pause button can be used to stop the test case while it is running. The icon of this button then changes to indicate the Resume button. To continue click Resume.
- **Stop in the Middle** - You can set a breakpoint in the test case to cause it to stop on a particular command. This is useful for debugging your test case. To set a breakpoint, select a command, right-click, and from the context menu select Toggle Breakpoint.
- **Start from the Middle** - You can tell the IDE to begin running from a specific command in the middle of the test case. This also is used for debugging. To set a start point, select a command, right-click, and from the context menu select Set/Clear Start Point.

Running Test Cases (cont.)

- **Run Any Single Command** - Double-click any single command to run it by itself This is useful when writing a single command. It lets you immediately test a command you are constructing, when you are not sure if it is correct. You can double-click it to see if it runs correctly. This is also available from the context menu.
 3. Click on Run button and Run and verify the test case runs until completion
- Note: In case you have clicked on the
 - Logout button (at the bottom of Booking Page) instead of
 - Logout link (on the top right corner of Booking page)
- Your script might fail due to synchronization issue. Re-record your script and click on Logout link on top right corner of application.

Running Test Cases (cont.)

4. Verify the Results in the Log tab

The screenshot displays the Selenium IDE 2.9.1 interface. The 'Test Case' pane on the left shows 'MyIDEscript' with a green progress bar and 'Runs: 1', 'Failures: 0'. The 'Table' pane shows a list of commands and their targets/values. The 'Log' tab at the bottom shows a detailed execution log.

Command	Target	Value
clickAndWait	id=Submit	
click	id=radiobutton_1	
clickAndWait	id=continue	
type	id=first_name	James
type	id=last_name	Dummy
type	id=address	1234 Anywhere LnCity, St...
select	id=location	label=Sydney

Log Reference UI-Element Rollup Info Clear

```
[info] Playing test case MyIDEscript
[info] Executing: [open | /HotelApp/ | ]
[info] Executing: [type | id=username | BestTester |
[info] Executing: [type | id=password | 1234Anywhere |
[info] Executing: [clickAndWait | id=login | ]
[info] Executing: [select | id=location | label=Sydney |
[info] Executing: [select | id=room_nos | label=2 - Two |
[info] Executing: [select | id=adult_room | label=2 - Two |
[info] Executing: [clickAndWait | id=Submit | ]
[info] Executing: [click | id=radiobutton_1 | ]
[info] Executing: [clickAndWait | id=continue | ]
[info] Executing: [type | id=first_name | James |
[info] Executing: [type | id=last_name | Dummy |
[info] Executing: [type | id=address | 1234 Anywhere Ln City, St 12345 |
[info] Executing: [type | id=cc_num | 001002003004 |
[info] Executing: [select | id=cc_type | label=Master Card |
[info] Executing: [select | id=cc_exp_month | label=January |
[info] Executing: [select | id=cc_exp_year | label=2019 |
[info] Executing: [type | id=cc_cvv | 123 |
[info] Executing: [click | id=book_now | ]
[info] Executing: [type | id=cc_num | 0001000200030004 |
[info] Executing: [click | id=book_now | ]
[info] Executing: [clickAndWait | link=Logout | ]
[info] Executing: [clickAndWait | link=Click here to login again | ]
[info] Test case passed
```

Running Test Cases (cont.)

- **Log/Reference/UI-Element/Rollup Pane**
 - The bottom pane is used for four different functions - Log, Reference, UI-Element, and Rollup-depending on which tab is selected.
- **Log**
 - When you run your test case, error messages and information messages showing the
 - progress are displayed in this pane automatically, even if you do not first select the Log tab. These messages are often useful for test case debugging. Notice the Clear button for clearing the Log. Also notice the Info button is a drop-down allowing selection of different levels of information to log.

Running Test Cases (cont.)

- **Reference tab**

- The reference tab is the default selection whenever you are entering or modifying Selanese commands and parameters in table mode.
- While the reference pane is invaluable as a quick reference it is often necessary to consult the Selenium Reference documentation.

