Rapise®

Rapise® | Quick Start Guide

An Introduction to Testing Web Applications with Rapise

Date: May 8th, 2017

inflectra®

Contents

Introduction

Rapise® is a next generation software test automation tool that leverages the power of open architecture to improve application quality and reduce time to market.

This guide provides a quick step-by-step tutorial for creating a sample Rapise automated test against a web application and playing the results back.

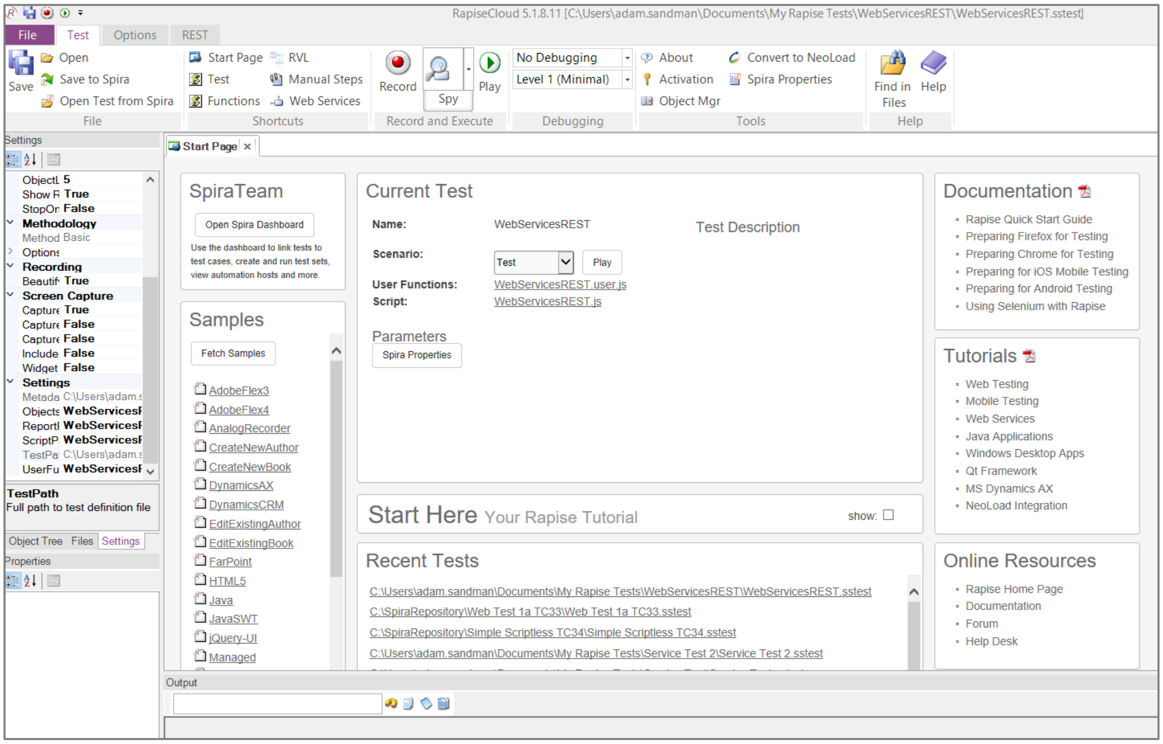
For further information on using Rapise, please refer to the more comprehensive *Rapise User Guide*. For information on using Rapise in conjunction with our SpiraTest test management system, please refer to the *Using Rapise with SpiraTest Guide.*

1. Recording Your First Script

In this section, you will learn how to record and execute a Rapise script. We will be using a demo web application called **Library Information System**. Our test will be simple. It will log on to the library catalog, navigate to the main menu, and click on all of the menu options to make sure the links are working.

1.1. Open Rapise

Go to Start **>** All Programs **>** Inflectra **>** Rapise**.** The following window should appear:



RapiseCloud 5.1.8.11 [C:\Users\adam.sandman\Documents\My Rapise Tests\WebServicesREST\WebServicesREST.sstest]

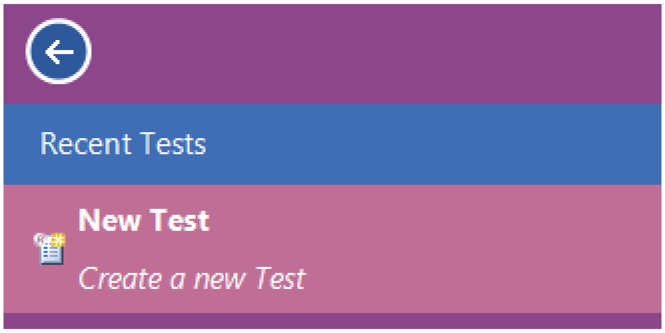
Name: WebServicesREST

Scenario: Test

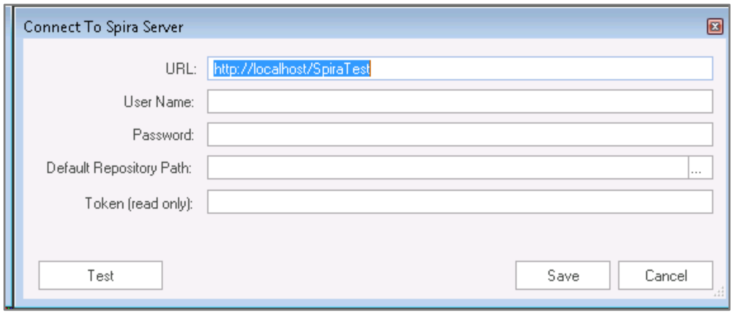
User Functions: WebServicesREST.user.js

Script: WebServicesREST.js

Click on the **File** tab in main menu and then click on the option to ‘Create New Test’:



If this is your first time using Rapise on this computer, you may see the following dialog box:



Connect To Spira Server

URL: http://localhost/SpiraTest

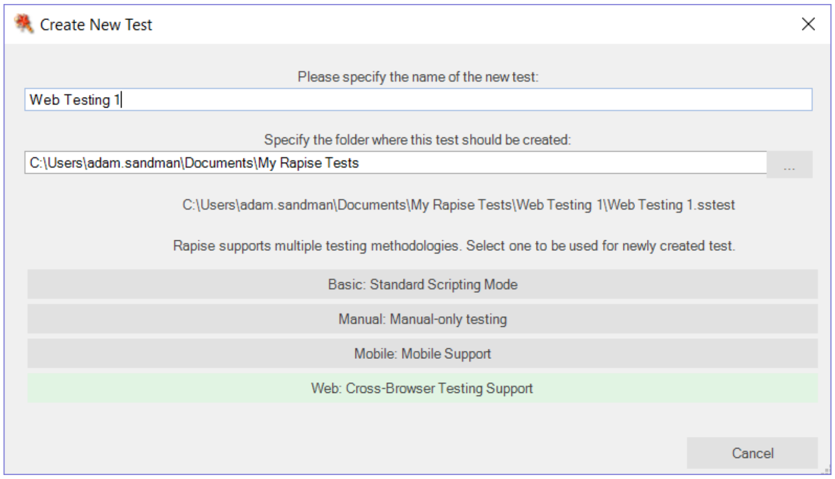
User Name:

Password:

Default Repository Path:

Token (read only):

If you see this, it means that Rapise is trying to connect to a SpiraTest server. SpiraTest is our web based test management system. It is a powerful tool that can store your Rapise tests and deploy them onto remote machines for automated regression testing. However, for now just click on the **[Cancel]** button and you will see the 'new test dialog':



Create New Test

Please specify the name of the new test: Web Testing 1

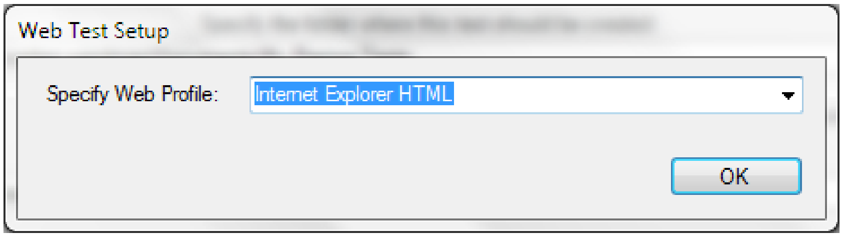
Specify the folder where this test should be created: C:\Users\adam.sandman\Documents\My Rapise Tests

C:\Users\adam.sandman\Documents\My Rapise Tests\Web Testing 1\Web Testing 1.sstest

Rapise supports multiple testing methodologies. Select one to be used for newly created test.

Basic: Standard Scripting Mode
Manual: Manual-only testing
Mobile: Mobile Support
Web: Cross-Browser Testing Support
Cancel

Now enter the name of your new test **'Web Testing 1'**, make sure the Methodology is set to **'Web: Cross-Browser Testing Support'** and click **[Create]**. Since you chose a web test, you will now need to choose the initial web browser profile (don't worry you can easily change it later):

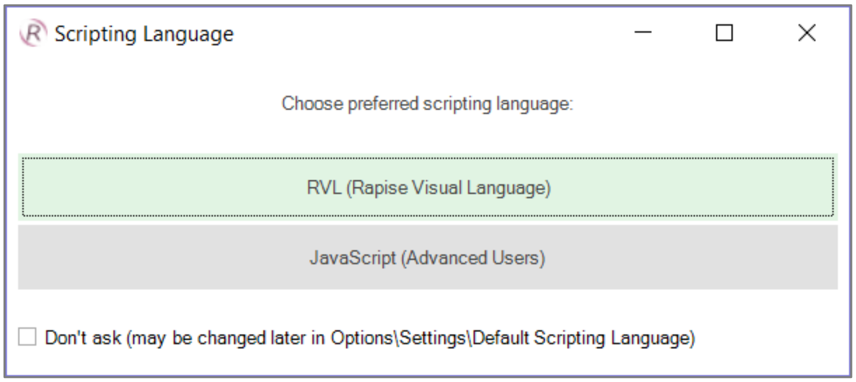


Web Test Setup

Specify Web Profile: Internet Explorer HTML

Choose **'Internet Explorer HTML'** from the list of options.

Next you will be asked if you want to create your tests using the scriptless **Rapise Visual Language (RVL)** technology or using JavaScript. For this example we will use the RVL scriptless approach. If you're interested in creating the test using JavaScript instead, please refer to **Appendix C – Using JavaScript Tests**.



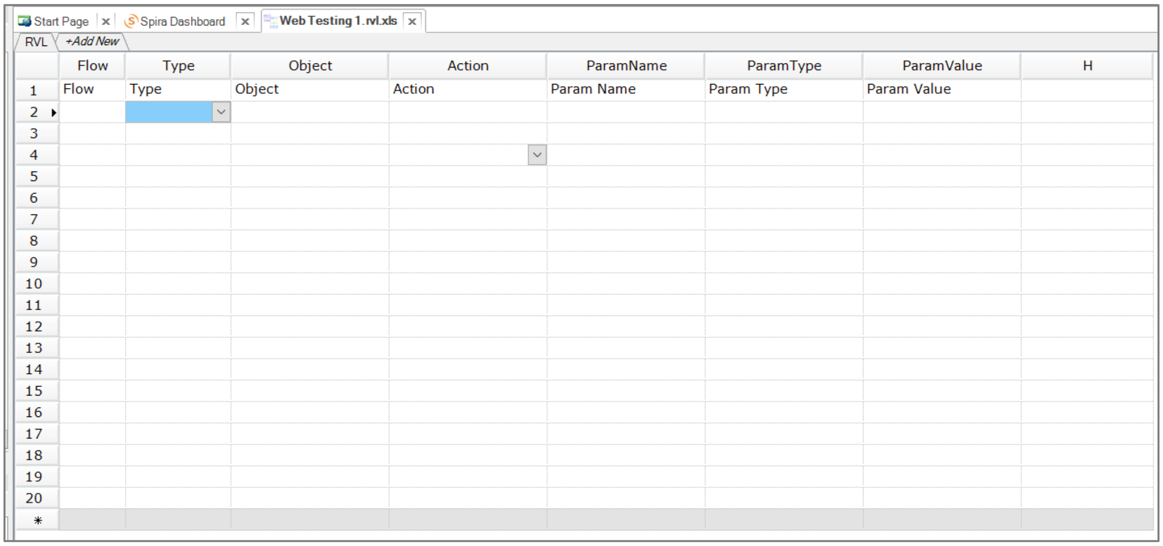
Choose preferred scripting language:

RVL (Rapise Visual Language)

JavaScript (Advanced Users)

□ Don't ask (may be changed later in Options\Settings\Default Scripting Language)

Rapise will create the new test and you will see the empty recording grid:

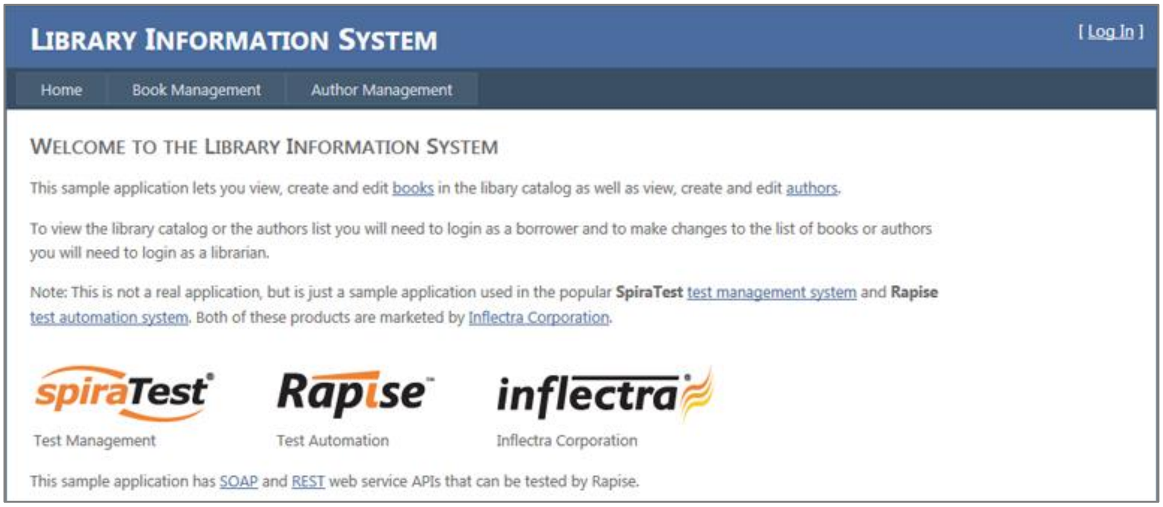


	Flow	Type	Object	Action	ParamName	ParamType	ParamValue	H
1	Flow	Type	Object	Action	Param Name	Param Type	Param Value	
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
*								

You are now ready to record your first test.

1.2. Opening the Application Under Test (AUT)

Open Internet Explorer. You will find it in **Start** > **All Programs** > **Internet Explorer**. In Internet Explorer, navigate to: http://www.libraryinformationsystem.org:



Welcome to the Library Information System

This sample application lets you view, create and edit books in the libary catalog as well as view, create and edit authors.

To view the library catalog or the authors list you will need to login as a borrower and to make changes to the list of books or authors you will need to login as a librarian.

Note: This is not a real application, but is just a sample application used in the popular **SpiraTest** test management system and **Rapise** test automation system. Both of these products are marketed by Inflectra Corporation.

Test Management

Test Automation

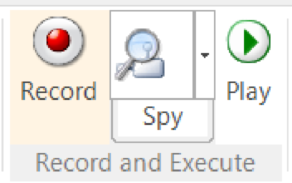
Inflectra Corporation

This sample application has SOAP and REST web service APIs that can be tested by Rapise.

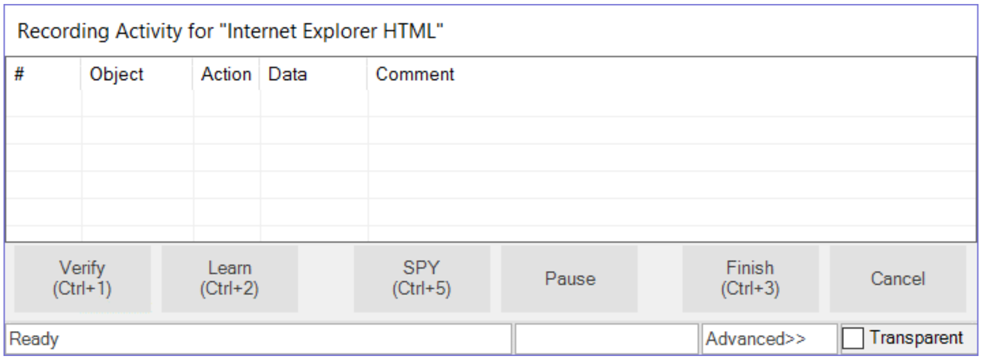
Make sure your browser zoom is set to exactly 100% for recording and playback.

1.3. Starting the Recording Session

In the Rapise window, press the **Record** button on the Ribbon.



The Recording Activity (RA) dialog box will appear:


Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment

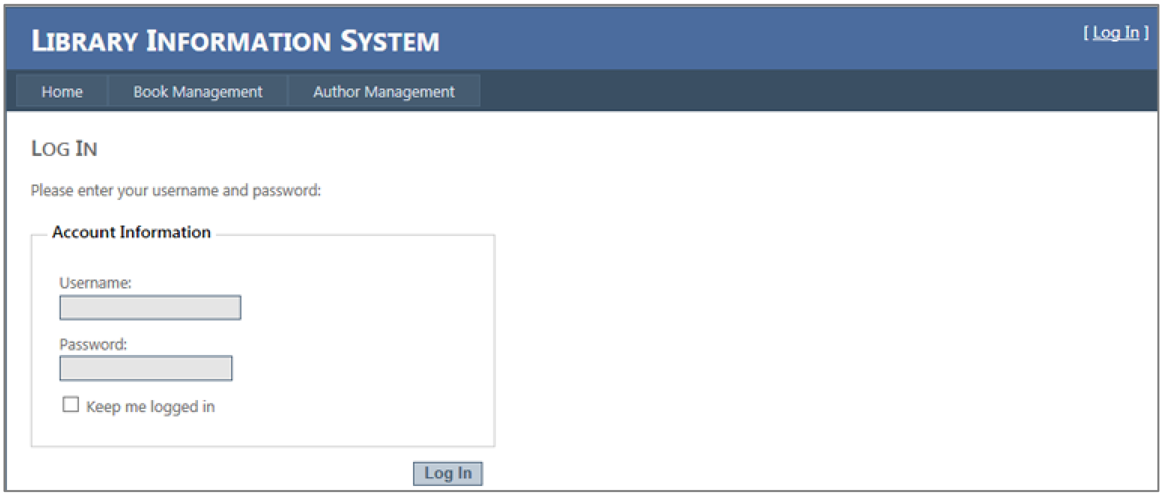
Ready

The **RA dialog** has a grid. As you interact with the Library Information System program, the grid will automatically populate with your actions.

> TIP: Please, don't hurry while doing recording. It is recommended to perform actions slowly, record in small portions and look at the recording activity window to see that the actions are captured as you expect them. You can record more actions later. Remember, that you will record it once and then play it back many times over. So, a good quality of recording will save you a lot of time in the future!

1.4. Recording the User Interactions

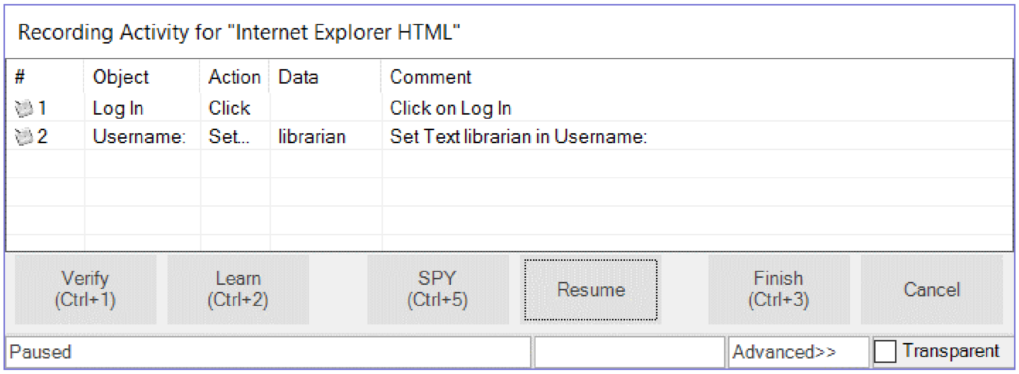
Let's begin creating the test. On the library information system login page, click on the **Log In** link in the top-right of the screen:



Please enter your username and password:

In the username text box, type *librarian*

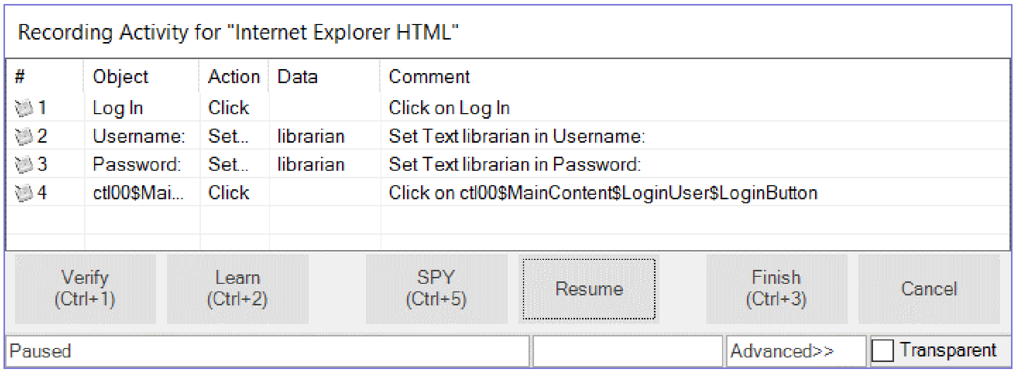
Click on the Password text box next. You'll notice that the **RA dialog** has changed. Your actions, clicking Log-In and entering a username, are listed in the grid:



Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Log In	Click		Click on Log In
2	Username:	Set...	librarian	Set Text librarian in Username:

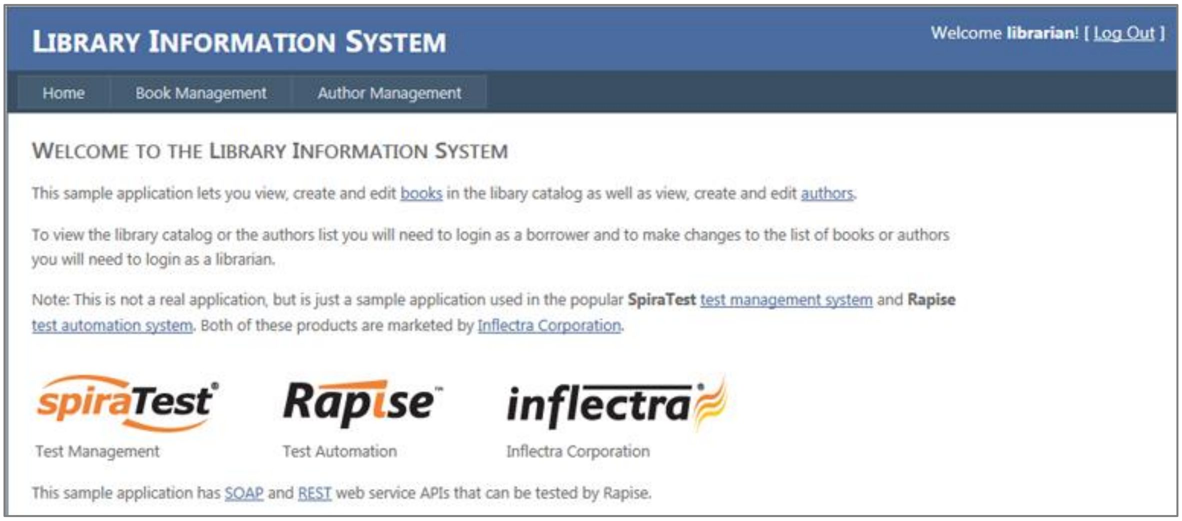
The password for user librarian is also *librarian*. Type the password in and then press the **Log-In** button. Two more rows should appear in the **RA dialog**: one to represent the password entry, and one to represent the button click:



Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Log In	Click		Click on Log In
2	Username:	Set...	librarian	Set Text librarian in Username:
3	Password:	Set...	librarian	Set Text librarian in Password:
4	ctl00$Mai...	Click		Click on ctl00$MainContent$LoginUser$LoginButton

You should now be on the main menu of the Library Information System with the user's name listed in the top-right:



LIBRARY INFORMATION SYSTEM

Welcome **librarian**! [Log Out]

WELCOME TO THE LIBRARY INFORMATION SYSTEM

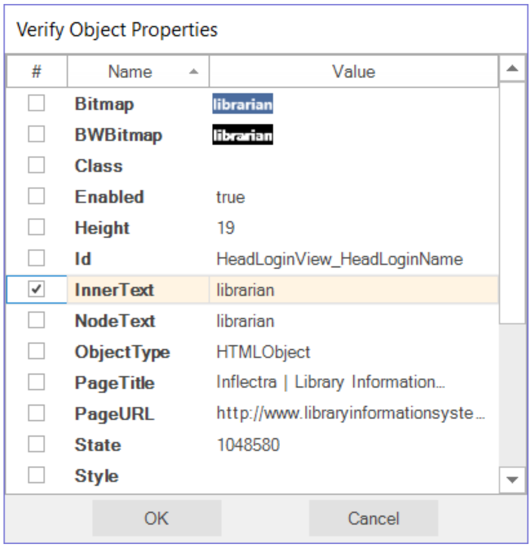
This sample application lets you view, create and edit books in the libary catalog as well as view, create and edit authors.

To view the library catalog or the authors list you will need to login as a borrower and to make changes to the list of books or authors you will need to login as a librarian.

Note: This is not a real application, but is just a sample application used in the popular **SpiraTest** test management system and **Rapise** test automation system. Both of these products are marketed by Inflectra Corporation.

This sample application has SOAP and REST web service APIs that can be tested by Rapise.

Hover the mouse over the “Welcome **librarian**” username label on the top-right and click CTRL+1 to bring up the Verify dialog box:

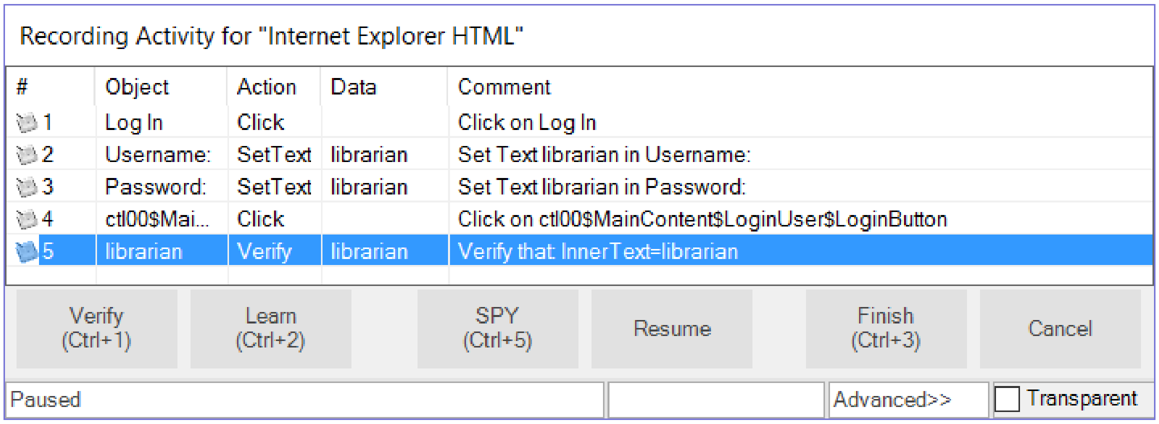


Verify Object Properties

#	Name	Value
☐	Bitmap	librarian
☐	BWBitmap	librarian
☐	Class	
☐	Enabled	true
☐	Height	19
☐	Id	HeadLoginView_HeadLoginName
☑	InnerText	librarian
☐	NodeText	librarian
☐	ObjectType	HTMLObject
☐	PageTitle	Inflectra | Library Information...
☐	PageURL	http://www.libraryinformationsyste...
☐	State	1048580
☐	Style	

This box lets you add a checkpoint to verify the properties of an object on the screen.

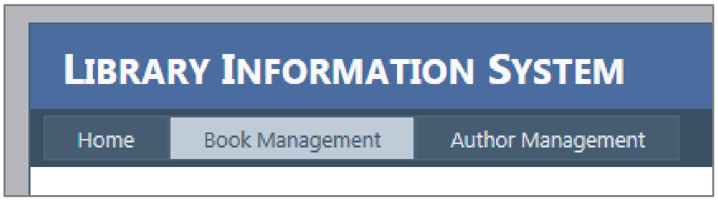
Select the “Inner Text” option and click X. That will add the verification check to your list of recorded actions:



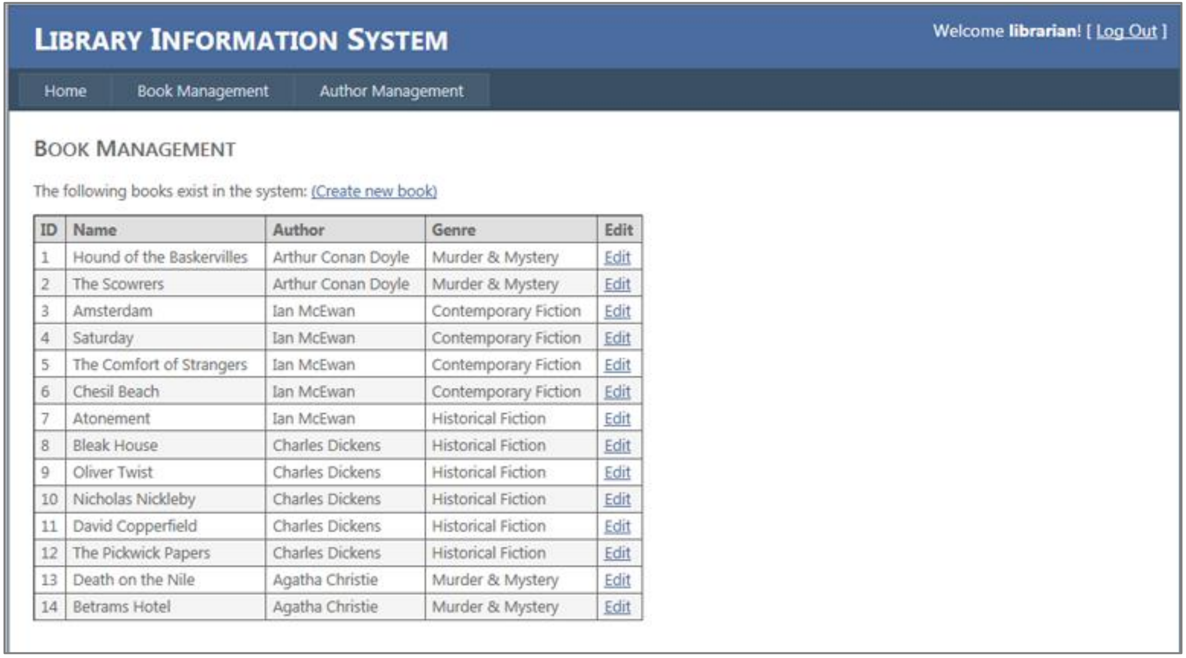
Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Log In	Click		Click on Log In
2	Username:	SetText	librarian	Set Text librarian in Username:
3	Password:	SetText	librarian	Set Text librarian in Password:
4	ctl00$Mai...	Click		Click on ctl00$MainContent$LoginUser$LoginButton
5	librarian	Verify	librarian	Verify that InnerText=librarian

Next, click the **Book Management** button. It is highlighted in the next screenshot:



You should now be on the Book Management page (see the below image).

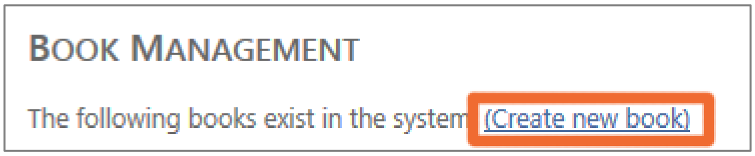


BOOK MANAGEMENT

The following books exist in the system: (Create new book)

ID	Name	Author	Genre	Edit
1	Hound of the Baskervilles	Arthur Conan Doyle	Murder & Mystery	Edit
2	The Scowrers	Arthur Conan Doyle	Murder & Mystery	Edit
3	Amsterdam	Ian McEwan	Contemporary Fiction	Edit
4	Saturday	Ian McEwan	Contemporary Fiction	Edit
5	The Comfort of Strangers	Ian McEwan	Contemporary Fiction	Edit
6	Chesil Beach	Ian McEwan	Contemporary Fiction	Edit
7	Atonement	Ian McEwan	Historical Fiction	Edit
8	Bleak House	Charles Dickens	Historical Fiction	Edit
9	Oliver Twist	Charles Dickens	Historical Fiction	Edit
10	Nicholas Nickleby	Charles Dickens	Historical Fiction	Edit
11	David Copperfield	Charles Dickens	Historical Fiction	Edit
12	The Pickwick Papers	Charles Dickens	Historical Fiction	Edit
13	Death on the Nile	Agatha Christie	Murder & Mystery	Edit
14	Betrams Hotel	Agatha Christie	Murder & Mystery	Edit

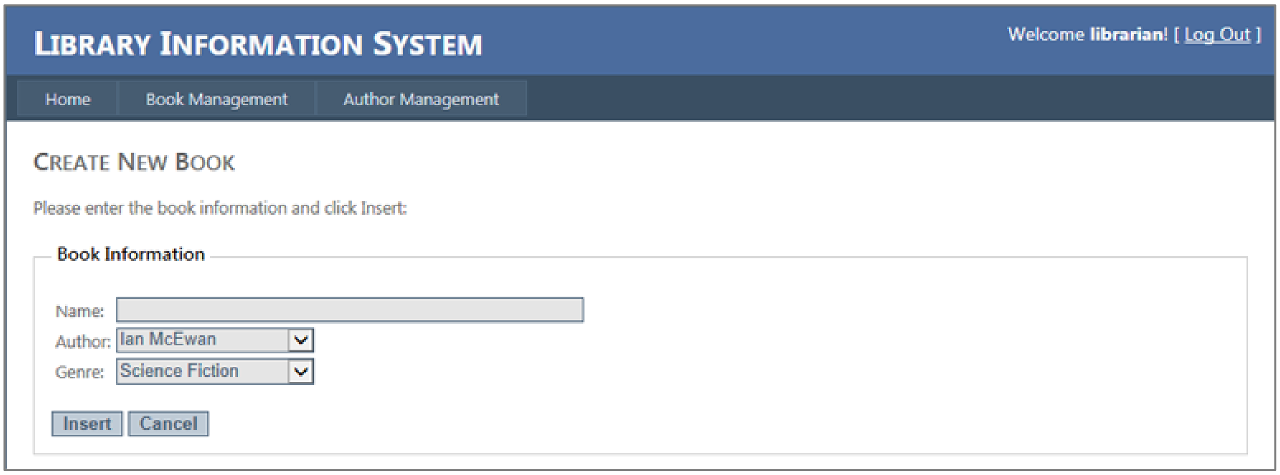
Click the **Create new book** link:



BOOK MANAGEMENT

The following books exist in the system: (Create new book)

You should now be on the Create New Book page (see image below). Click the **HOME** button to go back to the main menu.



CREATE NEW BOOK

Please enter the book information and click Insert:

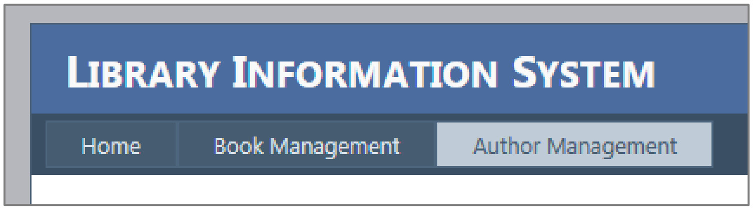
Book Information

Name:

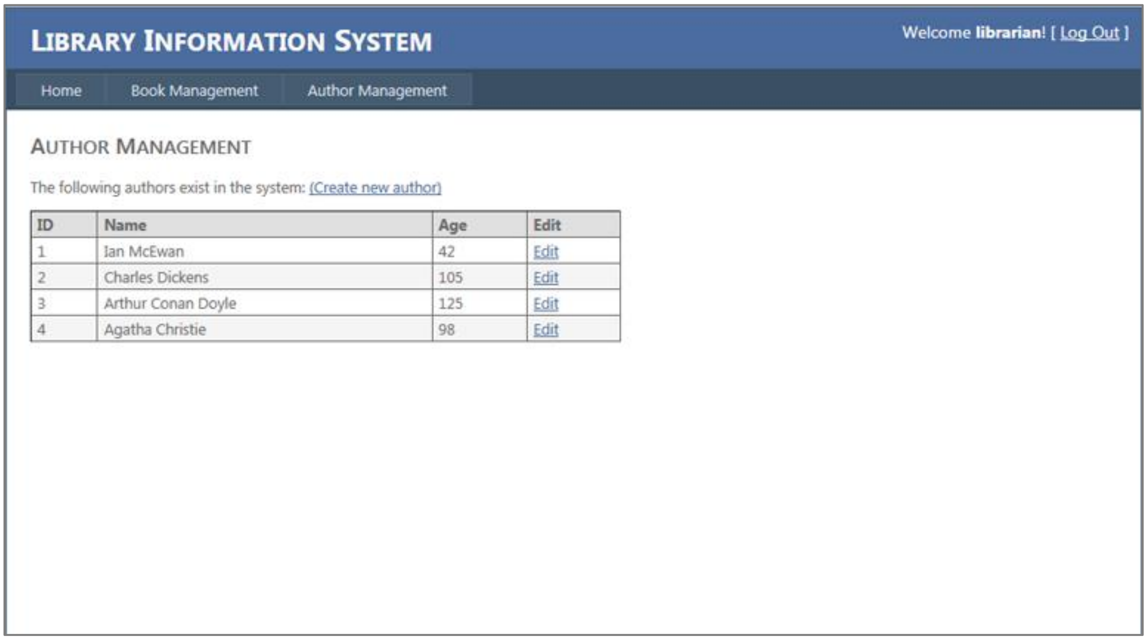
Author: Ian McEwan

Genre: Science Fiction

Now, click the **Author Management** button:



You should now be on the Author Management page (see image below):

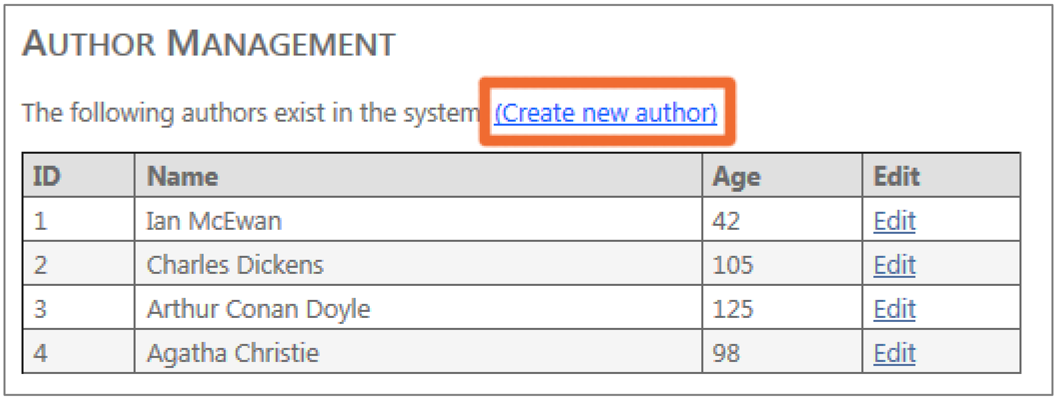


AUTHOR MANAGEMENT

The following authors exist in the system: (Create new author)

ID	Name	Age	Edit
1	Ian McEwan	42	Edit
2	Charles Dickens	105	Edit
3	Arthur Conan Doyle	125	Edit
4	Agatha Christie	98	Edit

Click the **Create New Author** link:

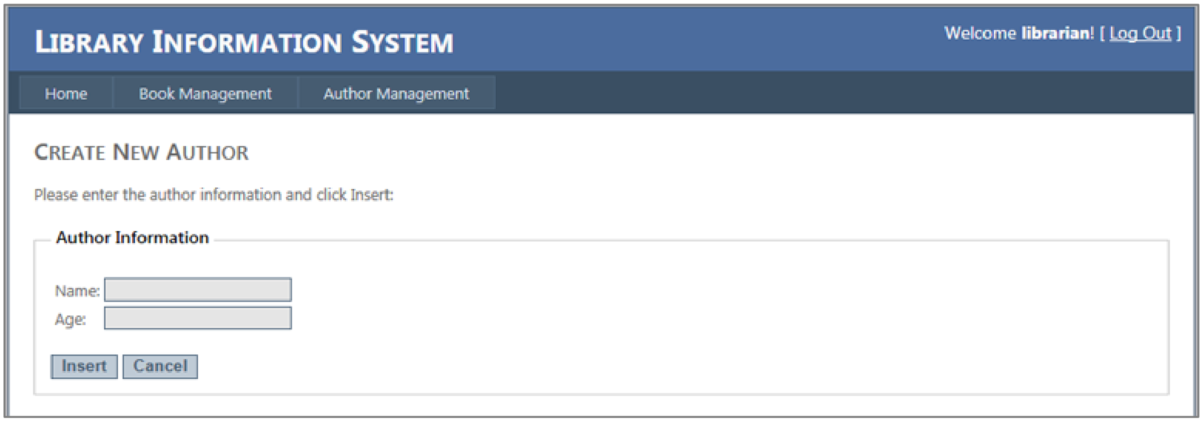


AUTHOR MANAGEMENT

The following authors exist in the system (Create new author)

ID	Name	Age	Edit
1	Ian McEwan	42	Edit
2	Charles Dickens	105	Edit
3	Arthur Conan Doyle	125	Edit
4	Agatha Christie	98	Edit

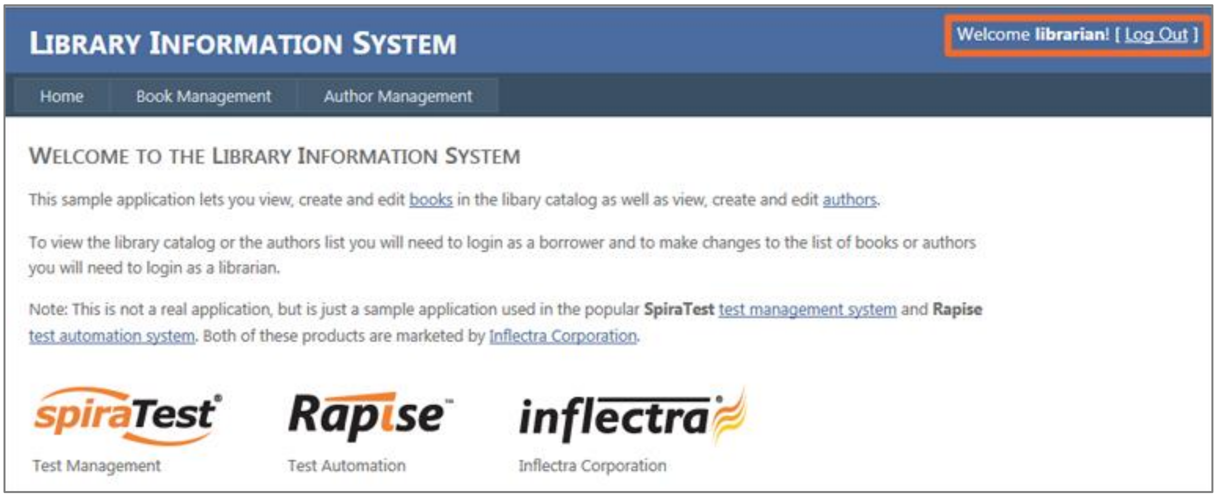
You should now be on the Create New Author page (see below). Click the **Home** button to go back to the main menu.



CREATE NEW AUTHOR

Please enter the author information and click Insert:

At this point, there should be approximately ten rows in the **RA dialog** grid.
You are now back on the Main Menu. Click **Log Out** (top-right).



WELCOME TO THE LIBRARY INFORMATION SYSTEM

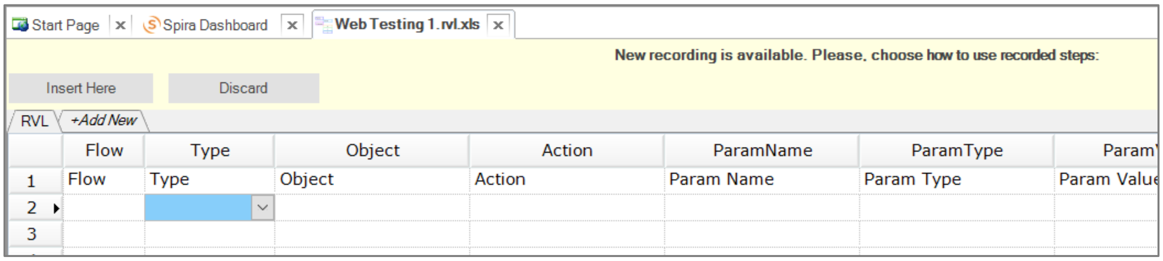
This sample application lets you view, create and edit books in the libary catalog as well as view, create and edit authors.

To view the library catalog or the authors list you will need to login as a borrower and to make changes to the list of books or authors you will need to login as a librarian.

Note: This is not a real application, but is just a sample application used in the popular **SpiraTest** test management system and **Rapise** test automation system. Both of these products are marketed by Inflectra Corporation.

To end the recording session, you can either press **CTRL+3** or press the **Finish** button on the Record dialog. End the recording session now.

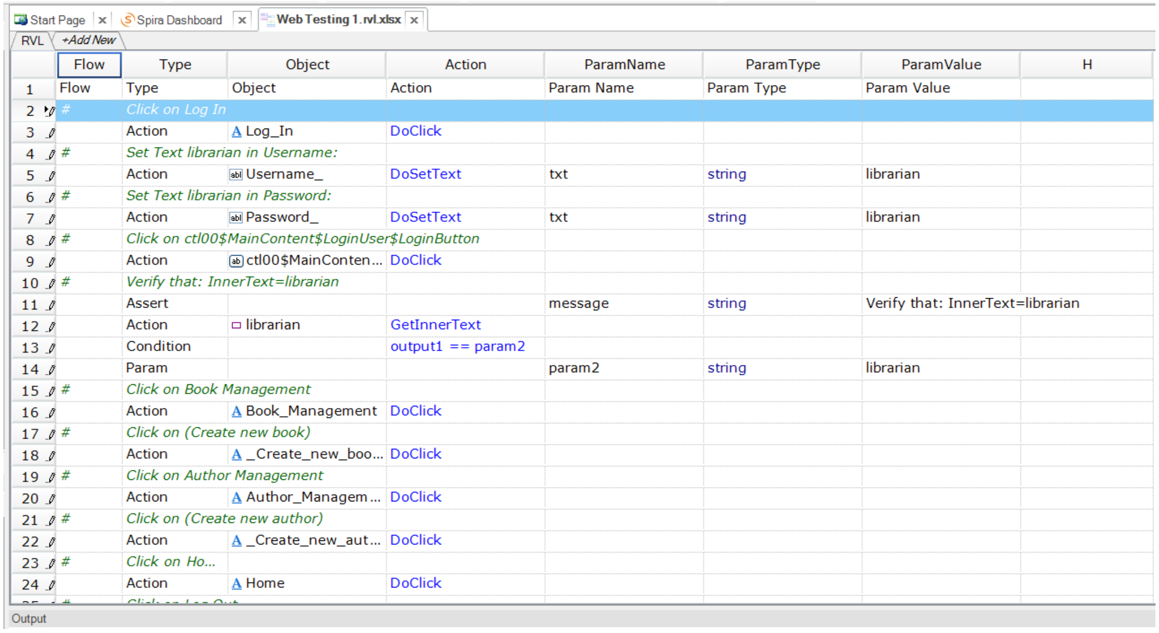
Rapise will ask you whether you want to use this recording or discard it:



New recording is available. Please, choose how to use recorded steps:

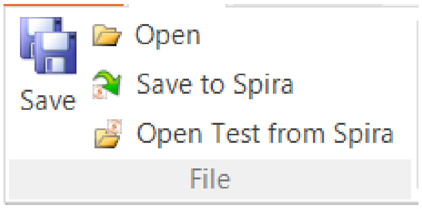
	Flow	Type	Object	Action	ParamName	ParamType	Param'
1	Flow	Type	Object	Action	Param Name	Param Type	Param Value
2							
3							

Click on the **Insert Here** button and Rapise will insert the recorded steps into the test grid:



	Flow	Type	Object	Action	ParamName	ParamType	ParamValue	H
1	Flow	Type	Object	Action	Param Name	Param Type	Param Value	
2	#	Click on Log In						
3		Action	Log_In	DoClick				
4	#	Set Text librarian in Username:						
5		Action	Username_	DoSetText	txt	string	librarian	
6	#	Set Text librarian in Password:						
7		Action	Password_	DoSetText	txt	string	librarian	
8	#	Click on ctl00$MainContent$LoginUser$LoginButton						
9		Action	ctl00$MainConten…	DoClick				
10	#	Verify that: InnerText=librarian						
11		Assert			message	string	Verify that: InnerText=librarian	
12		Action	librarian	GetInnerText				
13		Condition		output1 == param2				
14		Param			param2	string	librarian	
15	#	Click on Book Management						
16		Action	Book_Management	DoClick				
17	#	Click on (Create new book)						
18		Action	_Create_new_boo…	DoClick				
19	#	Click on Author Management						
20		Action	Author_Managem…	DoClick				
21	#	Click on (Create new author)						
22		Action	_Create_new_aut…	DoClick				
23	#	Click on Ho…						
24		Action	Home	DoClick				

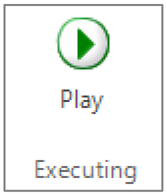
Let's save our test. Press the **Save** button at the top left of the Rapise window.