- **UI-Element tab**

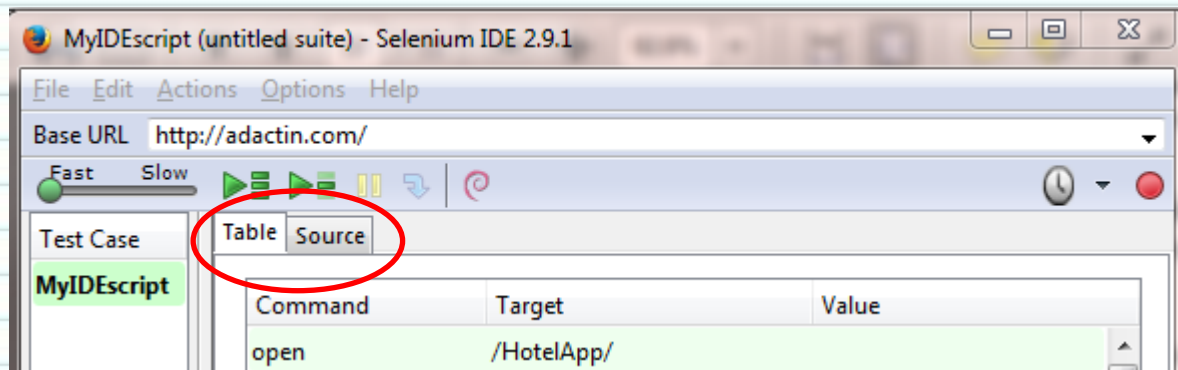
- UI-Element is a mapping between a meaningful name for a page element, and the means to locate that page element's DOM node.
- The page element is located via a locator. UI elements belong to page sets.

- **Roll Up tab**

- It is an optional piece of logic that modifies the command expansion of a roll up.