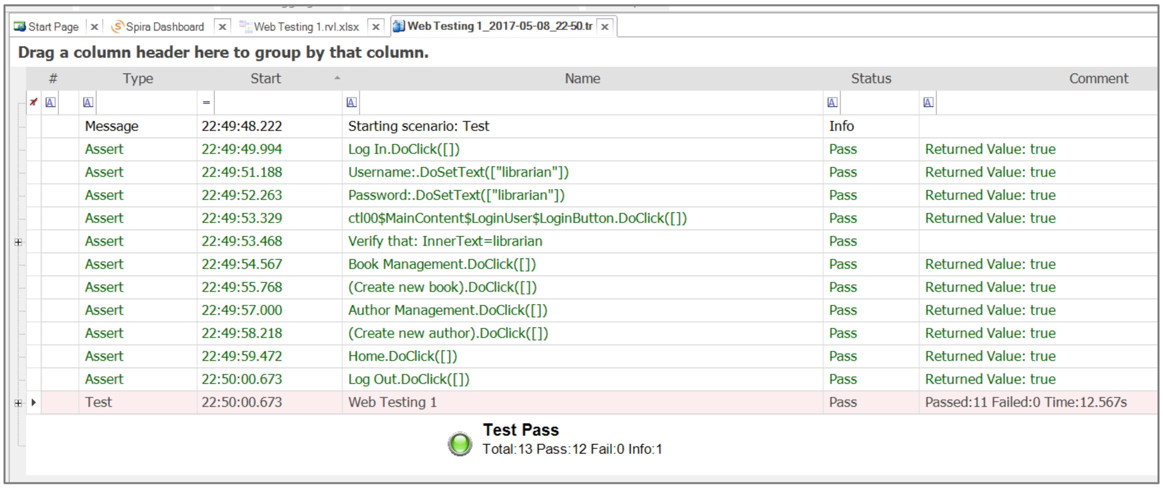
1.5. Playback of the Record Actions

Let’s execute the test we just created. First, close Internet explorer. Rapise will open a new instance of Internet Explorer to the correct URL (www.libraryinformationsystem.org) when the test begins.

To execute the script, press the Play button at the top middle of the Rapise window.



After execution, a screen like the one below will appear. Each row represents a step in the test. The rows with green text are steps which passed, whereas the rows with red text are the steps which failed.



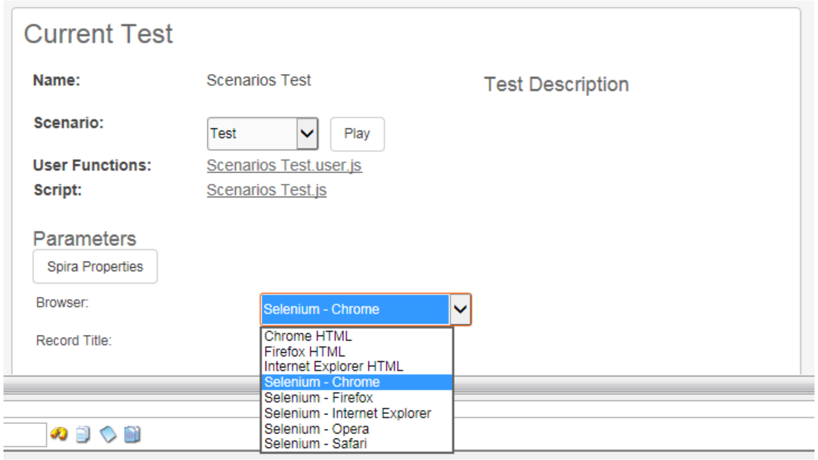
Drag a column header here to group by that column.

#	Type	Start	Name	Status	Comment
	Message	22:49:48.222	Starting scenario: Test	Info	
	Assert	22:49:49.994	Log In.DoClick([])	Pass	Returned Value: true
	Assert	22:49:51.188	Username:.DoSetText(["librarian"])	Pass	Returned Value: true
	Assert	22:49:52.263	Password:.DoSetText(["librarian"])	Pass	Returned Value: true
	Assert	22:49:53.329	ctl00$MainContent$LoginUser$LoginButton.DoClick([])	Pass	Returned Value: true
	Assert	22:49:53.468	Verify that: InnerText=librarian	Pass	
	Assert	22:49:54.567	Book Management.DoClick([])	Pass	Returned Value: true
	Assert	22:49:55.768	(Create new book).DoClick([])	Pass	Returned Value: true
	Assert	22:49:57.000	Author Management.DoClick([])	Pass	Returned Value: true
	Assert	22:49:58.218	(Create new author).DoClick([])	Pass	Returned Value: true
	Assert	22:49:59.472	Home.DoClick([])	Pass	Returned Value: true
	Assert	22:50:00.673	Log Out.DoClick([])	Pass	Returned Value: true
	Test	22:50:00.673	Web Testing 1	Pass	Passed:11 Failed:0 Time:12.567s

Test Pass
Total:13 Pass:12 Fail:0 Info:1

Congratulations… You have just recorded and played-back your first automated test script using Rapise.

Now that we have recorded our test in Internet Explorer, we want to play the same script back in other browsers. That is very easy to do. Simply click on the shortcut for the Rapise Start Page, and then change the **web browser dropdown** to a different browser (e.g. Firefox, Chrome, Selenium, etc.) and click the **Play** button.



Current Test

Name: Scenarios Test

Scenario: Test

User Functions: Scenarios Test.user.js

Script: Scenarios Test.js

Test Description

Parameters

Browser: Selenium - Chrome

Record Title:

Before the playback in other browsers will work correctly, you will need to make sure you have **configured the web browsers** appropriately. In the case of the Selenium options, you will need to make sure you have installed the Selenium WebDriver libraries correctly.

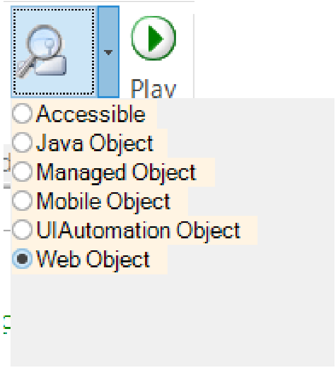
A copy of the instructions for setting up the web-browsers and Selenium is included in Appendix A and B of this guide respectively.

The next section will demonstrate how you can use Rapise to inspect the objects in a web page and Learn them for testing. This is useful in cases where you have more complex applications to test and you need to pick specific objects. For example, you may want to select one of the books in the grid based on its name rather than its row number (which may change if you add books).

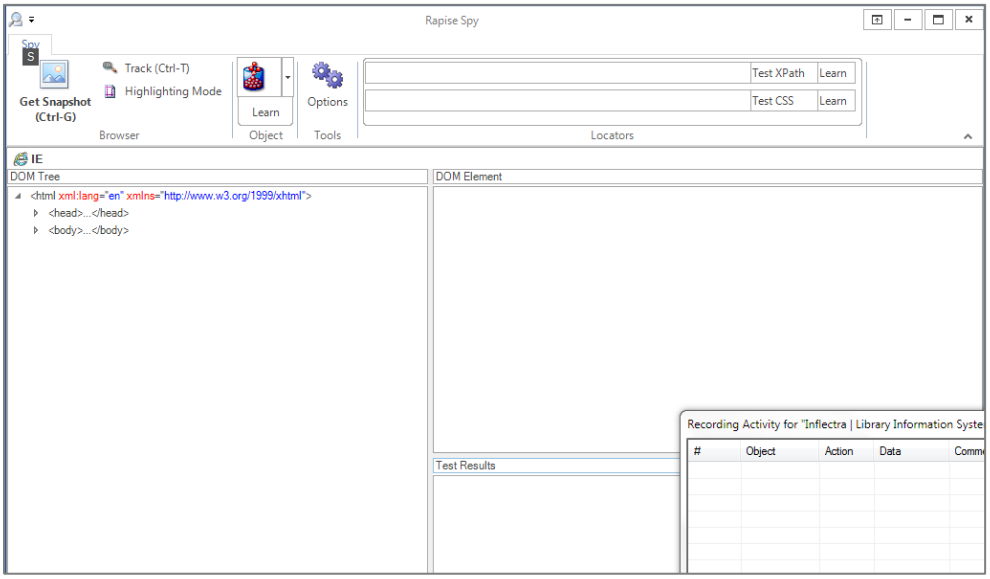
2. Using the Spy Tools in Rapise

2.1. Learning an Object Using the Web Spy

In the main Test ribbon of Rapise, expand the dropdown list for the ‘**Spy**’ tool and make sure that ‘Web Object’ is selected:

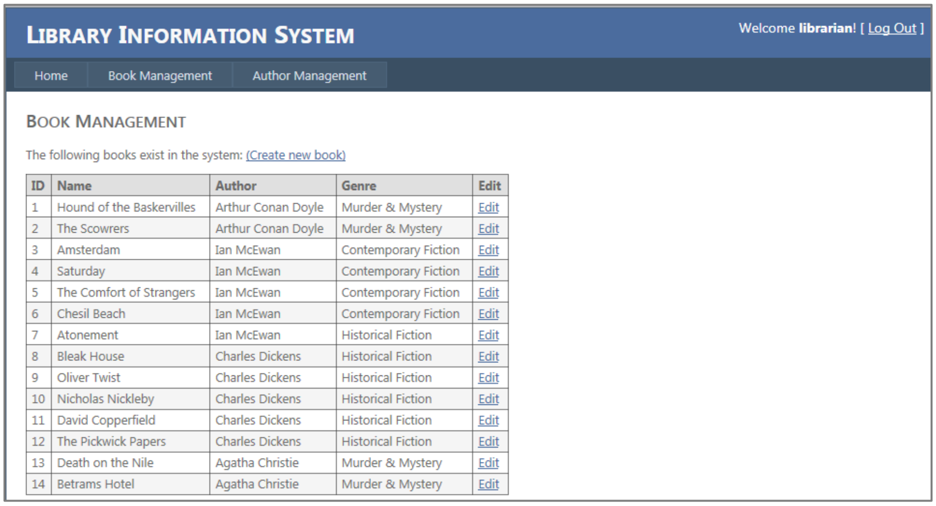


Now, click on the main ‘Spy’ tool icon and Rapise will start the Web Spy:



```
<html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
  <head>...</head>
  <body>...</body>
```

Go back to the web page and login to the library information system with the same login/password (librarian/librarian) and click on the ‘Book Management’ menu item so that the list of books is displayed:



LIBRARY INFORMATION SYSTEM

Welcome **librarian**! [Log Out]

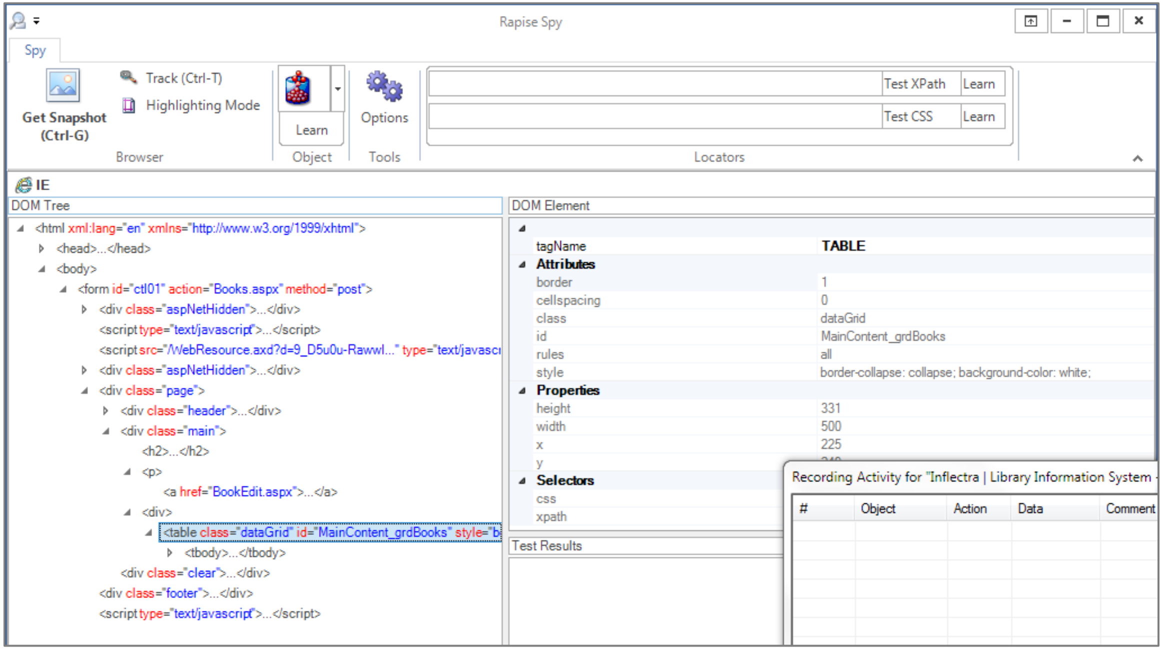
Home | Book Management | Author Management

BOOK MANAGEMENT

The following books exist in the system: (Create new book)

ID	Name	Author	Genre	Edit
1	Hound of the Baskervilles	Arthur Conan Doyle	Murder & Mystery	Edit
2	The Scowrers	Arthur Conan Doyle	Murder & Mystery	Edit
3	Amsterdam	Ian McEwan	Contemporary Fiction	Edit
4	Saturday	Ian McEwan	Contemporary Fiction	Edit
5	The Comfort of Strangers	Ian McEwan	Contemporary Fiction	Edit
6	Chesil Beach	Ian McEwan	Contemporary Fiction	Edit
7	Atonement	Ian McEwan	Historical Fiction	Edit
8	Bleak House	Charles Dickens	Historical Fiction	Edit
9	Oliver Twist	Charles Dickens	Historical Fiction	Edit
10	Nicholas Nickleby	Charles Dickens	Historical Fiction	Edit
11	David Copperfield	Charles Dickens	Historical Fiction	Edit
12	The Pickwick Papers	Charles Dickens	Historical Fiction	Edit
13	Death on the Nile	Agatha Christie	Murder & Mystery	Edit
14	Betrams Hotel	Agatha Christie	Murder & Mystery	Edit

Now back in the Web Spy, click on the 'Get Snapshot' option to refresh the Web Spy and display the HTML elements (called the DOM tree) that make up this page:



```
<html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
  <head>...</head>
  <body>
    <form id="ctl01" action="Books.aspx" method="post">
      <div class="aspNetHidden">...</div>
      <script type="text/javascript">...</script>
      <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." type="text/javasci
      <div class="aspNetHidden">...</div>
      <div class="page">
        <div class="header">...</div>
        <div class="main">
          <h2>...</h2>
          <p>
            <a href="BookEdit.aspx">...</a>
          <div>
            <table class="dataGrid" id="MainContent_grdBooks" style="b
              <tbody>...</tbody>
        <div class="clear">...</div>
      <div class="footer">...</div>
      <script type="text/javascript">...</script>
```

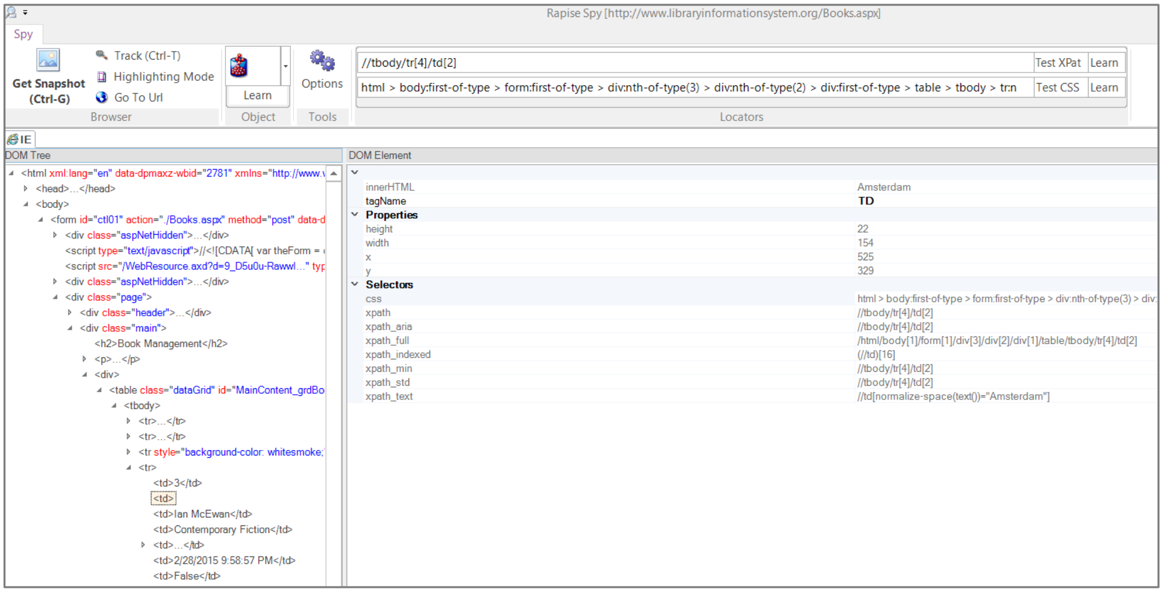
tagName	**TABLE**
Attributes	
border	1
cellspacing	0
class	dataGrid
id	MainContent_grdBooks
rules	all
style	border-collapse: collapse; background-color: white;
Properties	
height	331
width	500
x	225
y	
Selectors	
css	
xpath	

Recording Activity for "Inflectra | Library Information System

#	Object	Action	Data	Comment

Once it has loaded the DOM tree, you can expand/collapse the elements to see how the web page is constructed. This is useful when testing an application since many of the HTML elements on a page may be used for layout purposes and will not be visible in the browser. In the example page, we have expanded some of the nodes to display the main section of the page and the table that contains the list of books.

In addition, you can use the **Track (Ctrl+T)** tool to select and item in the web page and then have it be highlighted in the DOM tree. For example if we want to find the cell that contains the book title "Amsterdam", simply click CTRL+T on the keyboard, move the mouse over the cell in the webpage**, wait until the red highlighting rectangle appears** and then click CTRL+T again. Rapise will now highlight that item in the DOM Tree automatically:



Rapise Spy [http://www.libraryinformationsystem.org/Books.aspx]

//tbody/tr[4]/td[2]

html > body:first-of-type > form:first-of-type > div:nth-of-type(3) > div:nth-of-type(2) > div:first-of-type > table > tbody > tr:n

DOM Tree

```
<html xml:lang="en" data-dpmaxz-wbid="2781" xmlns="http://www.\
  <head>...</head>
  <body>
    <form id="ctl01" action="./Books.aspx" method="post" data-d
      <div class="aspNetHidden">...</div>
      <script type="text/javascript">//<![CDATA[ var theForm = (
      <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." typ
      <div class="aspNetHidden">...</div>
      <div class="page">
        <div class="header">...</div>
        <div class="main">
          <h2>Book Management</h2>
          <p>...</p>
          <div>
            <table class="dataGrid" id="MainContent_grdBo
              <tbody>
                <tr>...</tr>
                <tr>...</tr>
                <tr style="background-color: whitesmoke;
                <tr>
                  <td>3</td>
                  <td>
                  <td>Ian McEwan</td>
                  <td>Contemporary Fiction</td>
                  <td>...</td>
                  <td>2/28/2015 9:58:57 PM</td>
                  <td>False</td>
```

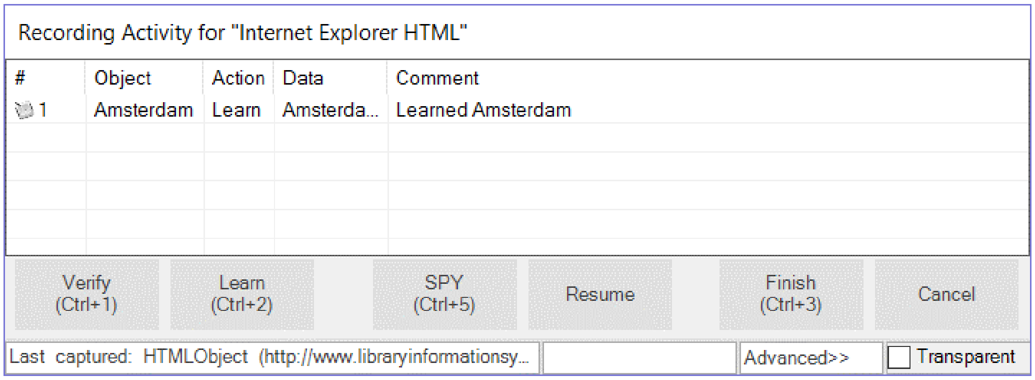
DOM Element

innerHTML	Amsterdam
tagName	**TD**
Properties	
height	22
width	154
x	525
y	329
Selectors	
css	html > body:first-of-type > form:first-of-type > div:nth-of-type(3) > div:
xpath	//tbody/tr[4]/td[2]
xpath_aria	//tbody/tr[4]/td[2]
xpath_full	/html/body[1]/form[1]/div[3]/div[2]/div[1]/table/tbody/tr[4]/td[2]
xpath_indexed	(//td)[16]
xpath_min	//tbody/tr[4]/td[2]
xpath_std	//tbody/tr[4]/td[2]
xpath_text	//td[normalize-space(text())="Amsterdam"]

You can see all of the properties of this HTML element displayed on the right, specifically:

- The tagName is displayed as 'TD' (always upper case)
- The innerHTML of the element is displayed (Amsterdam)
- The CSS and XPath for locating this element is displayed
    - Rapise will list various different XPATH / CSS options that you can choose from.

If you want to use this object in a Rapise test script, you can simply click the **Learn** button and the HTML element will be added to the Recording Activity Dialog:

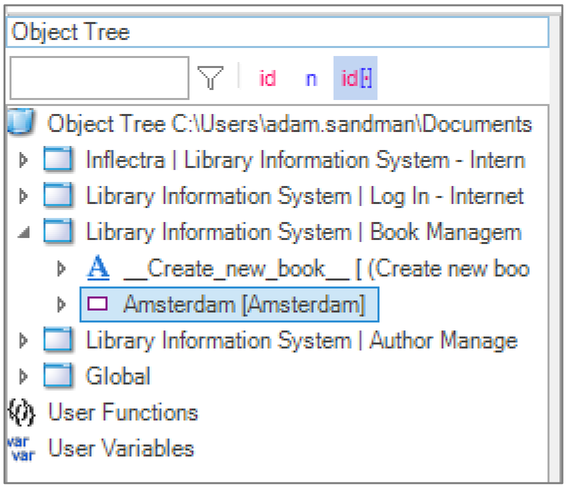


Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Amsterdam	Learn	Amsterda...	Learned Amsterdam

Last captured: HTMLObject (http://www.libraryinformationsy...