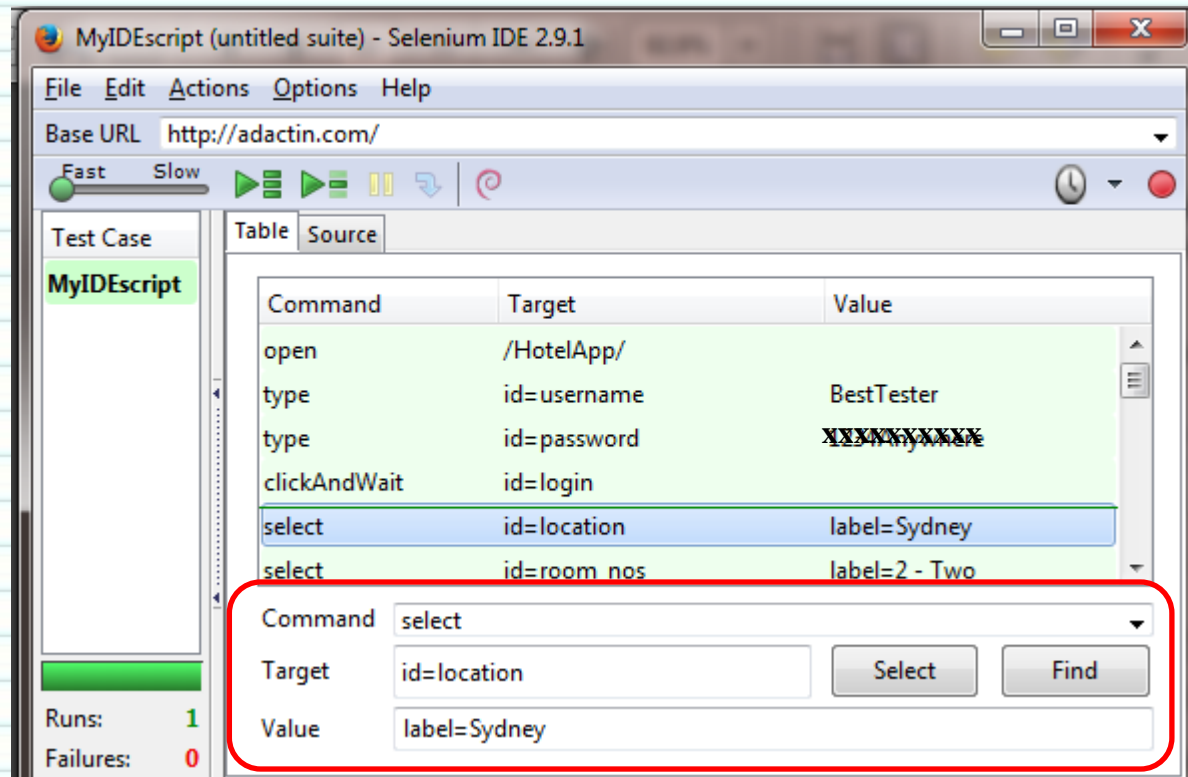
Inserting/Editing Steps Manually

- You can not only edit the steps inside the selenium IDE but can also add new steps or change the sequence of steps inside the IDE itself.
- To do that we need to understand the following:
- **Test case table Pane:** your script is displayed in the test case table pane. It has two tabs, one for displaying the command and their parameters in a readable table format. The other tab is described below.
- **Test case Source Pane:** Source displays the test case in the native format in which the file will be stored. By default, this is HTML although it can be changed to a programming language such as Java or C sharp, or a scripting language like python. This allows one to edit the test case and its raw form, including copy, cut and paste operations.



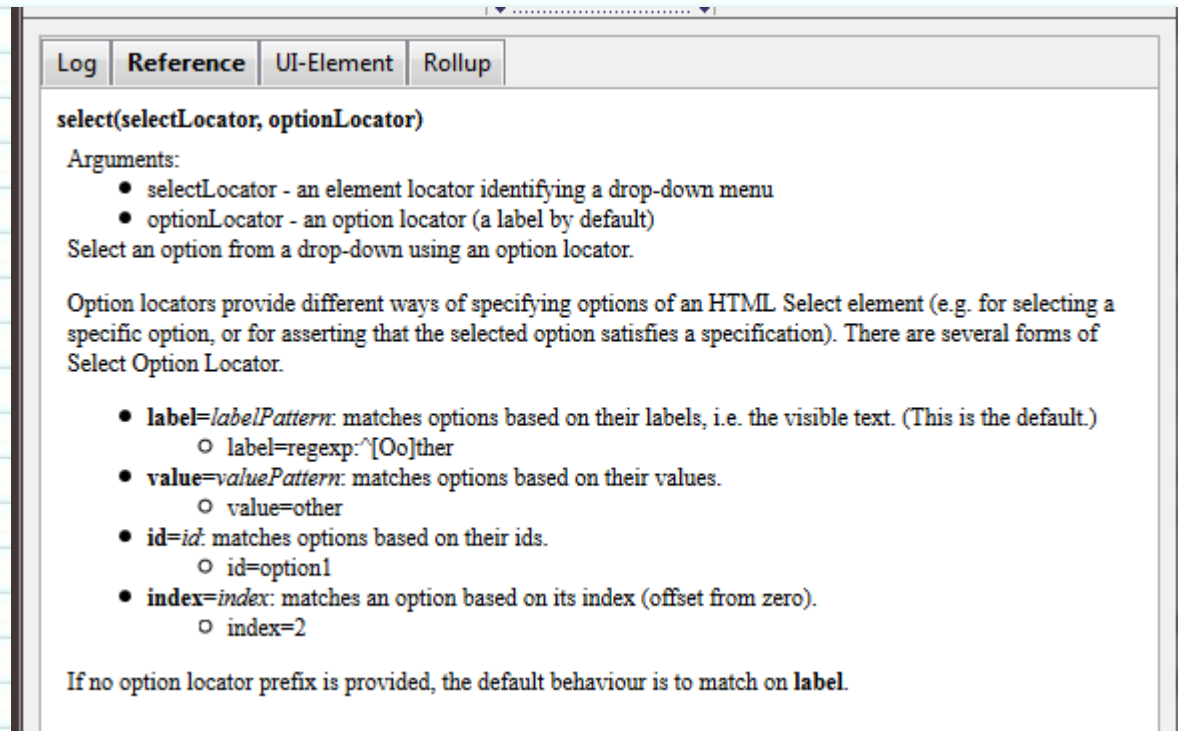
Inserting/Editing Steps Manually (cont.)