Click **Finish** and the object will have been added to your test's Object Tree:

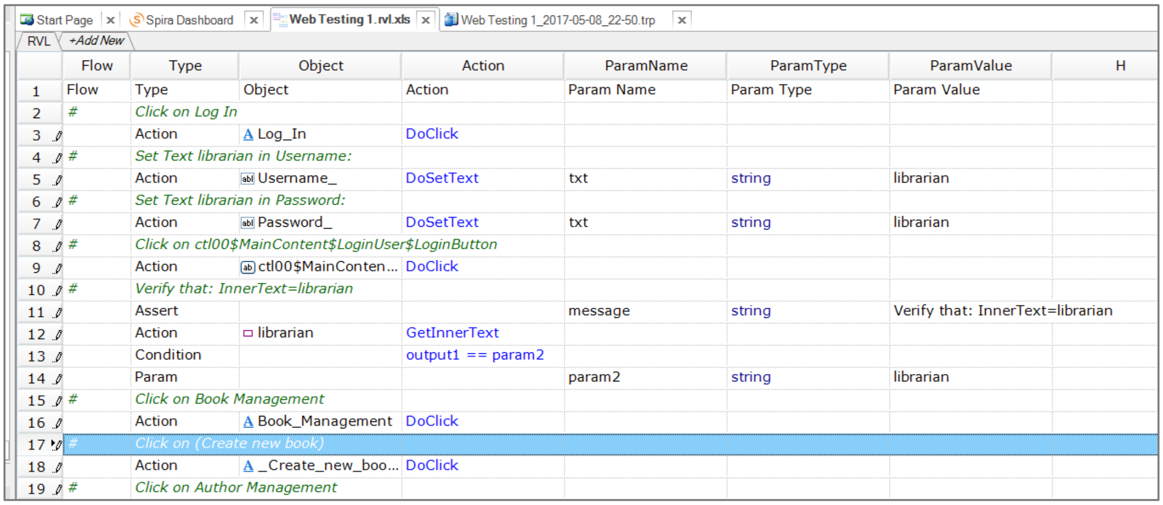


Object Tree

id n id[]

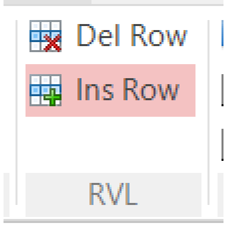
- Object Tree C:\Users\adam.sandman\Documents
  - Inflectra | Library Information System - Intern
  - Library Information System | Log In - Internet
  - Library Information System | Book Managem
    - __Create_new_book__ [(Create new boo
    - Amsterdam [Amsterdam]
  - Library Information System | Author Manage
  - Global
- User Functions
- User Variables

To use this new object in our test script, we can simply use the test editor to add the appropriate command. For example, if you wanted to get the textual value of the cell in your test, you should click on the row in the grid after the **Book Management – DoClick**:

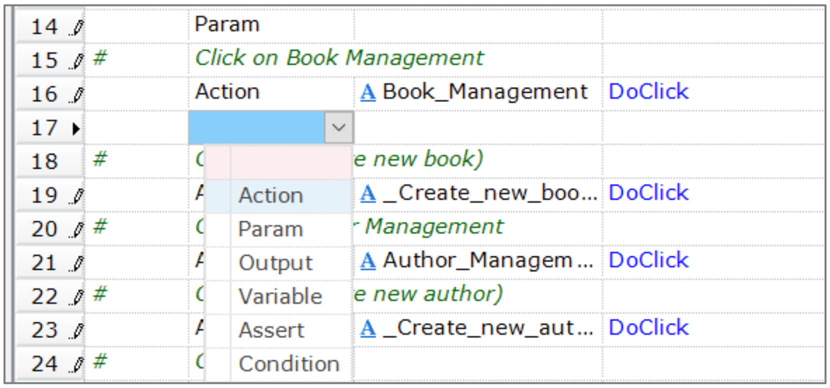


	Flow	Type	Object	Action	ParamName	ParamType	ParamValue	H
1	Flow	Type	Object	Action	Param Name	Param Type	Param Value	
2	#	Click on Log In						
3		Action	Log_In	DoClick				
4	#	Set Text librarian in Username:						
5		Action	Username_	DoSetText	txt	string	librarian	
6	#	Set Text librarian in Password:						
7		Action	Password_	DoSetText	txt	string	librarian	
8	#	Click on ctl00$MainContent$LoginUser$LoginButton						
9		Action	ctl00$MainConten...	DoClick				
10	#	Verify that: InnerText=librarian						
11		Assert			message	string	Verify that: InnerText=librarian	
12		Action	librarian	GetInnerText				
13		Condition		output1 == param2				
14		Param			param2	string	librarian	
15	#	Click on Book Management						
16		Action	Book_Management	DoClick				
17	#	Click on (Create new book)						
18		Action	_Create_new_boo...	DoClick				
19	#	Click on Author Management						

Now click on the **Ins Row** button in the main Test ribbon to add a new row:



This will insert a new row into the test. In this new row, right-click on each of the cells (as illustrated below) and pick the following values from the dropdown lists:

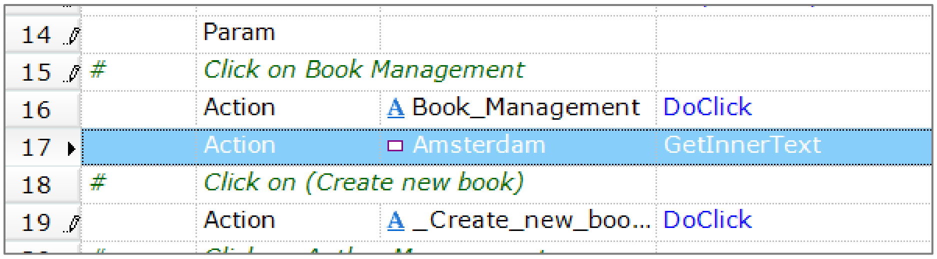


14		Param		
15	#	Click on Book Management		
16		Action	A Book_Management	DoClick
17				
18	#	(	e new book)	
19		A	A _Create_new_boo...	DoClick
20	#	(	Management	
21		A	A Author_Managem...	DoClick
22	#	(	e new author)	
23		A	A _Create_new_aut...	DoClick
24	#	(		

Then choose the following:

- **Type** = Action
- **Object** = Amsterdam
- **Action** = GetInnerText

You should now have the following:



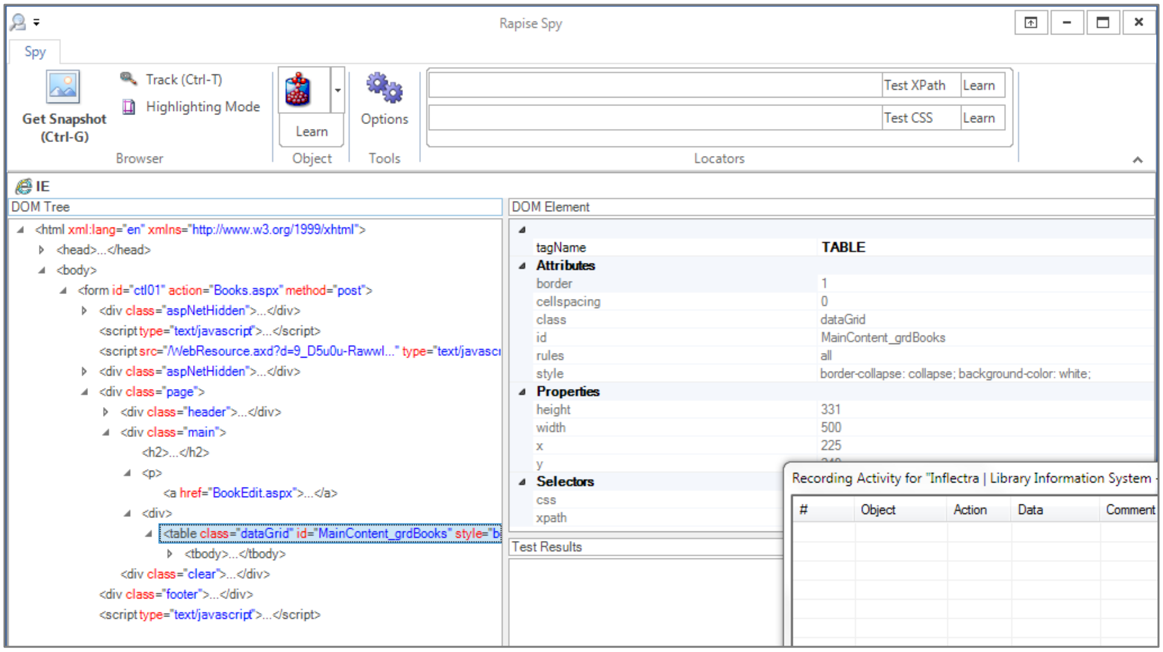
14		Param		
15	#	Click on Book Management		
16		Action	A Book_Management	DoClick
17		Action	Amsterdam	GetInnerText
18	#	Click on (Create new book)		
19		Action	A _Create_new_boo...	DoClick

2.2. Learning an Object from XPATH

In addition to letting Rapise automatically learn the object from the Web Spy, you can manually enter in XPATH or CSS queries to find matching elements on the page and then learn those for use in your test.

For example, suppose we want to dynamically find the row that has the cell containing Amsterdam and then click on its Edit hyperlink.

Open up the Web Spy as before:



```
<html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
  <head>...</head>
  <body>
    <form id="ctl01" action="Books.aspx" method="post">
      <div class="aspNetHidden">...</div>
      <script type="text/javascript">...</script>
      <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." type="text/javasci
      <div class="aspNetHidden">...</div>
      <div class="page">
        <div class="header">...</div>
        <div class="main">
          <h2>...</h2>
          <p>
            <a href="BookEdit.aspx">...</a>
          <div>
            <table class="dataGrid" id="MainContent_grdBooks" style="b
              <tbody>...</tbody>
        <div class="clear">...</div>
      <div class="footer">...</div>
      <script type="text/javascript">...</script>
```

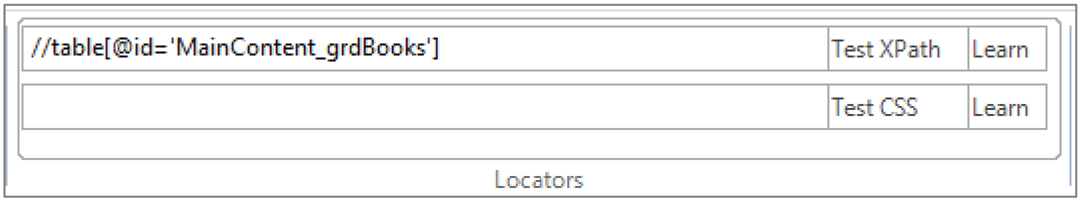
tagName	TABLE
Attributes	
border	1
cellspacing	0
class	dataGrid
id	MainContent_grdBooks
rules	all
style	border-collapse: collapse; background-color: white;
Properties	
height	331
width	500
x	225
y	
Selectors	
css	
xpath	

#	Object	Action	Data	Comment

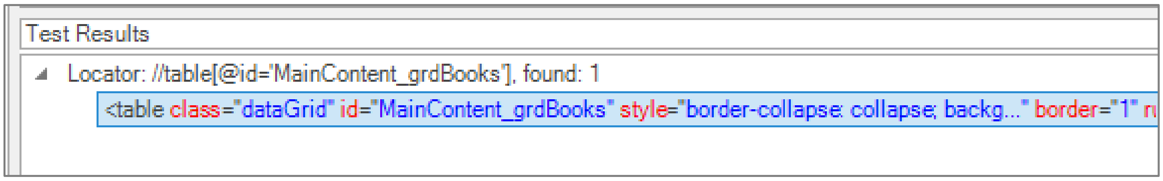
In the **Locators** section of the Web Spy, enter in the following to locate the table:

```
//table[@id='MainContent_grdBooks']
```

Now click on the **Test XPath** button to display the matching results:



//table[@id='MainContent_grdBooks']



Locator: //table[@id='MainContent_grdBooks'], found: 1

<table class="dataGrid" id="MainContent_grdBooks" style="border-collapse: collapse; backg..." border="1" ru

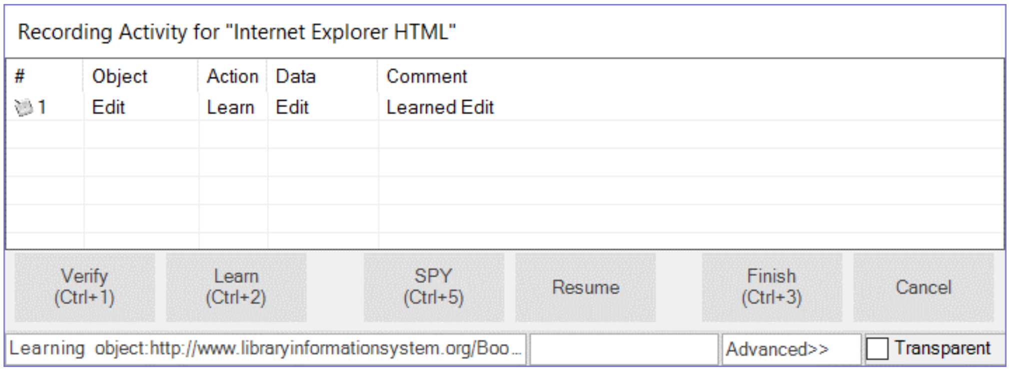
Now that we have matched the table, we need to add dynamic XPath to find any row that has the cell containing ‘Amsterdam’ and find the edit link. You can expand the table and see the rows and cells visually and that will help us create the XPATH:

```
//table[@id='MainContent_grdBooks']//tr/td[text()='Amsterdam']/../td[5]/a
```

This XPath consists of the following elements:

1. Finds the table with the specified ID
2. Find any row inside that table that contains a cell with the text ‘Amsterdam’
3. For any matching cell, get its parent row and inside the fifth cell, get any hyperlink

In this case that will correctly locate the Edit link for the book ‘Amsterdam’. Now that we have the correct item identified, click on the ‘**Learn’** button to the right of the ‘**Test XPath**’ button. That will now learn a new object that corresponds to the ‘Edit Amsterdam’ object:

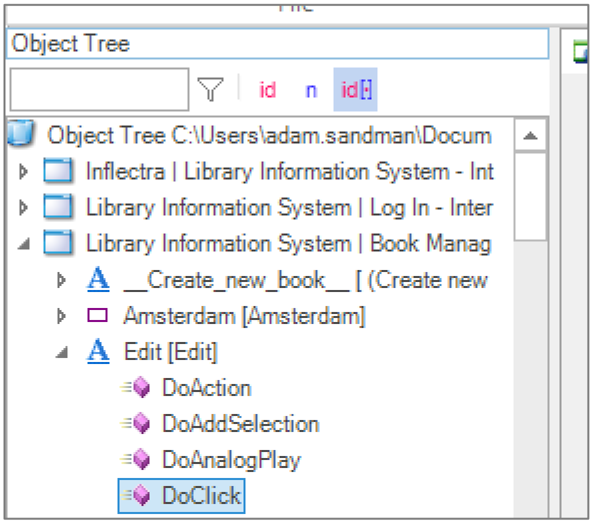


Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Edit	Learn	Edit	Learned Edit

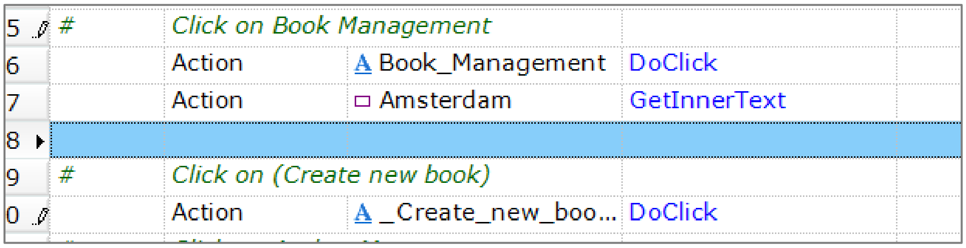
Learning object:http://www.libraryinformationsystem.org/Boo...

Click Finish and the object will have been added to the Object Tree of the current test:



Object Tree
id
n
id[:]
Object Tree C:\Users\adam.sandman\Docum
Inflectra | Library Information System - Int
Library Information System | Log In - Inter
Library Information System | Book Manag
__Create_new_book__ [(Create new
Amsterdam [Amsterdam]
Edit [Edit]
DoAction
DoAddSelection
DoAnalogPlay
DoClick

To click on this object, simply click on the row in the grid where you want this action to occur and choose “**Ins Row**” from the test ribbon:

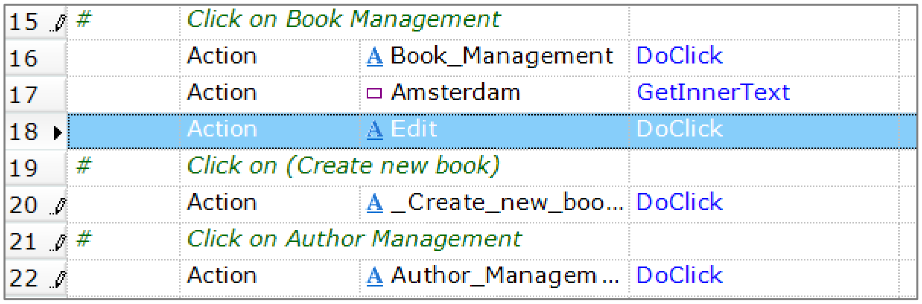


5	#	Click on Book Management		
6		Action	Book_Management	DoClick
7		Action	Amsterdam	GetInnerText
8				
9	#	Click on (Create new book)		
0		Action	_Create_new_boo...	DoClick

In this new row, right-click on each of the cells, and pick the following values from the dropdown lists:

- **Type** = Action
- **Object** = Edit
- **Action** = DoClick

So your test will now look like:



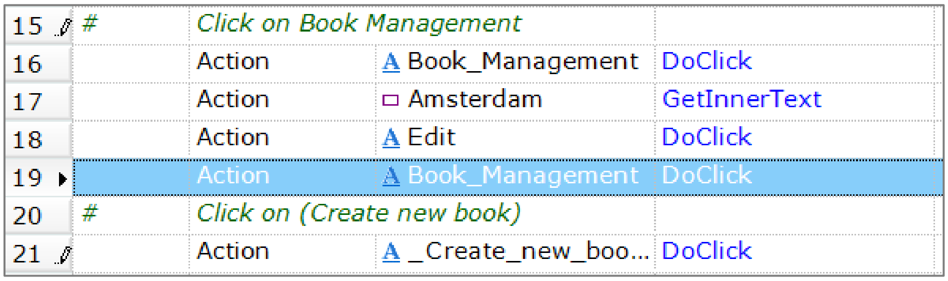
15	#	Click on Book Management		
16		Action	A Book_Management	DoClick
17		Action	□ Amsterdam	GetInnerText
18		Action	A Edit	DoClick
19	#	Click on (Create new book)		
20		Action	A _Create_new_boo...	DoClick
21	#	Click on Author Management		
22		Action	A Author_Managem...	DoClick

Since clicking on the Edit link will take you to a different page than where the ‘Create New Book’ link is available, we need to add another row and add:

- **Type** = Action
- **Object** = Book_Management
- **Action** = DoClick

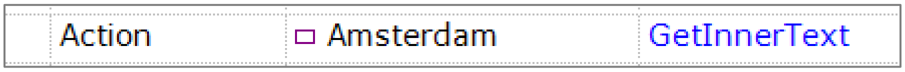
so that Rapise goes back to the main book list page before executing the Create New Book step. If we did not do this, the test would have failed.

This means the test will now look like:



15	#	Click on Book Management		
16		Action	A Book_Management	DoClick
17		Action	□ Amsterdam	GetInnerText
18		Action	A Edit	DoClick
19		Action	A Book_Management	DoClick
20	#	Click on (Create new book)		
21		Action	A _Create_new_boo...	DoClick

Now the line:

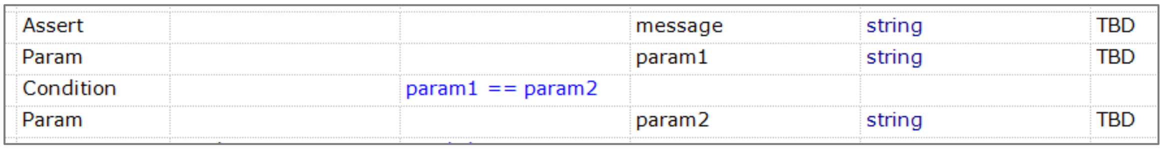


Action	□ Amsterdam	GetInnerText

by itself does not do anything, it just gets the text.

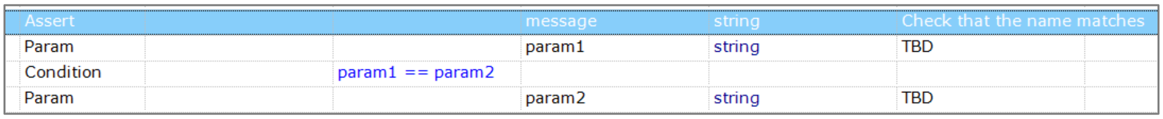
So, to make the test more useful, we can use the **Assert** command to turn this into a test for the correct value. To make the change, simply delete this entire row using the **Del Row** option on the ribbon and add a new blank row in its place.

Now choose the Type as **Assert** and press **ENTER** on the keyboard. You will now have the following:



Assert			message	string	TBD
Param			param1	string	TBD
Condition		param1 == param2			
Param			param2	string	TBD

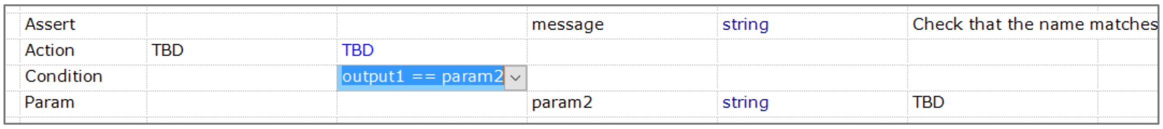
This is a generic placeholder for any type of conditional test. We need to first change the value of the message “TBD” to something meaningful (e.g. “Check that the name matches”):



Assert			message	string	Check that the name matches
Param			param1	string	TBD
Condition		param1 == param2			
Param			param2	string	TBD

This is the message that will be displayed if the test passes correctly.

Now we need to replace the first parameter with the output from the result of the Amsterdam test. To do this, change the **Condition** on the second row from: **param1 == param2** to **output1 == param2**:

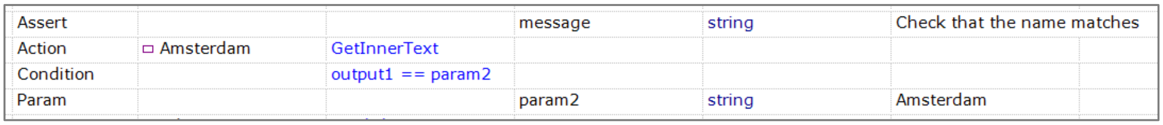


Assert			message	string	Check that the name matches
Action	TBD	TBD			
Condition		output1 == param2			
Param			param2	string	TBD

Notice how Rapise automatically changed the second row from **Param** to **Action**.

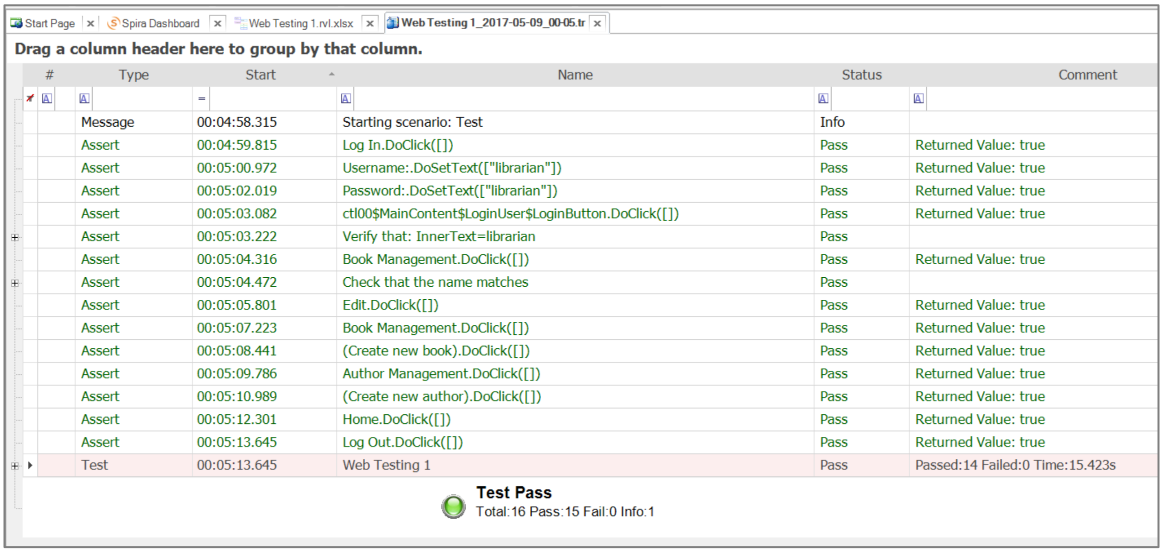
Now in the TBD cell next to the Action type, choose the **Amsterdam** object name and the **GetInnerText** action from the dropdown lists. Finally change the **param2** string from TBD to the expected value "Amsterdam".