- **Operations Pane** - The **Command**, **Target**, and **Value** entry fields display the currently selected command along with its parameters. The user can modify Command, Target or Value in this pane.
- If you select the step where we selected location as Sydney you would see Command, Target and Value as in below snapshot.



Inserting/Editing Steps Manually (cont.)

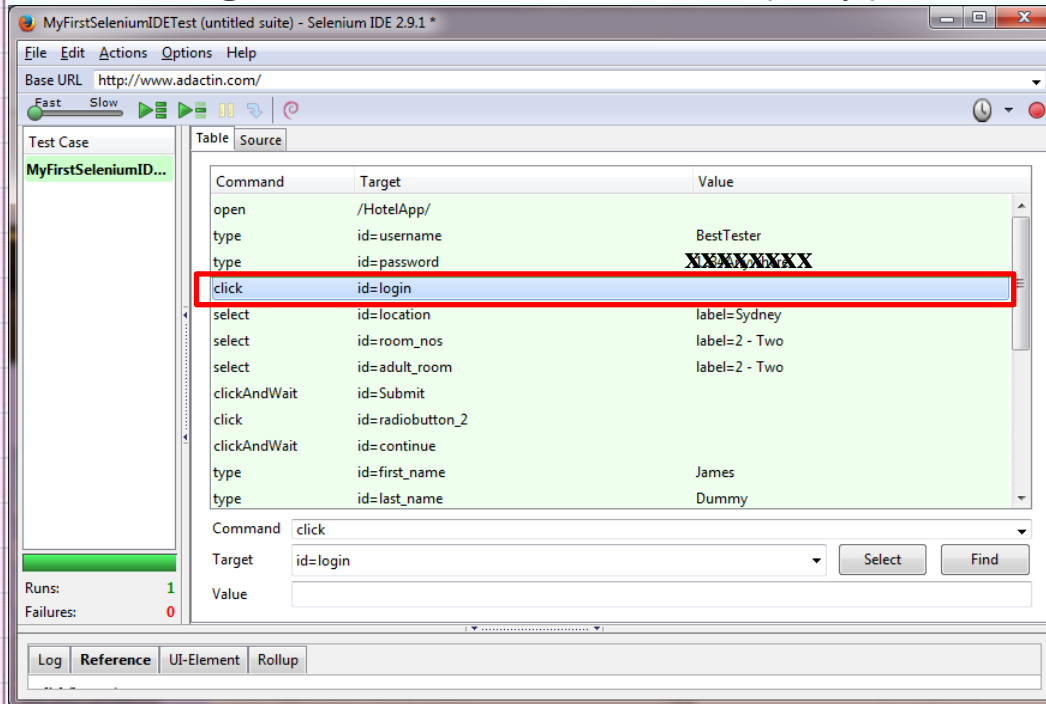
- Also, if you select the Reference Tab you will see details on the command and arguments that should be entered for that command (in this case the select command)



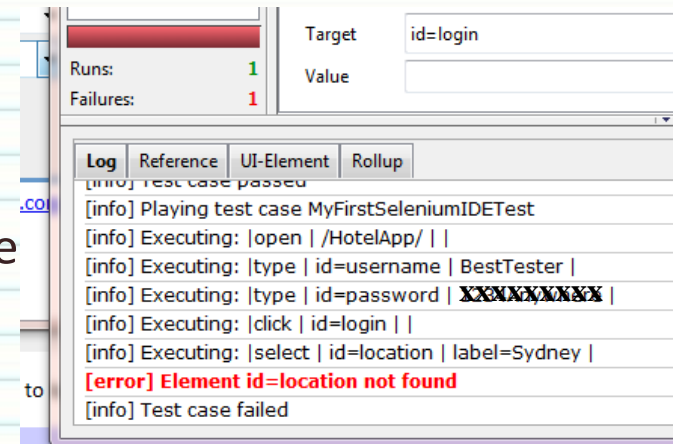
- If you start typing in the Command field, a drop-down list will be populated based on the first characters you type; you can then select your desired command from the drop-down.

In-Class Work

- Change the fourth instruction (step) from **clickAndWait** to **click**



- re-run tests - notice the test fails
- Click on the log tab in the bottom roll-up pane
- See what the error message says here
- This is a common script error message
- What was the failure here?



In-Class Work (cont.)

- The failure was that the test scripts run so fast that the page had not yet loaded and it tried to find an element on a page that wasn't present yet
- Change the instruction back to `clickAndWait`
- Notice all the other `clickAndWait` instructions in your test script
- Logout of your application and return to the main application “click here to login again”
- Re-run your script to make sure it passes

Adding Verifications/Asserts

- So far all we have done is to play a script but tests need to check expected outputs against actual output.
- Your test cases will also need to check the properties of a Web page. To do the same in your script you can insert assert or verify commands in Selenium IDE.
- Both of these commands work to compare results, with different responses to the failure of a test condition.

Command	Response
Assert	"assert" will fail the test and abort the current test case
Verify	"verify" will fail the test and continue the current test case

Adding Verifications/Asserts (cont.)

- Problem: As an example, let us say we want to verify that our login is successful.
- We can verify this by checking that the Logout link exists once a user logs in. To add a verification point:
 1. Open your IDE test case **MyFirstSeleniumIDETest.side**
 2. Click on File > Save Test Case As...
 3. Save the Testcase as **IDEVerificationScript**
 4. Select the step where the user selects the location Sydney
 5. Select Actions > Toggle Breakpoint to **insert a Breakpoint**
 6. Run a test via Selenium IDE to go to the browser displaying your test application by selecting **Run**. You will notice that the script runs until login and pauses.
 7. In the application you will notice that user is logged in. Now we want to verify if the logout link exists.
 8. Place your cursor on top of Logout, **right click** and select **Show All Available Commands**.

Adding Verifications/Asserts (cont.)

9. Since we want to verify that logout link is present and if logout link is not present (which means we are not successfully logged in) we would want to abort the script. We'll insert **AssertElementPresent link=logout** command. Select this command from the list.
 10. In the IDE script, you will notice a new statement added "assertElementPresent"
 11. File > Save Test Case
 12. You can resume the test by either clicking on resume icon or selecting **Actions > Pause/Resume**.
 13. **Rerun the test to confirm that the search element present statement got executed correctly and verifies the logout link presence.**
- Note: Also keep a note of other verify and assert commands like:
 - VerifyTextPresent (verifies the text is present on the page)
 - VerifyTitle (verifies title of the page).

Adding Verifications/Asserts (cont.)

The screenshot shows the Selenium IDE 2.9.1 interface. The 'Test Case' pane on the left is titled 'IDEVerificationScript'. The main table lists the following commands:

Command	Target	Value
open	/HotelApp/	
type	id=username	BestTester
type	id=password	XXXXXXXXXX
clickAndWait	id=login	
assertElementPresent	link=Logout	
select	id=location	label=Sydney
select	id=room_nos	label=2 - Two
select	id=adult_room	label=2 - Two
clickAndWait	id=Submit	
click	id=radiobutton_1	
clickAndWait	id=continue	
type	id=first_name	James
type	id=last_name	Dummy
type	id=address	1234 Anywhere LnCity, St 1...
type	id=cc_num	001002003004

Below the table, there are input fields for 'Command', 'Target', and 'Value', along with 'Select' and 'Find' buttons. At the bottom, there is a 'Log' tab and a 'Reference' tab, with a 'UI-Element' tab selected. The 'Log' pane shows 'Runs: 0' and 'Failures: 0'.

AssertElementPresent
link=logout command

Exercise

- Submit the attached exercise (from the Canvas Exercises Folder) as **E06 - Exercise for M09.zip** in Canvas