You will now have:



Assert			message	string	Check that the name matches
Action	□ Amsterdam	GetInnerText			
Condition		output1 == param2			
Param			param2	string	Amsterdam

Now click **Play** to playback the new test:



Drag a column header here to group by that column.

#	Type	Start	Name	Status	Comment
	Message	00:04:58.315	Starting scenario: Test	Info	
	Assert	00:04:59.815	Log In.DoClick([])	Pass	Returned Value: true
	Assert	00:05:00.972	Username:.DoSetText(["librarian"])	Pass	Returned Value: true
	Assert	00:05:02.019	Password:.DoSetText(["librarian"])	Pass	Returned Value: true
	Assert	00:05:03.082	ctl00$MainContent$LoginUser$LoginButton.DoClick([])	Pass	Returned Value: true
	Assert	00:05:03.222	Verify that: InnerText=librarian	Pass	
	Assert	00:05:04.316	Book Management.DoClick([])	Pass	Returned Value: true
	Assert	00:05:04.472	Check that the name matches	Pass	
	Assert	00:05:05.801	Edit.DoClick([])	Pass	Returned Value: true
	Assert	00:05:07.223	Book Management.DoClick([])	Pass	Returned Value: true
	Assert	00:05:08.441	(Create new book).DoClick([])	Pass	Returned Value: true
	Assert	00:05:09.786	Author Management.DoClick([])	Pass	Returned Value: true
	Assert	00:05:10.989	(Create new author).DoClick([])	Pass	Returned Value: true
	Assert	00:05:12.301	Home.DoClick([])	Pass	Returned Value: true
	Assert	00:05:13.645	Log Out.DoClick([])	Pass	Returned Value: true
	Test	00:05:13.645	Web Testing 1	Pass	Passed:14 Failed:0 Time:15.423s

Test Pass
Total:16 Pass:15 Fail:0 Info:1

The test should now pass successfully.

3. Next Steps

Now we recommend that you take a look at the *Rapise User Manual*. This contains several more in-depth tutorials for using the system as well as an in-depth reference guide of all the different features available.

Appendix A – Configuring Web Browsers

Before you can use Rapise with certain web browsers to do web testing, you will need to install plugins so that Rapise can communicate with them.

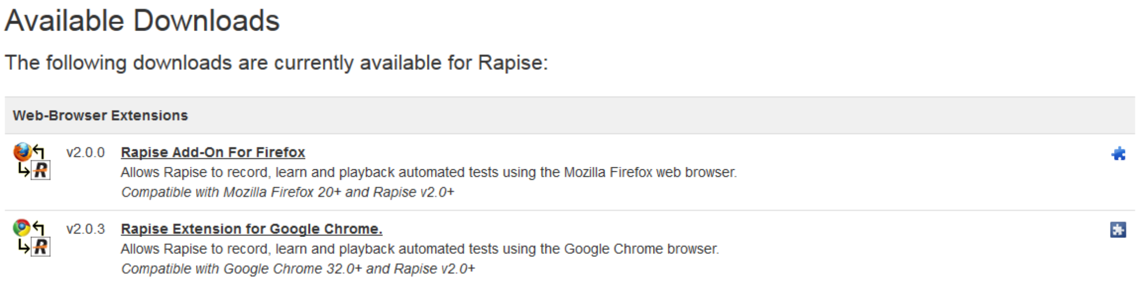
A.1. Preparing Internet Explorer (IE)

Unlike the other web browsers, there are actually no steps needed to configure IE. Once Rapise is installed, it is ready to connect to IE for recording, playback and learning without any configuration steps.

A.2. Preparing Firefox

In order to test web applications using the Firefox web browser, you will need to install an extension for Firefox that allows Rapise to interact with it:

1. Launch Firefox. Navigate to the following URL: http://www.inflectra.com/Rapise/Downloads.aspx:

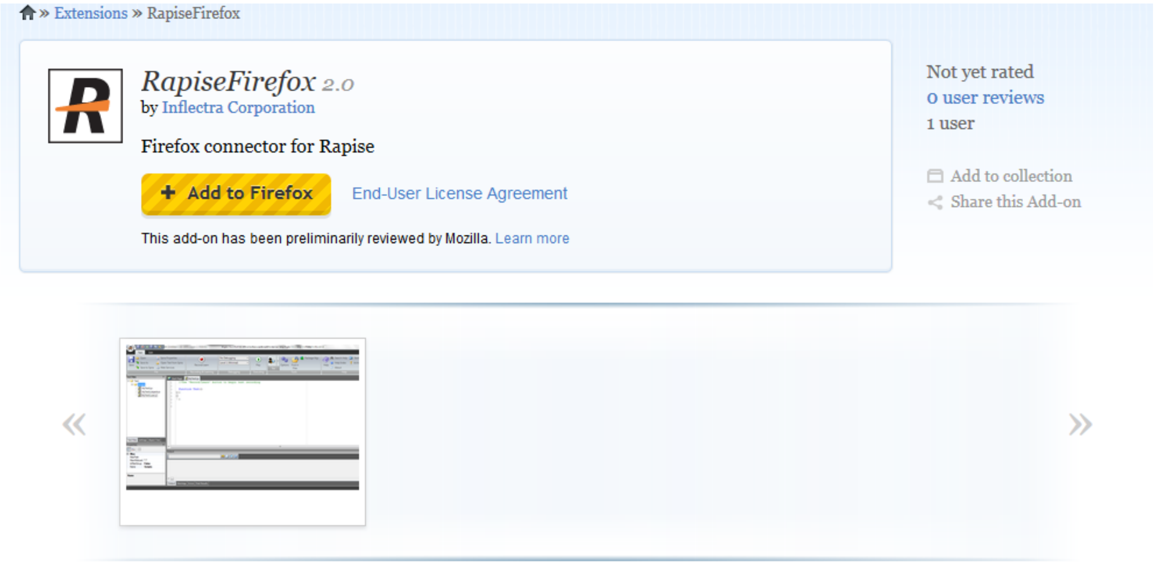


Available Downloads

The following downloads are currently available for Rapise:

Web-Browser Extensions	
v2.0.0	**Rapise Add-On For Firefox** Allows Rapise to record, learn and playback automated tests using the Mozilla Firefox web browser. *Compatible with Mozilla Firefox 20+ and Rapise v2.0+*
v2.0.3	**Rapise Extension for Google Chrome.** Allows Rapise to record, learn and playback automated tests using the Google Chrome browser. *Compatible with Google Chrome 32.0+ and Rapise v2.0+*

2. Click on the 'Rapise Add-In For Firefox' hyperlink and it will take you to the Firefox Add-Ons page:



» Extensions » RapiseFirefox

RapiseFirefox 2.0
by Inflectra Corporation

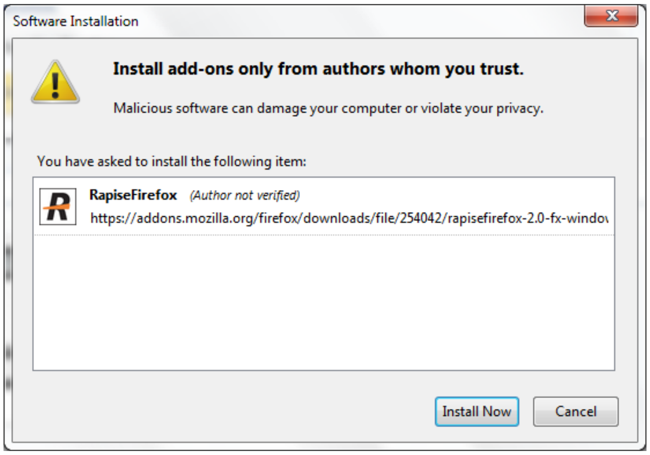
Firefox connector for Rapise

This add-on has been preliminarily reviewed by Mozilla. Learn more

Not yet rated
0 user reviews
1 user

3. Click on the 'Add to Firefox' button to install the Add-On into your instance of Firefox.

4. Click “Install Now” in the software installation dialog that appears:



Software Installation

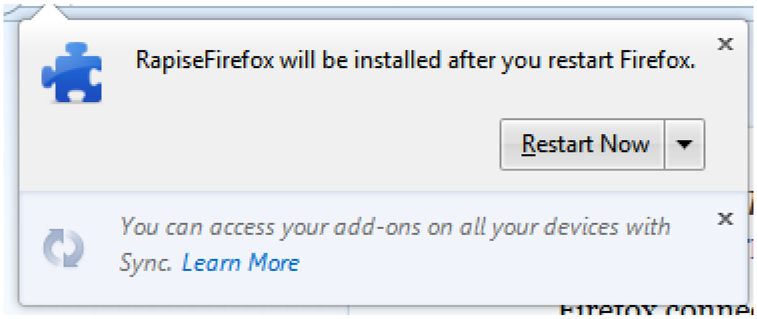
Install add-ons only from authors whom you trust.

Malicious software can damage your computer or violate your privacy.

You have asked to install the following item:

RapiseFirefox *(Author not verified)*
https://addons.mozilla.org/firefox/downloads/file/254042/rapisefirefox-2.0-fx-windo

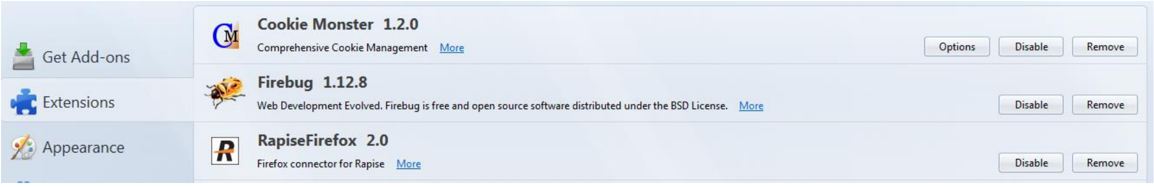
5. When installation is complete press “Restart Now” to complete the installation:



RapiseFirefox will be installed after you restart Firefox.

You can access your add-ons on all your devices with Sync. Learn More

6. Once Firefox is started again, you should now see the RapiseFirefox Add-On listed in the ‘Extensions’ section of the Firefox Add-ons Manager:



Cookie Monster 1.2.0
Comprehensive Cookie Management More

Firebug 1.12.8
Web Development Evolved. Firebug is free and open source software distributed under the BSD License. More

RapiseFirefox 2.0
Firefox connector for Rapise More

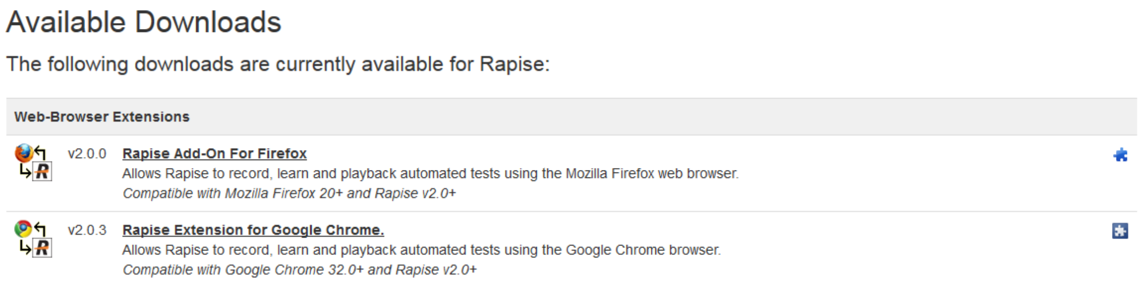
7. Firefox is ready to be used with Rapise for automated testing.

Note: Rapise requires localhost port 4247 to be accessible for correct operation. Please, make sure that this port is unblocked in your Firewall.

A.3. Preparing Google Chrome

In order to test web applications using the Google Chrome browser, you will need to install an extension for Chrome that allows Rapise to interact with it:

1. Launch Chrome. Navigate to the following URL: http://www.inflectra.com/Rapise/Downloads.aspx:



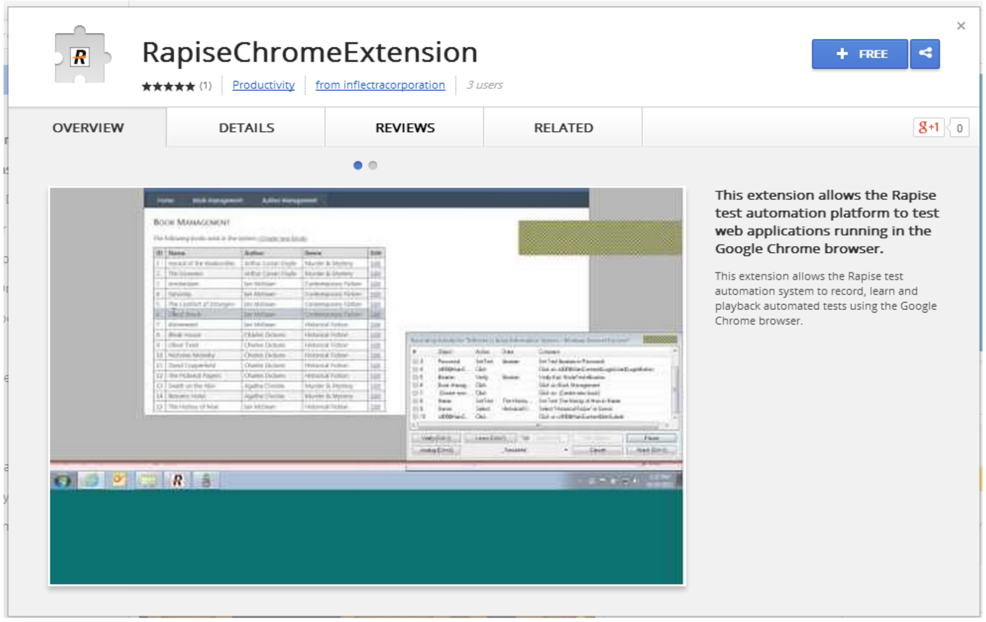
Available Downloads

The following downloads are currently available for Rapise:

Web-Browser Extensions

v2.0.0	**Rapise Add-On For Firefox** Allows Rapise to record, learn and playback automated tests using the Mozilla Firefox web browser. *Compatible with Mozilla Firefox 20+ and Rapise v2.0+*	
v2.0.3	**Rapise Extension for Google Chrome.** Allows Rapise to record, learn and playback automated tests using the Google Chrome browser. *Compatible with Google Chrome 32.0+ and Rapise v2.0+*	

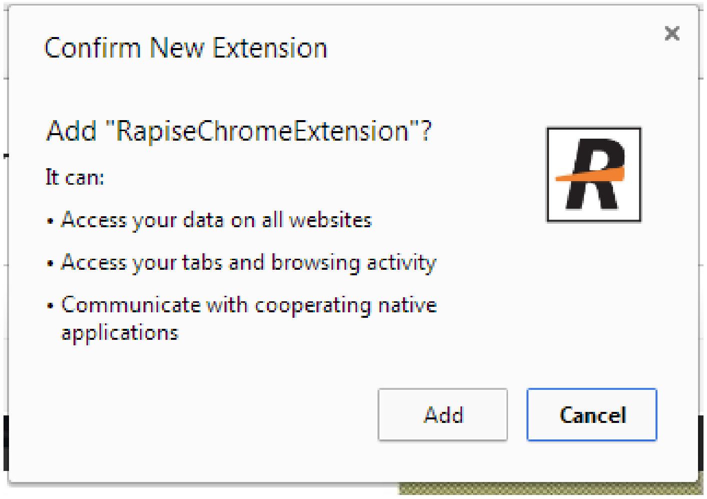
2. Click on the ‘Rapise Extension for Google Chrome’ hyperlink and it will take you to the Chome store:



This extension allows the Rapise test automation platform to test web applications running in the Google Chrome browser.

This extension allows the Rapise test automation system to record, learn and playback automated tests using the Google Chrome browser.

3. Click on the ‘Free’ button to install the Rapise extension into your instance of Chrome. You will be asked to confirm that you want to Add the extension:



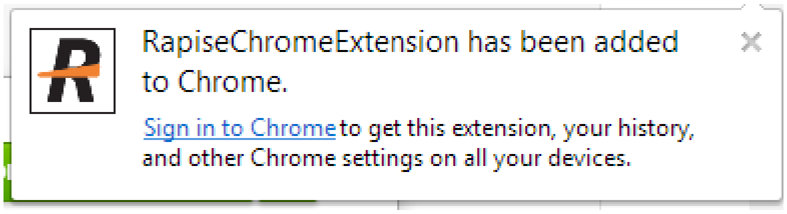
Confirm New Extension

Add "RapiseChromeExtension"?

It can:

- Access your data on all websites
- Access your tabs and browsing activity
- Communicate with cooperating native applications

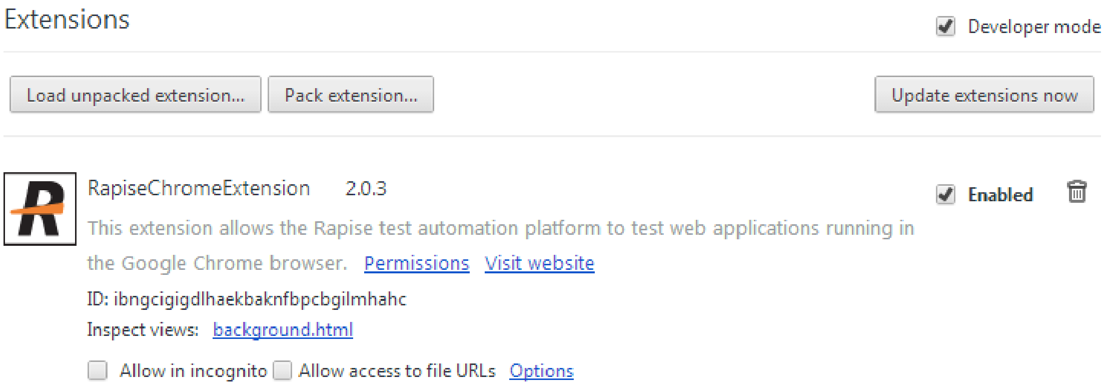
4. Once the extension has been installed, you will see the following confirmation dialog box:



RapiseChromeExtension has been added to Chrome.

Sign in to Chrometo get this extension, your history, and other Chrome settings on all your devices.

5. To view the extension, click on the “wrench” icon next to the omnibox and choose Tools > extensions. You should now see the chrome extension manager with the Rapise extension listed:



RapiseChromeExtension 2.0.3

This extension allows the Rapise test automation platform to test web applications running in the Google Chrome browser. Permissions Visit website

ID: ibngcigigdlhaekbaknfbpcbgilmhahc

Inspect views: background.html

□ Allow in incognito □ Allow access to file URLs Options

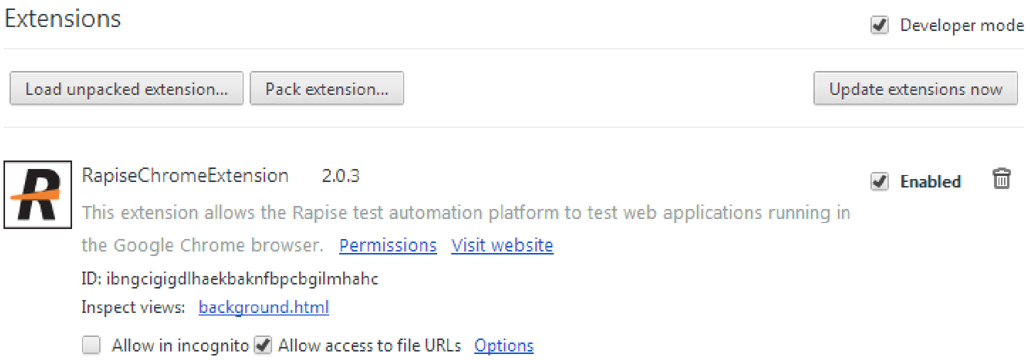
6. It is recommended to set the "Allow access to file URLs" checkbox so that web pages opened from the local folder (using the file:// protocol) can be also accessed by Rapise when running automated browser tests.

Testing Local Applications

The Rapise Chrome extension is not loaded by Chrome if a web page is opened from the local folder (file://). To overcome this restriction make sure that "Allow access to local URLs" checkbox is set.

1. In Chrome type following URL: chrome://extensions If you don't see the omnibox (the field where to type URL) please press Ctrl+T to show new chrome window.

2. 2. Set "Allow access to local URLs" checkbox:



RapiseChromeExtension 2.0.3

This extension allows the Rapise test automation platform to test web applications running in the Google Chrome browser. Permissions Visit website

ID: ibngcigigdlhaekbaknfbpcbgilmhahc

Inspect views: background.html

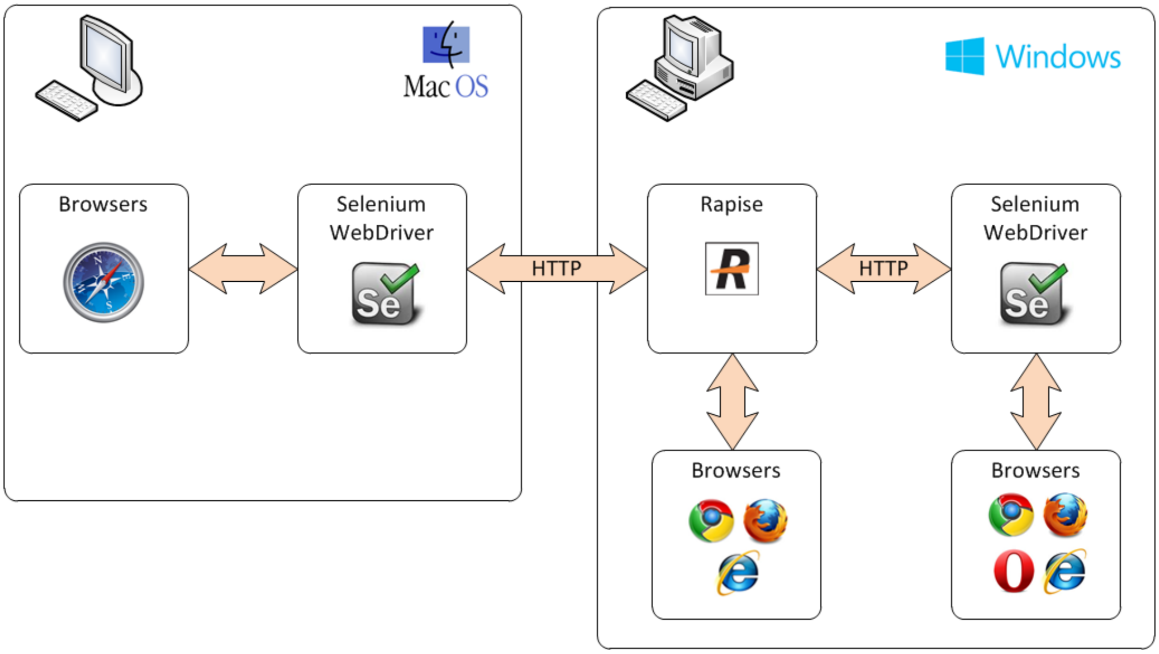
□ Allow in incognito ☑ Allow access to file URLs Options

Appendix B – Setting up Selenium WebDriver

This section describes the process for setting up Rapise to work with Selenium. Since Rapise is a Windows® application, you can use a single computer running Rapise to use the following web browsers:

- Internet Explorer
- Google Chrome
- Mozilla Firefox
- Opera
- Microsoft Edge

However because Safari only runs on Apple Mac computers, you will need to use two computers (a Mac running Safari) and a PC running Rapise to test using the Apple Safari web browser:

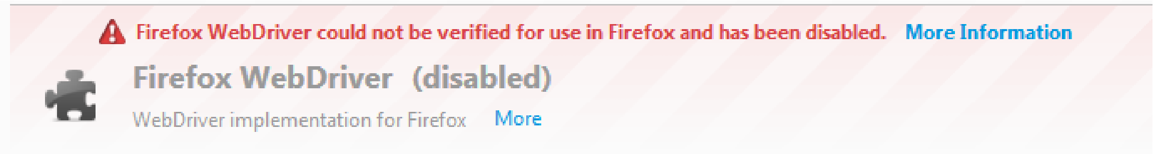


B.1. Configuring Selenium on a PC

Once you have installed Rapise on your local computer, you need to perform the following steps to configure each of the web browsers to use Selenium and Rapise:

Firefox

Unlike the other web browsers, Firefox does not require anything special to be done, it already includes a built-in plugin for use by Selenium WebDriver. However if you start using Rapise with Firefox and you see the following issue inside of Firefox:

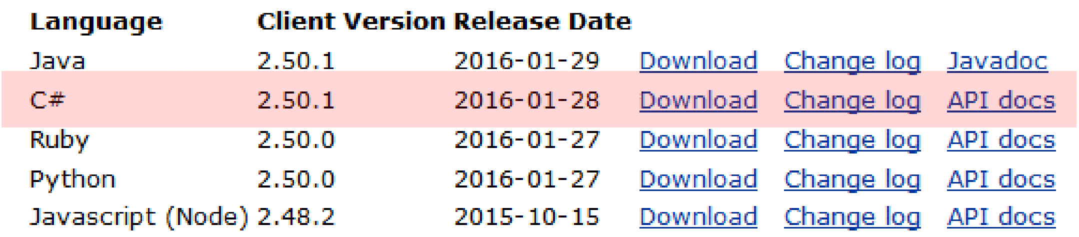


Firefox WebDriver could not be verified for use in Firefox and has been disabled. More Information

Firefox WebDriver (disabled)

WebDriver implementation for Firefox More

Then it means that the version of Selenium WebDriver that shipped with Rapise is no longer compatible with the installed version for Firefox. The solution is straightforward, just go to the main Selenium website: http://www.seleniumhq.org/download/ and then download the **C# WebDriver Bindings**:



Language	Client Version	Release Date			
Java	2.50.1	2016-01-29	Download	Change log	Javadoc
C#	2.50.1	2016-01-28	Download	Change log	API docs
Ruby	2.50.0	2016-01-27	Download	Change log	API docs
Python	2.50.0	2016-01-27	Download	Change log	API docs
Javascript (Node)	2.48.2	2015-10-15	Download	Change log	API docs

Download the **Selenium-dotnot-x.x.x.zip** file from the website. Proceed to unzip the archive and then look in the **net40** subfolder and extract the following two files and copy into the **C:\Program Files (x86)\Inflectra\Rapise\Bin** folder (or wherever you installed Rapise):

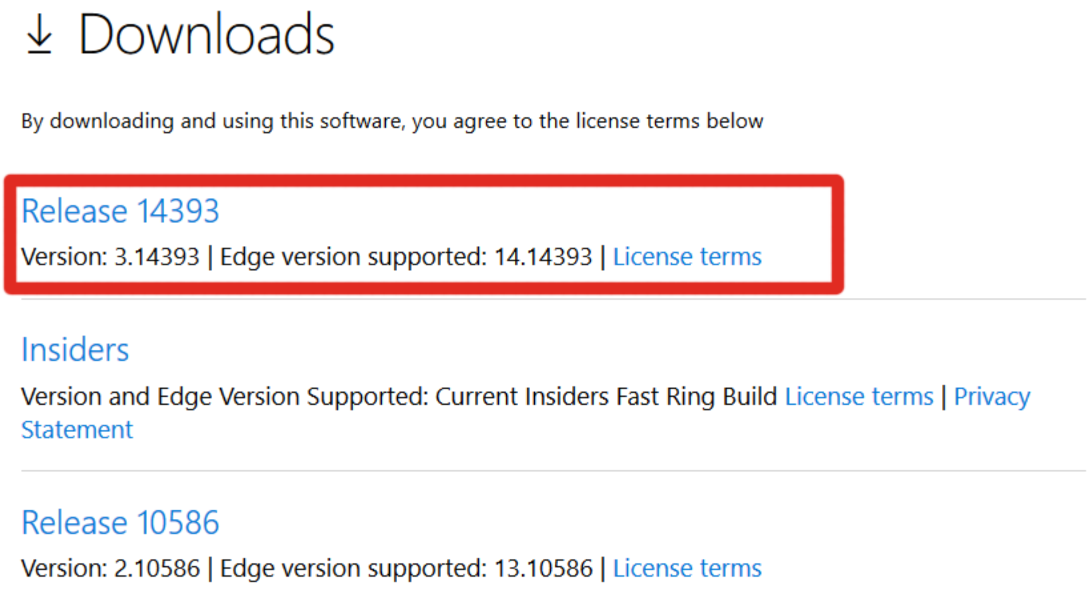
- WebDriver.dll
- WebDriver.Support.dll

Note: You will need to close Rapise before copying these files into the Bin folder.

Microsoft Edge

To use Selenium with Microsoft Edge, you will need to download the latest version of the Edge Driver from the Microsoft website:

https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/#downloads



↓ Downloads

By downloading and using this software, you agree to the license terms below

Release 14393

Version: 3.14393 | Edge version supported: 14.14393 | License terms

Insiders

Version and Edge Version Supported: Current Insiders Fast Ring Build License terms | Privacy Statement

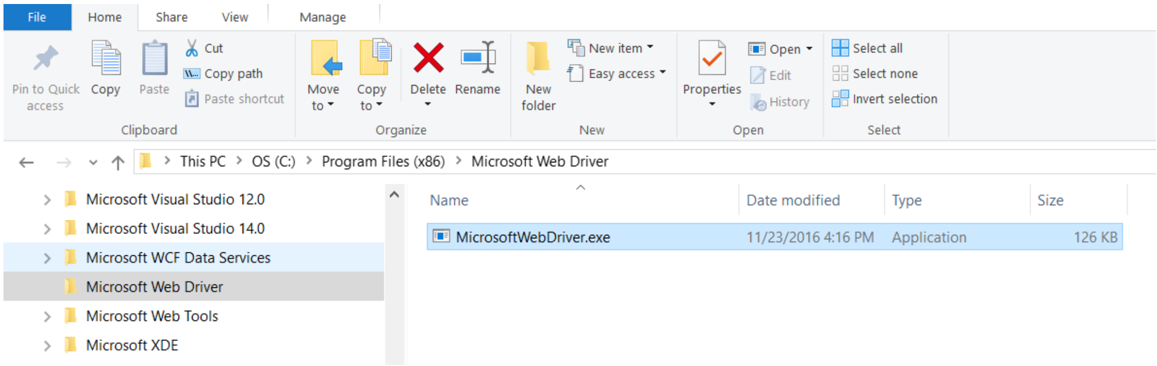
Release 10586

Version: 2.10586 | Edge version supported: 13.10586 | License terms

Create a new folder on your local PC called:

C:\Program Files (x86)\Microsoft Web Driver

Download the **MicrosoftWebDriver.exe** to this local folder you just created:



This PC > OS (C:) > Program Files (x86) > Microsoft Web Driver

- Microsoft Visual Studio 12.0
- Microsoft Visual Studio 14.0
- Microsoft WCF Data Services
- Microsoft Web Driver
- Microsoft Web Tools
- Microsoft XDE

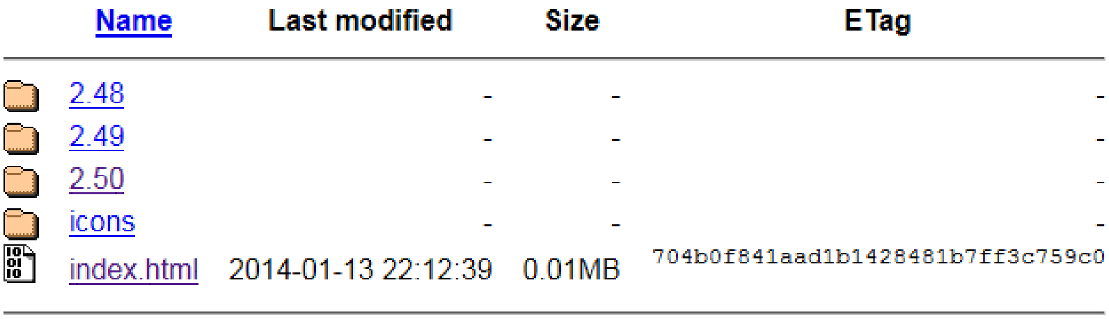
Name	Date modified	Type	Size
MicrosoftWebDriver.exe	11/23/2016 4:16 PM	Application	126 KB

Internet Explorer

To use Selenium with Internet Explorer, you will need to download the latest version of the Internet Explorer IE Driver:

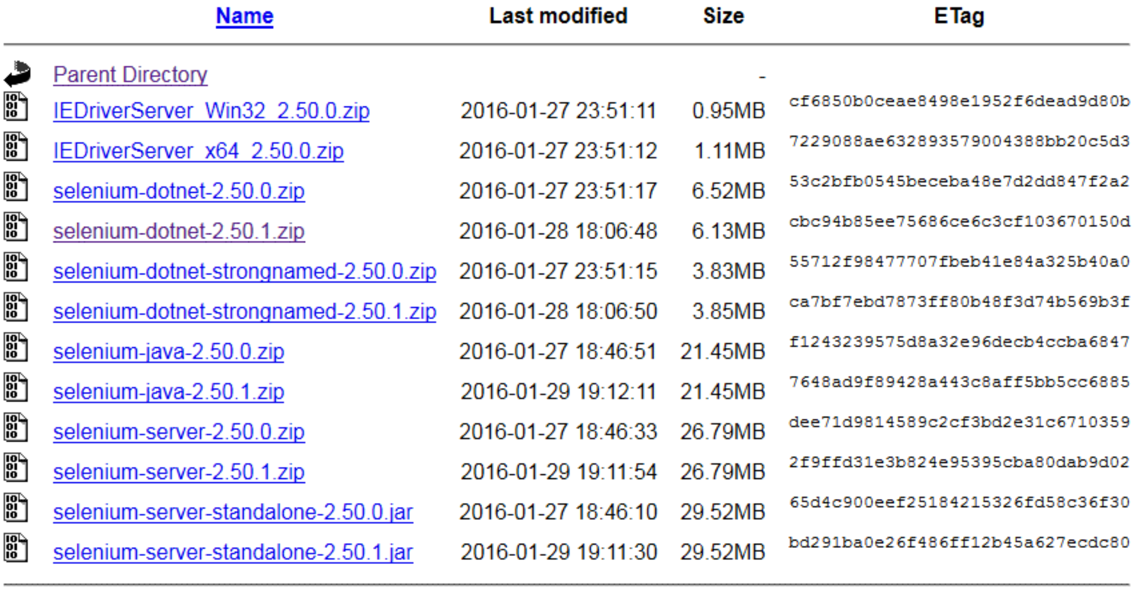
http://selenium-release.storage.googleapis.com/index.html

The list of versions at time of writing was:



Name	Last modified	Size	ETag
2.48	-	-	-
2.49	-	-	-
2.50	-	-	-
icons	-	-	-
index.html	2014-01-13 22:12:39	0.01MB	704b0f841aad1b1428481b7ff3c759c0

When you click on the folder for the latest version you will see the various files that can be downloaded:



Name	Last modified	Size	ETag
Parent Directory		-	
IEDriverServer_Win32_2.50.0.zip	2016-01-27 23:51:11	0.95MB	cf6850b0ceae8498e1952f6dead9d80b
IEDriverServer_x64_2.50.0.zip	2016-01-27 23:51:12	1.11MB	7229088ae632893579004388bb20c5d3
selenium-dotnet-2.50.0.zip	2016-01-27 23:51:17	6.52MB	53c2bfb0545beceba48e7d2dd847f2a2
selenium-dotnet-2.50.1.zip	2016-01-28 18:06:48	6.13MB	cbc94b85ee75686ce6c3cf103670150d
selenium-dotnet-strongnamed-2.50.0.zip	2016-01-27 23:51:15	3.83MB	55712f98477707fbeb41e84a325b40a0
selenium-dotnet-strongnamed-2.50.1.zip	2016-01-28 18:06:50	3.85MB	ca7bf7ebd7873ff80b48f3d74b569b3f
selenium-java-2.50.0.zip	2016-01-27 18:46:51	21.45MB	f1243239575d8a32e96decb4ccba6847
selenium-java-2.50.1.zip	2016-01-29 19:12:11	21.45MB	7648ad9f89428a443c8aff5bb5cc6885
selenium-server-2.50.0.zip	2016-01-27 18:46:33	26.79MB	dee71d9814589c2cf3bd2e31c6710359
selenium-server-2.50.1.zip	2016-01-29 19:11:54	26.79MB	2f9ffd31e3b824e95395cba80dab9d02
selenium-server-standalone-2.50.0.jar	2016-01-27 18:46:10	29.52MB	65d4c900eef25184215326fd58c36f30
selenium-server-standalone-2.50.1.jar	2016-01-29 19:11:30	29.52MB	bd291ba0e26f486ff12b45a627ecdc80

Download the **IEDriverServer_XXXX_X.X.X.zip** to your local PC:

- IEDriverServer_Win32_X.X.X.zip (for 32-bit Internet Explorer)
- IEDriverServer_x64_X.X.X.zip (for 64-bit Internet Explorer)

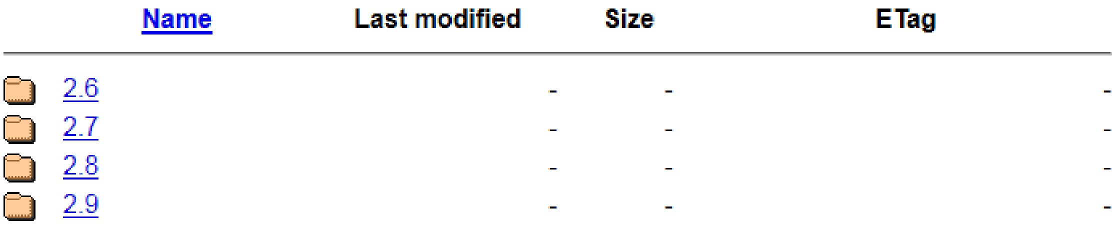
The file inside the zip archive is called **IEDriverServer.exe** and you need to copy it into the **C:\Program Files (x86)\Inflectra\Rapise\Bin** folder (or wherever you installed Rapise).

Chrome

To use Selenium with Google Chrome, you will need to download the latest version of the Chrome Driver:

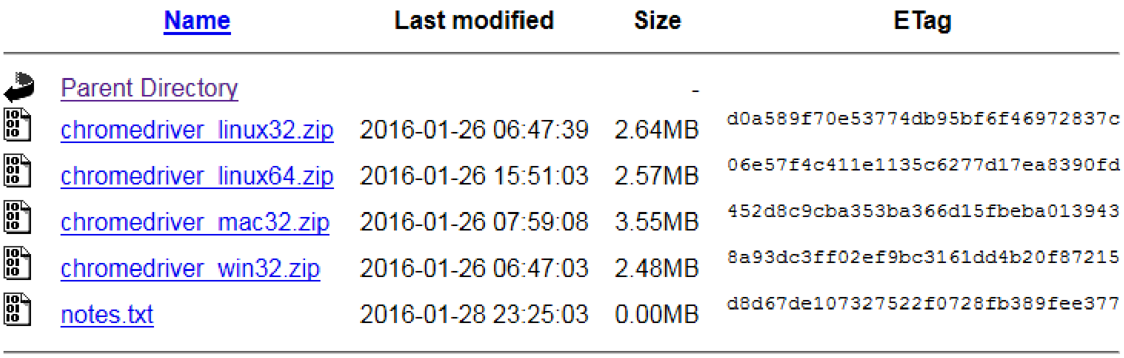
http://chromedriver.storage.googleapis.com/index.html

The list of versions at time of writing was:



Name	Last modified	Size	ETag
2.6	-	-	-
2.7	-	-	-
2.8	-	-	-
2.9	-	-	-

When you click on the folder for the latest version you will see the various files that can be downloaded:



Name	Last modified	Size	ETag
Parent Directory		-	
chromedriver_linux32.zip	2016-01-26 06:47:39	2.64MB	d0a589f70e53774db95bf6f46972837c
chromedriver_linux64.zip	2016-01-26 15:51:03	2.57MB	06e57f4c411e1135c6277d17ea8390fd
chromedriver_mac32.zip	2016-01-26 07:59:08	3.55MB	452d8c9cba353ba366d15fbeba013943
chromedriver_win32.zip	2016-01-26 06:47:03	2.48MB	8a93dc3ff02ef9bc3161dd4b20f87215
notes.txt	2016-01-28 23:25:03	0.00MB	d8d67de107327522f0728fb389fee377

Download the **chromedriver_win32.zip** to your local PC.

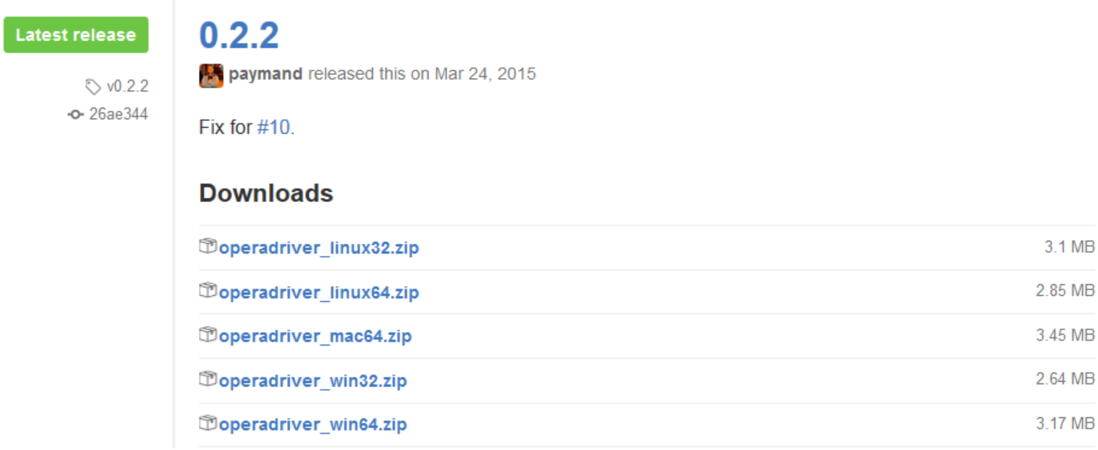
The file inside the zip archive is called **chromedriver.exe** and you need to copy it into the **C:\Program Files (x86)\Inflectra\Rapise\Bin** folder (or wherever you installed Rapise).

Opera

To use Selenium with Opera, you will need to download the latest version of the Opera Driver:

https://github.com/operasoftware/operachromiumdriver/releases

This page will list the latest version of the driver at the top of the page:



Latest release

v0.2.2

26ae344

0.2.2

paymand released this on Mar 24, 2015

Fix for #10.

Downloads

operadriver_linux32.zip	3.1 MB
operadriver_linux64.zip	2.85 MB
operadriver_mac64.zip	3.45 MB
operadriver_win32.zip	2.64 MB
operadriver_win64.zip	3.17 MB

Download the **operadriver_winXX.zip** to your local PC:

- operadriver_win32.zip (for 32-bit Opera)
- operadriver_win64.zip (for 64-bit Opera)

The file inside the zip archive is called **operadriver.exe** and you need to copy it into the **C:\Program Files (x86)\Inflectra\Rapise\Bin** folder (or wherever you installed Rapise).

B.2. Installing Selenium on a Mac

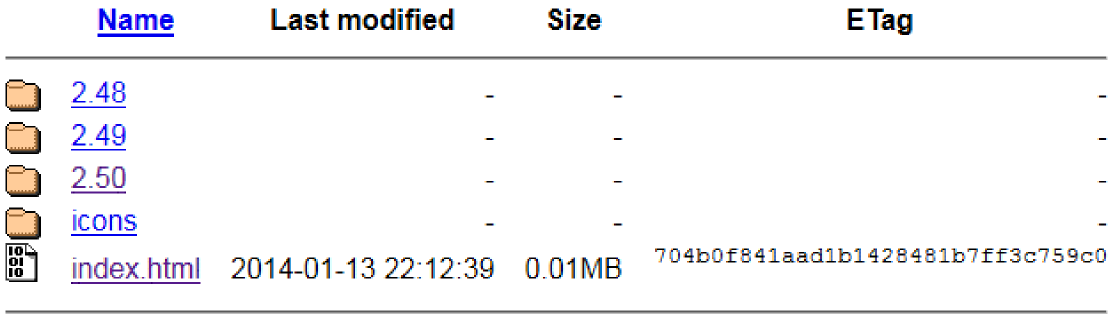
The reason for using Selenium running on a Mac is to be able to execute tests against the Safari web browser. So although you can also use the Mac to test with Firefox, Opera and Chrome, we do not recommend this as it adds needless complexity.

Safari

The first thing you need to do is download the latest version of the Selenium server for Apple Mac computers:

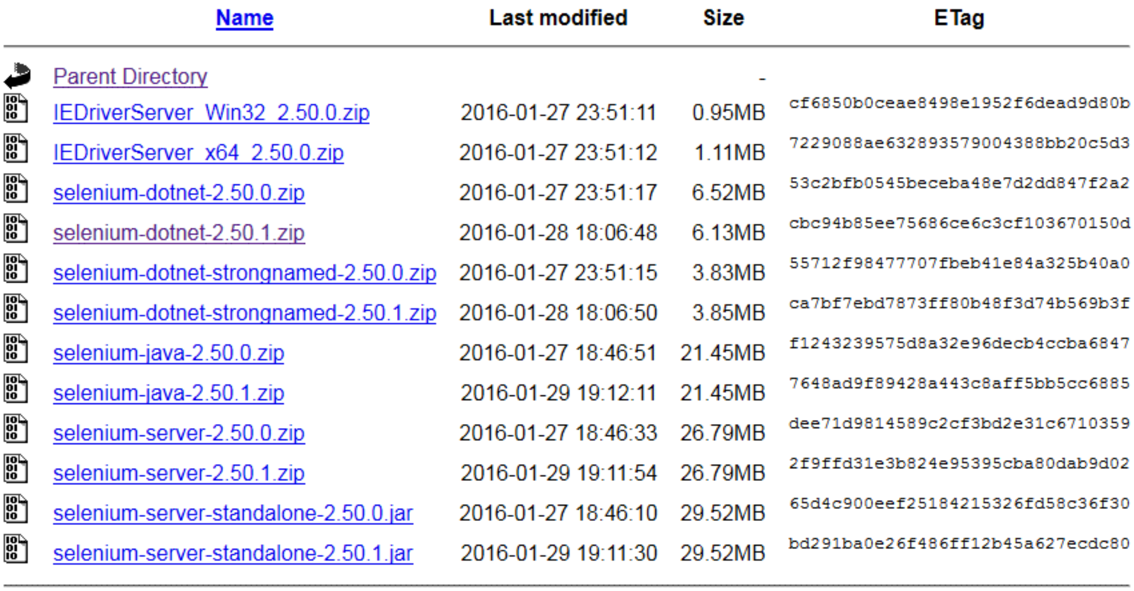
http://selenium-release.storage.googleapis.com/index.html

The list of versions at time of writing was:



Name	Last modified	Size	ETag
2.48	-	-	-
2.49	-	-	-
2.50	-	-	-
icons	-	-	-
index.html	2014-01-13 22:12:39	0.01MB	704b0f841aad1b1428481b7ff3c759c0

When you click on the folder for the latest version you will see the various files that can be downloaded:



Name	Last modified	Size	ETag
Parent Directory		-	
IEDriverServer_Win32_2.50.0.zip	2016-01-27 23:51:11	0.95MB	cf6850b0ceae8498e1952f6dead9d80b
IEDriverServer_x64_2.50.0.zip	2016-01-27 23:51:12	1.11MB	7229088ae63289357900438 8bb20c5d3
selenium-dotnet-2.50.0.zip	2016-01-27 23:51:17	6.52MB	53c2bfb0545beceba48e7d2dd847f2a2
selenium-dotnet-2.50.1.zip	2016-01-28 18:06:48	6.13MB	cbc94b85ee75686ce6c3cf103670150d
selenium-dotnet-strongnamed-2.50.0.zip	2016-01-27 23:51:15	3.83MB	55712f98477707fbeb41e84a325b40a0
selenium-dotnet-strongnamed-2.50.1.zip	2016-01-28 18:06:50	3.85MB	ca7bf7ebd7873ff80b48f3d74b569b3f
selenium-java-2.50.0.zip	2016-01-27 18:46:51	21.45MB	f1243239575d8a32e96decb4ccba6847
selenium-java-2.50.1.zip	2016-01-29 19:12:11	21.45MB	7648ad9f89428a443c8aff5bb5cc6885
selenium-server-2.50.0.zip	2016-01-27 18:46:33	26.79MB	dee71d9814589c2cf3bd2e31c6710359
selenium-server-2.50.1.zip	2016-01-29 19:11:54	26.79MB	2f9ffd31e3b824e95395cba80dab9d02
selenium-server-standalone-2.50.0.jar	2016-01-27 18:46:10	29.52MB	65d4c900eef25184215326fd58c36f30
selenium-server-standalone-2.50.1.jar	2016-01-29 19:11:30	29.52MB	bd291ba0e26f486ff12b45a627ecdc80

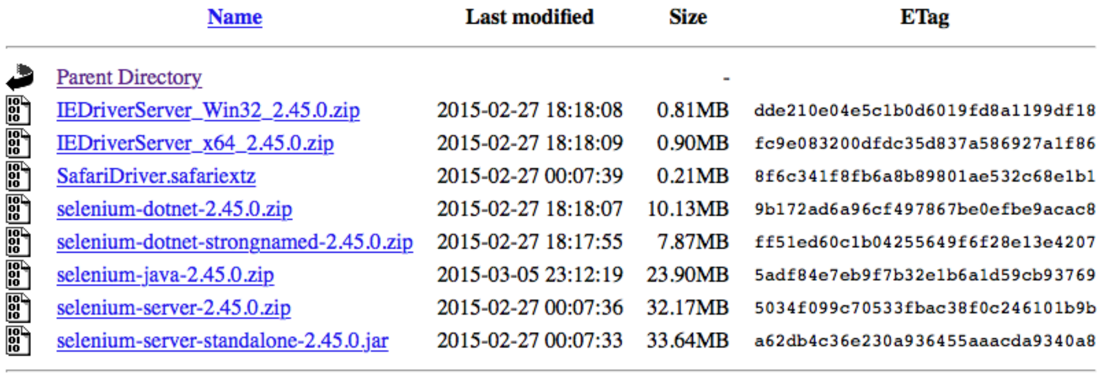
Download the **selenium-server-standalone-X.XX.X.jar** to the Mac.

Run this Java application by double clicking the downloaded .JAR file in Finder. This will startup the Selenium server.

Note: You will need to have the Java (ideally the latest version) installed on the Mac first.

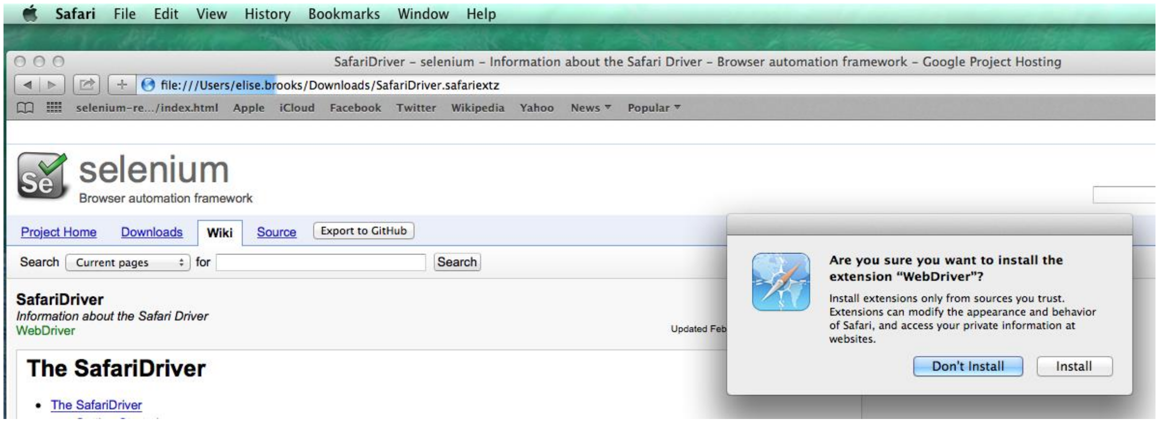
Once you have this running, you will need to then download the actual Safari WebDriver plugin. This can be found at the following location:

https://github.com/SeleniumHQ/selenium/wiki/SafariDriver



Name	Last modified	Size	ETag
Parent Directory		-	
IEDriverServer_Win32_2.45.0.zip	2015-02-27 18:18:08	0.81MB	dde210e04e5c1b0d6019fd8a1199df18
IEDriverServer_x64_2.45.0.zip	2015-02-27 18:18:09	0.90MB	fc9e083200dfdc35d837a586927a1f86
SafariDriver.safariextz	2015-02-27 00:07:39	0.21MB	8f6c341f8fb6a8b89801ae532c68e1b1
selenium-dotnet-2.45.0.zip	2015-02-27 18:18:07	10.13MB	9b172ad6a96cf497867be0efbe9acac8
selenium-dotnet-strongnamed-2.45.0.zip	2015-02-27 18:17:55	7.87MB	ff51ed60c1b04255649f6f28e13e4207
selenium-java-2.45.0.zip	2015-03-05 23:12:19	23.90MB	5adf84e7eb9f7b32e1b6a1d59cb93769
selenium-server-2.45.0.zip	2015-02-27 00:07:36	32.17MB	5034f099c70533fbac38f0c246101b9b
selenium-server-standalone-2.45.0.jar	2015-02-27 00:07:33	33.64MB	a62db4c36e230a936455aaacda9340a8

Download the **SafariDriver.safariextz** file to the local computer and the double-click to install in Safari:



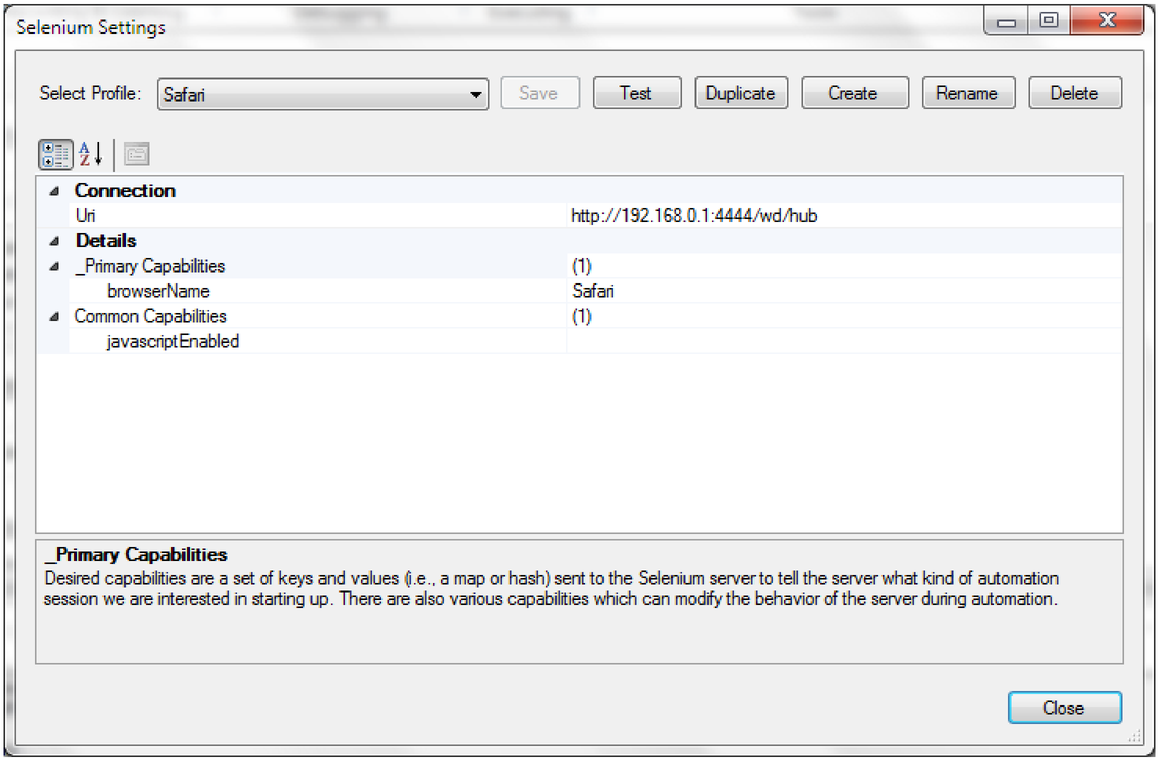
SafariDriver – selenium – Information about the Safari Driver – Browser automation framework – Google Project Hosting

file:///Users/elise.brooks/Downloads/SafariDriver.safariextz

Are you sure you want to install the extension "WebDriver"?

Install extensions only from sources you trust. Extensions can modify the appearance and behavior of Safari, and access your private information at websites.

Once that has been installed, you are now ready to test web applications running on Safari. The final step is to tell Rapise where it can find that instance of Selenium. To do that, open up Rapise (on your PC) and click on **Options > Tools** and then click on the **'Selenium Settings…'** entry:



Connection

Uri	http://192.168.0.1:4444/wd/hub

Details

_Primary Capabilities	(1)
browserName	Safari
Common Capabilities	(1)
javascriptEnabled	

_Primary Capabilities

Desired capabilities are a set of keys and values (i.e., a map or hash) sent to the Selenium server to tell the server what kind of automation session we are interested in starting up. There are also various capabilities which can modify the behavior of the server during automation.

Now you need to change the **Uri** field to point to your Mac. The format of the URI will be:

- `http://<IP or DNS name of MAC computer>:4444/wd/hub`

(for example it could be http://test-mac01.local:4444/wd/hub or http://192.168.0.52:4444/wd/hub)

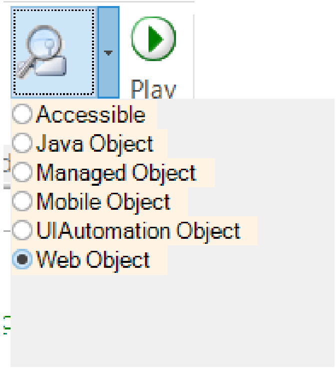
Appendix C- Using JavaScript Tests

This section will demonstrate how you can use Rapise to inspect the objects in a web page and Learn them for testing using the **JavaScript test script language** instead of the **Rapise Visual Language (RVL)** that was illustrated in section 2.

This is useful in cases where you have more complex applications to test and you want to be able to use the power of a full programming language such as JavaScript to iterate over data, perform calculations and use if…then…else branches to follow different steps during the test.

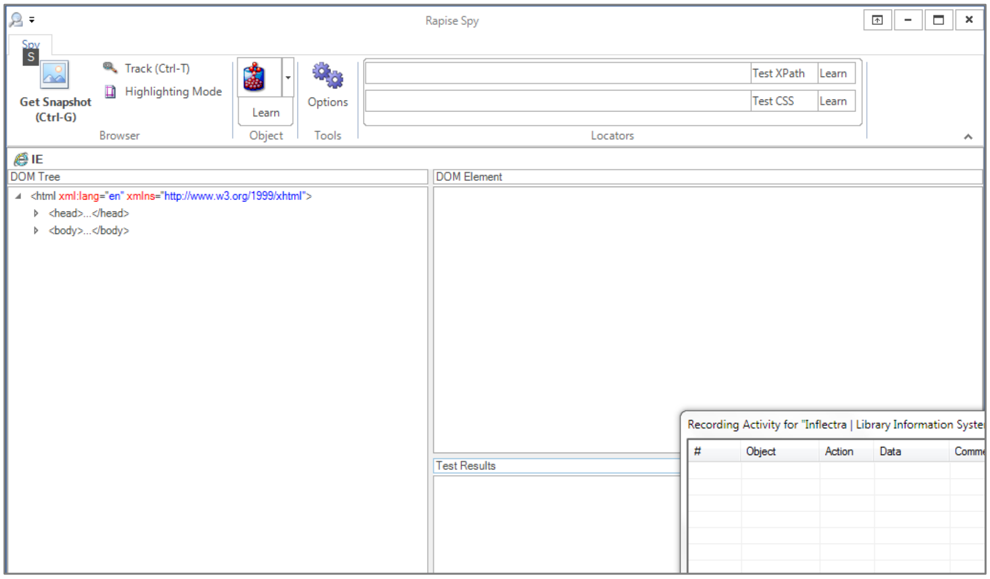
C.1. Learning an Object Using the Web Spy

In the main Test ribbon of Rapise, expand the dropdown list for the **'Spy**' tool and make sure that 'Web Object' is selected:



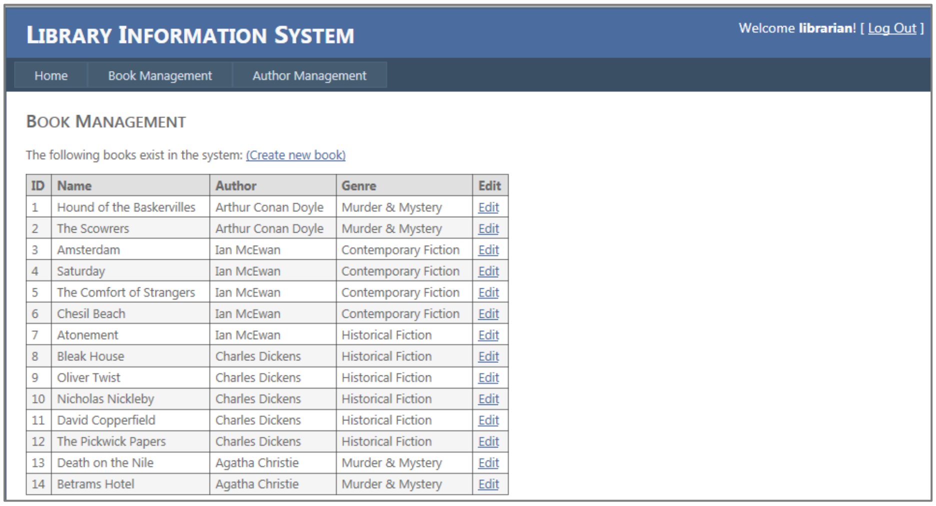
○ Accessible
○ Java Object
○ Managed Object
○ Mobile Object
○ UIAutomation Object
● Web Object

Now, click on the main 'Spy' tool icon and Rapise will start the Web Spy:



```
◢ <html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
   ▷ <head>...</head>
   ▷ <body>...</body>
```

Go back to the web page and login to the library information system with the same login/password (librarian/librarian) and click on the 'Book Management' menu item so that the list of books is displayed:

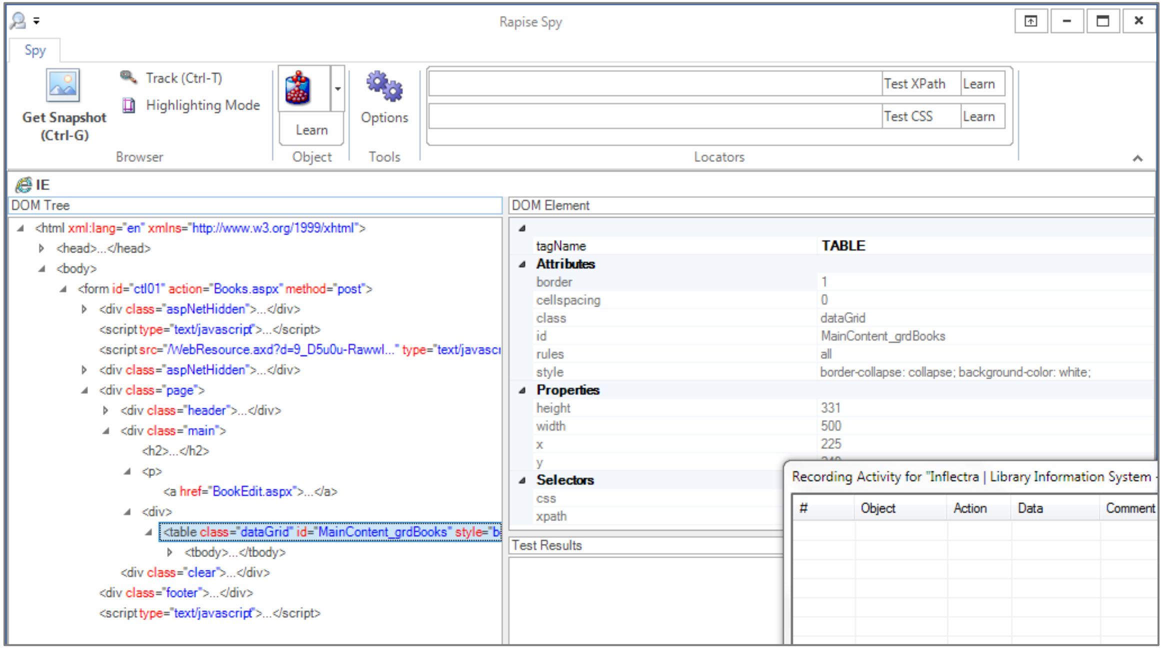


BOOK MANAGEMENT

The following books exist in the system: (Create new book)

ID	Name	Author	Genre	Edit
1	Hound of the Baskervilles	Arthur Conan Doyle	Murder & Mystery	Edit
2	The Scowrers	Arthur Conan Doyle	Murder & Mystery	Edit
3	Amsterdam	Ian McEwan	Contemporary Fiction	Edit
4	Saturday	Ian McEwan	Contemporary Fiction	Edit
5	The Comfort of Strangers	Ian McEwan	Contemporary Fiction	Edit
6	Chesil Beach	Ian McEwan	Contemporary Fiction	Edit
7	Atonement	Ian McEwan	Historical Fiction	Edit
8	Bleak House	Charles Dickens	Historical Fiction	Edit
9	Oliver Twist	Charles Dickens	Historical Fiction	Edit
10	Nicholas Nickleby	Charles Dickens	Historical Fiction	Edit
11	David Copperfield	Charles Dickens	Historical Fiction	Edit
12	The Pickwick Papers	Charles Dickens	Historical Fiction	Edit
13	Death on the Nile	Agatha Christie	Murder & Mystery	Edit
14	Betrams Hotel	Agatha Christie	Murder & Mystery	Edit

Now back in the Web Spy, click on the ‘Get Snapshot’ option to refresh the Web Spy and display the HTML elements (called the DOM tree) that make up this page:

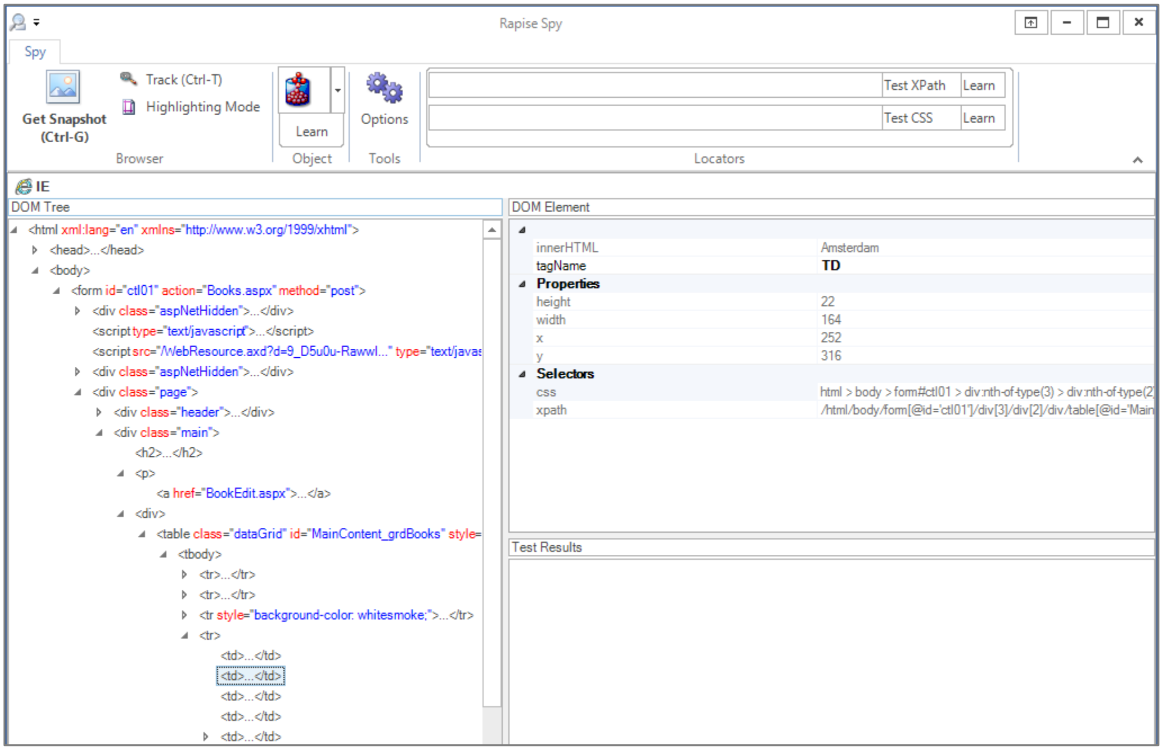


```
<html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
  <head>...</head>
  <body>
    <form id="ctl01" action="Books.aspx" method="post">
      <div class="aspNetHidden">...</div>
      <script type="text/javascript">...</script>
      <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." type="text/javascr
      <div class="aspNetHidden">...</div>
      <div class="page">
        <div class="header">...</div>
        <div class="main">
          <h2>...</h2>
          <p>
            <a href="BookEdit.aspx">...</a>
          <div>
            <table class="dataGrid" id="MainContent_grdBooks" style="b
              <tbody>...</tbody>
        <div class="clear">...</div>
      <div class="footer">...</div>
      <script type="text/javascript">...</script>
```

tagName	TABLE
Attributes	
border	1
cellspacing	0
class	dataGrid
id	MainContent_grdBooks
rules	all
style	border-collapse: collapse; background-color: white;
Properties	
height	331
width	500
x	225
y	
Selectors	
css	
xpath	

Once it has loaded the DOM tree, you can expand/collapse the elements to see how the web page is constructed. This is useful when testing an application since many of the HTML elements on a page may be used for layout purposes and will not be visible in the browser. In the example page, we have expanded some of the nodes to display the main section of the page and the table that contains the list of books.

In addition, you can use the **Track (Ctrl+T)** tool to select and item in the web page and then have it be highlighted in the DOM tree. For example if we want to find the cell that contains the book title “Amsterdam”, simply click CTRL+T on the keyboard, move the mouse over the cell in the webpage**, wait until the red highlighting rectangle appears** and then click CTRL+T again. Rapise will now highlight that item in the DOM Tree automatically:



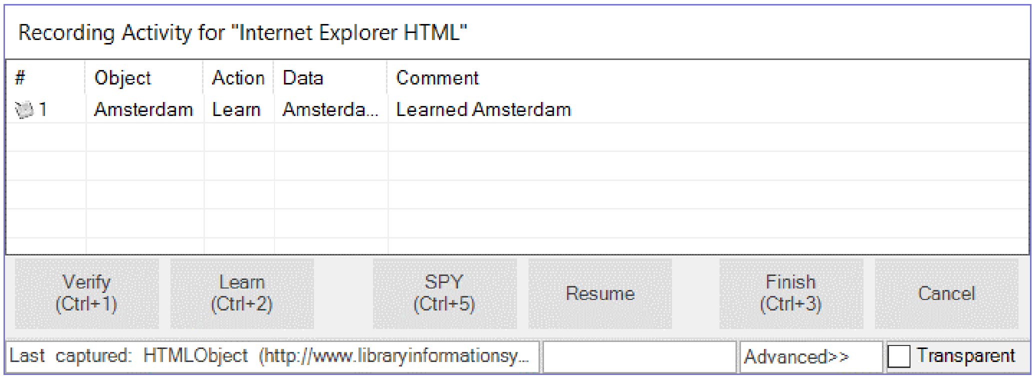
```
◢ <html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
    ▷ <head>...</head>
    ◢ <body>
        ◢ <form id="ctl01" action="Books.aspx" method="post">
            ▷ <div class="aspNetHidden">...</div>
              <script type="text/javascript">...</script>
              <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." type="text/javas
            ▷ <div class="aspNetHidden">...</div>
            ◢ <div class="page">
                ▷ <div class="header">...</div>
                ◢ <div class="main">
                      <h2>...</h2>
                    ◢ <p>
                          <a href="BookEdit.aspx">...</a>
                    ◢ <div>
                        ◢ <table class="dataGrid" id="MainContent_grdBooks" style=
                            ◢ <tbody>
                                ▷ <tr>...</tr>
                                ▷ <tr>...</tr>
                                ▷ <tr style="background-color: whitesmoke;">...</tr>
                                ◢ <tr>
                                      <td>...</td>
                                      <td>...</td>
                                      <td>...</td>
                                      <td>...</td>
                                    ▷ <td>...</td>
```

innerHTML	Amsterdam
tagName	**TD**
Properties	
height	22
width	164
x	252
y	316
Selectors	
css	html > body > form#ctl01 > div:nth-of-type(3) > div:nth-of-type(2
xpath	/html/body/form[@id='ctl01']/div[3]/div[2]/div/table[@id='Main

You can see all of the properties of this HTML element displayed on the right, specifically:

- The tagName is displayed as ‘TD’ (always upper case)
- The innerHTML of the element is displayed (Amsterdam)
- The CSS and XPath for locating this element is displayed

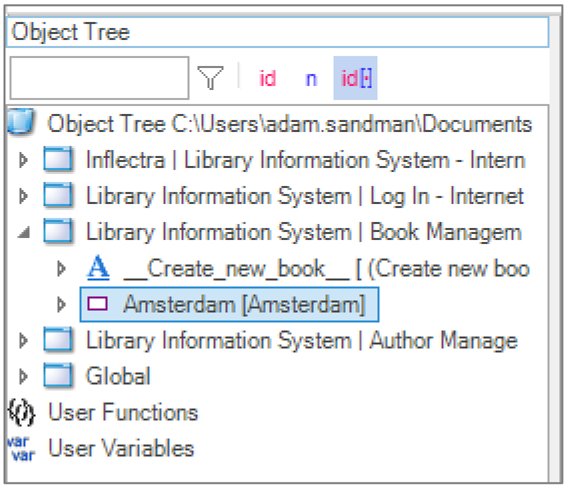
If you want to use this object in a Rapise test script, you can simply click the **Learn** button and the HTML element will be added to the Recording Activity Dialog:


Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Amsterdam	Learn	Amsterda...	Learned Amsterdam

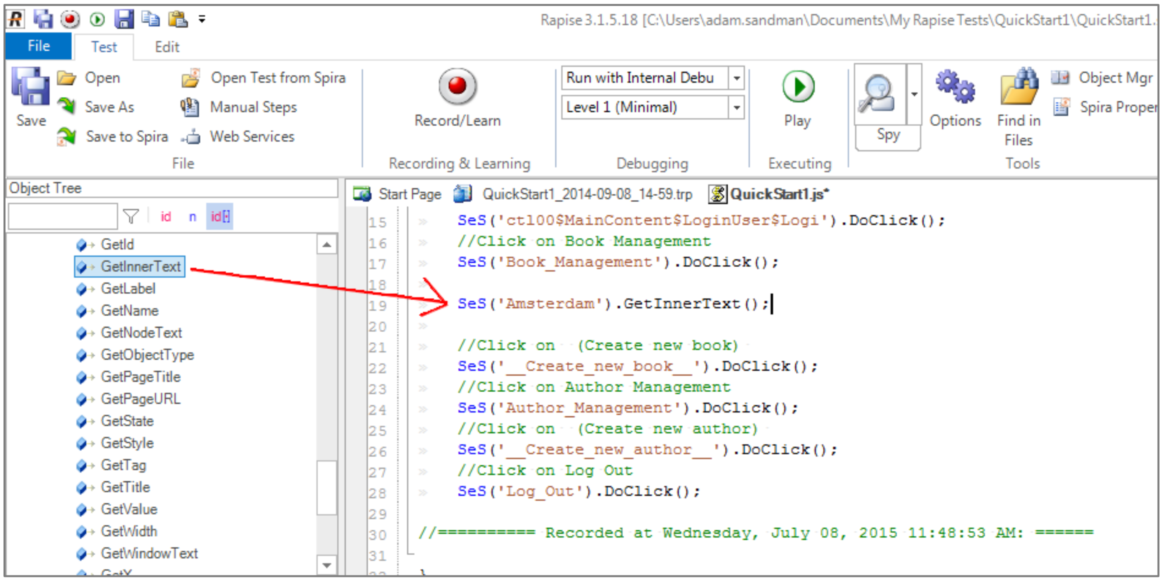
Last captured: HTMLObject (http://www.libraryinformationsy...

Click **Finish** and the object will have been added to your test’s Object Tree:



Object Tree
id
n
id[]
Object Tree C:\Users\adam.sandman\Documents
Inflectra | Library Information System - Intern
Library Information System | Log In - Internet
Library Information System | Book Managem
__Create_new_book__ [(Create new boo
Amsterdam [Amsterdam]
Library Information System | Author Manage
Global
User Functions
User Variables

You can now expand this object and drag a test function to your test script. For example if you wanted to get the textual value of the cell in your test, drag the “GetInnerText” function into your test script:



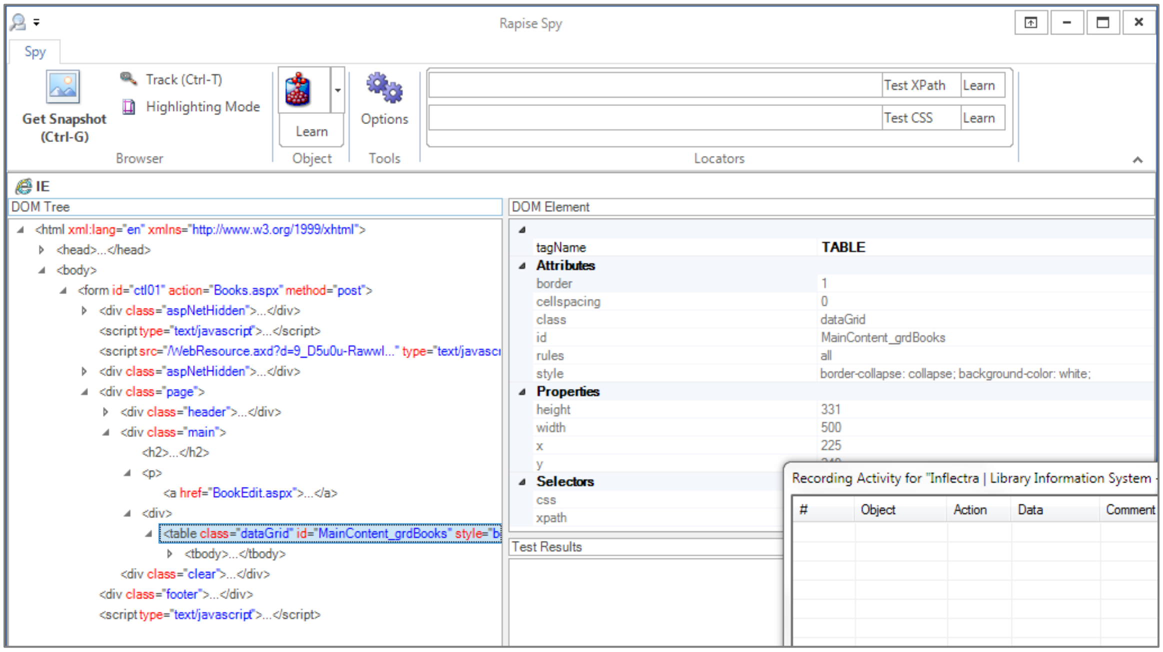
```
    SeS('ctl00$MainContent$LoginUser$Logi').DoClick();
    //Click on Book Management
    SeS('Book_Management').DoClick();

    SeS('Amsterdam').GetInnerText();

    //Click on  (Create new book)
    SeS('__Create_new_book__').DoClick();
    //Click on Author Management
    SeS('Author_Management').DoClick();
    //Click on  (Create new author)
    SeS('__Create_new_author__').DoClick();
    //Click on Log Out
    SeS('Log_Out').DoClick();

//========== Recorded at Wednesday, July 08, 2015 11:48:53 AM: ======
```

C.2. Learning an Object from XPATH

In addition to letting Rapise automatically learn the object from the Web Spy, you can manually enter in XPATH or CSS queries to find matching elements on the page and then learn those for use in your test.

For example, suppose we want to dynamically find the row that has the cell containing Amsterdam and then click on its Edit hyperlink.

Open up the Web Spy as before:



```
<html xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
  <head>...</head>
  <body>
    <form id="ctl01" action="Books.aspx" method="post">
      <div class="aspNetHidden">...</div>
      <script type="text/javascript">...</script>
      <script src="/WebResource.axd?d=9_D5u0u-Rawwl..." type="text/javasc
      <div class="aspNetHidden">...</div>
      <div class="page">
        <div class="header">...</div>
        <div class="main">
          <h2>...</h2>
          <p>
            <a href="BookEdit.aspx">...</a>
          <div>
            <table class="dataGrid" id="MainContent_grdBooks" style="b
              <tbody>...</tbody>
        <div class="clear">...</div>
      <div class="footer">...</div>
      <script type="text/javascript">...</script>
```

tagName	TABLE
Attributes	
border	1
cellspacing	0
class	dataGrid
id	MainContent_grdBooks
rules	all
style	border-collapse: collapse; background-color: white;
Properties	
height	331
width	500
x	225
y	
Selectors	
css	
xpath	

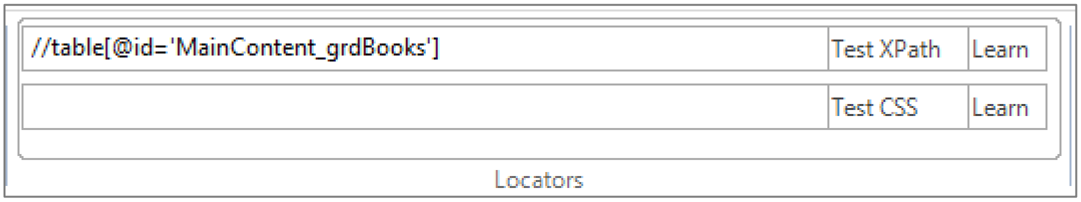
Recording Activity for "Inflectra | Library Information System

#	Object	Action	Data	Comment

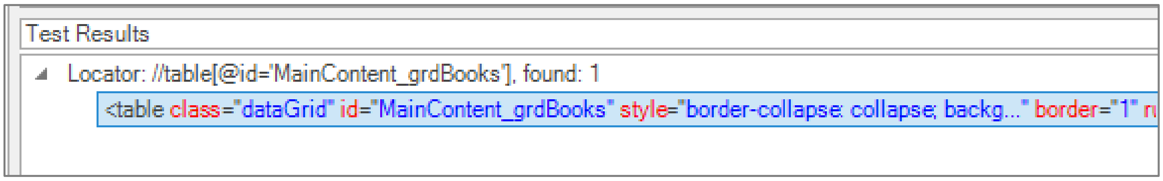
In the **Locators** section of the Web Spy, enter in the following to locate the table:

```
//table[@id='MainContent_grdBooks']
```

Now click on the **Test XPath** button to display the matching results:



//table[@id='MainContent_grdBooks']



Locator: //table[@id='MainContent_grdBooks'], found: 1

<table class="dataGrid" id="MainContent_grdBooks" style="border-collapse: collapse; backg..." border="1" ru

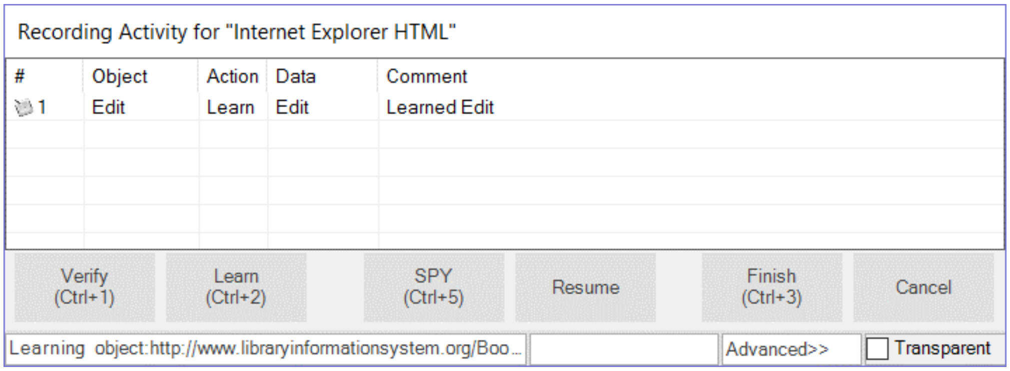
Now that we have matched the table, we need to add dynamic XPath to find any row that has the cell containing ‘Amsterdam’ and find the edit link. You can expand the table and see the rows and cells visually and that will help us create the XPATH:

```
//table[@id='MainContent_grdBooks']//tr/td[text()='Amsterdam']/../td[5]/a
```

This XPath consists of the following elements:

4. Finds the table with the specified ID
5. Find any row inside that table that contains a cell with the text ‘Amsterdam’
6. For any matching cell, get its parent row and inside the fifth cell, get any hyperlink

In this case that will correctly locate the Edit link for the book ‘Amsterdam’. Now that we have the correct item identified, click on the ‘**Learn’** button to the right of the ‘**Test XPath**’ button. That will now learn a new object that corresponds to the ‘Edit Amsterdam’ object:

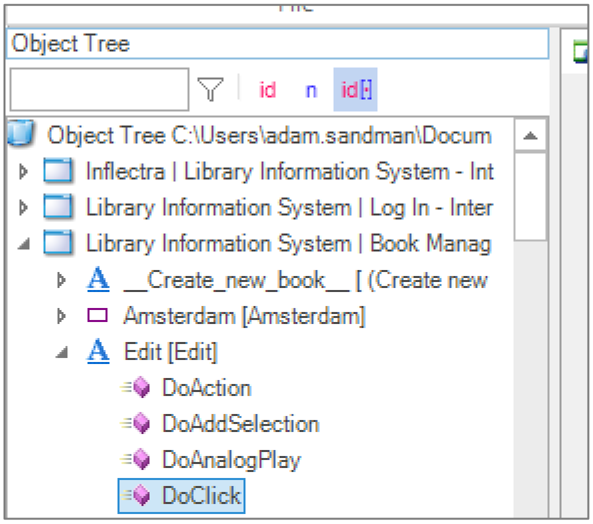


Recording Activity for "Internet Explorer HTML"

#	Object	Action	Data	Comment
1	Edit	Learn	Edit	Learned Edit

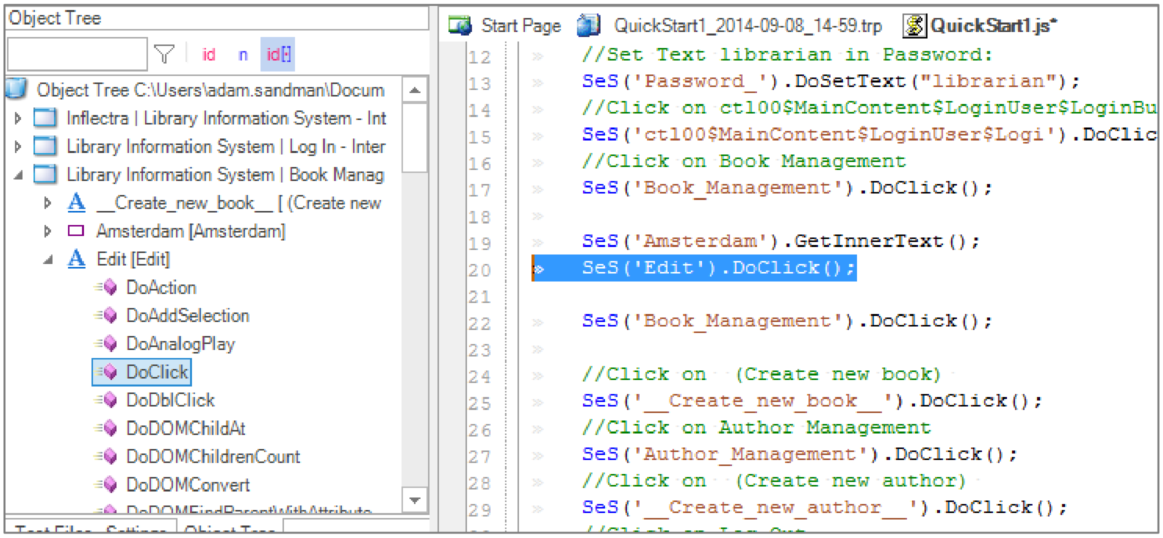
Learning object:http://www.libraryinformationsystem.org/Boo...

Click Finish and the object will have been added to the Object Tree of the current test:



Object Tree
id
n
id
Object Tree C:\Users\adam.sandman\Docum
Inflectra | Library Information System - Int
Library Information System | Log In - Inter
Library Information System | Book Manag
__Create_new_book__ [(Create new
Amsterdam [Amsterdam]
Edit [Edit]
DoAction
DoAddSelection
DoAnalogPlay
DoClick

To click on this object, simply drag the ‘DoClick’ function for the new object from the Object Tree into the test script. This will cause the book details page for Amsterdam to be display during test playback:



Object Tree
id
n
id
Object Tree C:\Users\adam.sandman\Docum
Inflectra | Library Information System - Int
Library Information System | Log In - Inter
Library Information System | Book Manag
__Create_new_book__ [(Create new
Amsterdam [Amsterdam]
Edit [Edit]
DoAction
DoAddSelection
DoAnalogPlay
DoClick
DoDblClick
DoDOMChildAt
DoDOMChildrenCount
DoDOMConvert
Start Page
QuickStart1_2014-09-08_14-59.trp
QuickStart1.js*

```
//Set Text librarian in Password:
SeS('Password_').DoSetText("librarian");
//Click on ctl00$MainContent$LoginUser$LoginBu
SeS('ctl00$MainContent$LoginUser$Logi').DoClic
//Click on Book Management
SeS('Book_Management').DoClick();

SeS('Amsterdam').GetInnerText();
SeS('Edit').DoClick();

SeS('Book_Management').DoClick();

//Click on  (Create new book)
SeS('__Create_new_book__').DoClick();
//Click on Author Management
SeS('Author_Management').DoClick();
//Click on  (Create new author)
SeS('__Create_new_author__').DoClick();
```

Since clicking on the Edit link will take you to a different page than where the ‘Create New Book’ link is available, in the example we have added a second instance of the:

```
SeS('Book_Management').DoClick();
```

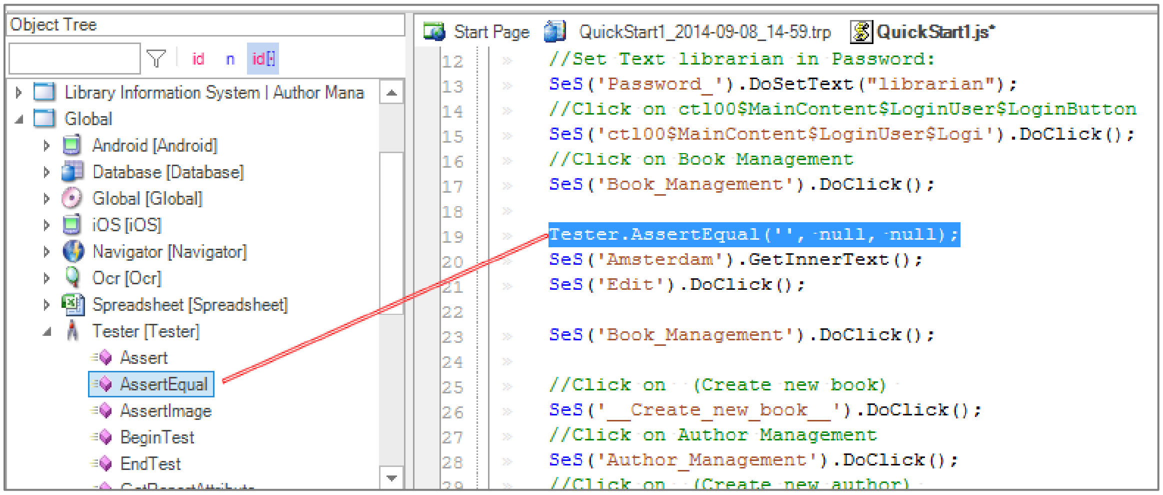
Command, so that Rapise goes back to the main book list page before executing the Create New Book step. If we did not do this, the test would have failed.

Now the line:

```
SeS('Amsterdam').GetInnerText();
```

by itself does not do anything, it just gets the text.

So to make the test more useful, we can use the global **Tester** object to add a step to verify this value. Drag the “AssertEqual” function from the Tester object to your test script just above the SeS(“Amsterdam”) line:



```
//Set Text librarian in Password:
SeS('Password_').DoSetText("librarian");
//Click on ctl00$MainContent$LoginUser$LoginButton
SeS('ctl00$MainContent$LoginUser$Logi').DoClick();
//Click on Book Management
SeS('Book_Management').DoClick();

Tester.AssertEqual('', null, null);
SeS('Amsterdam').GetInnerText();
SeS('Edit').DoClick();

SeS('Book_Management').DoClick();

//Click on  (Create new book)
SeS('__Create_new_book__').DoClick();
//Click on Author Management
SeS('Author_Management').DoClick();
//Click on  (Create new author)
```

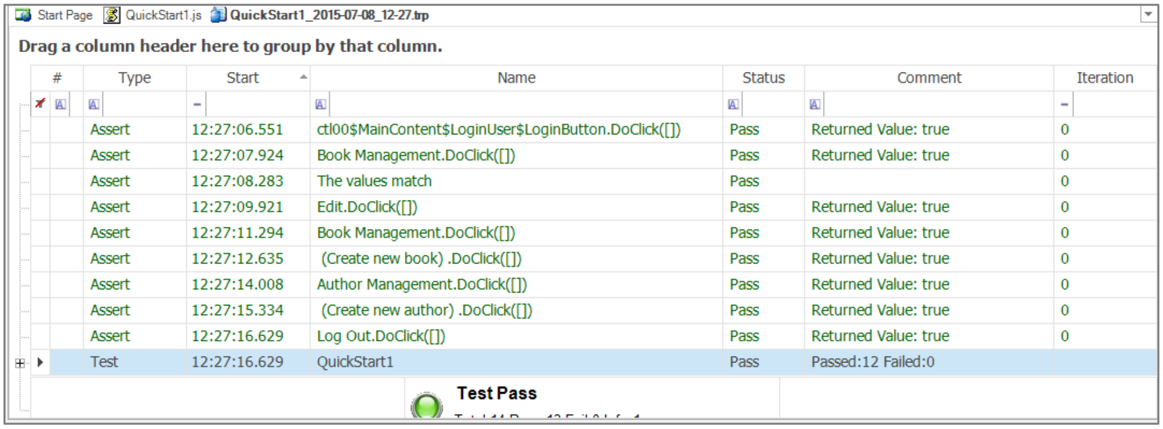
Now we need to just consolidate these two lines into the actual test. Using the script editor, change the two lines from:

```
        Tester.AssertEqual('', null, null);
        SeS('Amsterdam').GetInnerText();
```

To

```
        Tester.AssertEqual('The values match', 'Amsterdam',
SeS('Amsterdam').GetInnerText());
```

Now click **Play** to playback the new test:



Drag a column header here to group by that column.

#	Type	Start	Name	Status	Comment	Iteration
	Assert	12:27:06.551	ctl00$MainContent$LoginUser$LoginButton.DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:07.924	Book Management.DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:08.283	The values match	Pass		0
	Assert	12:27:09.921	Edit.DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:11.294	Book Management.DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:12.635	(Create new book) .DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:14.008	Author Management.DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:15.334	(Create new author) .DoClick([])	Pass	Returned Value: true	0
	Assert	12:27:16.629	Log Out.DoClick([])	Pass	Returned Value: true	0
	Test	12:27:16.629	QuickStart1	Pass	Passed:12 Failed:0	

Test Pass

Total 14 Pass 13 Fail 0 Info 1

The test should now pass successfully.

Legal Notices

This publication is provided as is without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

This publication could include technical inaccuracies or typographical errors. Changes are periodically added to the information contained herein; these changes will be incorporated in new editions of the publication. Inflectra Corporation may make improvements and/or changes in the product(s) and/or program(s) and/or service(s) described in this publication at any time.

The sections in this guide that discuss internet web security are provided as suggestions and guidelines. Internet security is constantly evolving field, and our suggestions are no substitute for an up-to-date understanding of the vulnerabilities inherent in deploying internet or web applications, and Inflectra cannot be held liable for any losses due to breaches of security, compromise of data or other cyber-attacks that may result from following our recommendations.

SpiraTest®, SpiraPlan®, SpiraTeam®, Rapise® and Inflectra® are either trademarks or registered trademarks of Inflectra Corporation in the United States of America and other countries. Microsoft®, Windows®, Explorer® and Microsoft Project® are registered trademarks of Microsoft Corporation. All other trademarks and product names are property of their respective holders.

Please send comments and questions to:

Technical Publications

Inflectra Corporation

8121 Georgia Ave, Suite 504

Silver Spring, MD 20910-4957

U.S.A.

support@inflectra.com