Web Based Automation Framework for Beginners

Scott Rodgers

Scott9Rodgers@hotmail.com

Abstract

Quality Assurance teams are overwhelmed with the amount of testing needed for a successful release. The answer is automated testing but easier said than done. This paper shows how to design and build an easy framework for Automation Testing using Freeware products that beginners can use to rapidly build automation for a web based product.

Upon being hired as a developer in QA to setup automated tests on a product that had been in production for over ten years with no real automated tests the amount of work far exceeded what one person could do. Other QA team members had no experience in writing automated tests. It was important to have a way to leverage their help in getting the work done.

By using TestNG, Selenium, Java, and a basic organizational style of programming to setup a framework other team members were able to quickly create automated tests. A single team member was able to create over 50 tests within 2 days to test navigation and verify the titles of the web pages. These tests used to take an hour to manually navigate the page links and are now are done in less than two minutes.

This class will instruct others on how to setup the framework and will help QA organizations with little or no automation in place to enable beginners to write tests successfully and quickly.

Biography

Since graduating from Cal State Fullerton in June of 1991 with a BS in Computer Science, Scott Rodgers has worked in the computer industry as Technical Support rep., API Support rep., Senior Developer, Build Master, QA engineer, and Automation engineer. He has also worked for a variety of different companies from printers, banks, start-ups, and Family History dept. of the LDS church. He has set-up several different frameworks for automation using Silktest, HttpUnit, and Selenium.

1 Automation Backlog

Let's presume that you have just been hired by this company that wants to start creating several suites of automation tests and you are the only developer on the QA team. How do you get through the Backlog of tests that need to be coded or written? The Backlog is all the test or test-cases that need to be run to meet the turnaround allotted by the Product Management team. With none of your team members having very much or any coding experience, it leaves you with a dilemma: how to meet this deadline?

One way to meet the deadline, is to have the some of the team members (beginners) write these automation tests. How this possible? With the proper framework and tools that are outlined in this paper, this is possible. Within a few days they will actually be writing fully functioning tests and reducing the Backlog.

At Simplifile we have found that these select tools, TestNG, Selenium, and Java along with the basic organization style of programming have provided the means for our QA team members to come up to speed and write tests that work properly in short order.

2 Freeware Trio

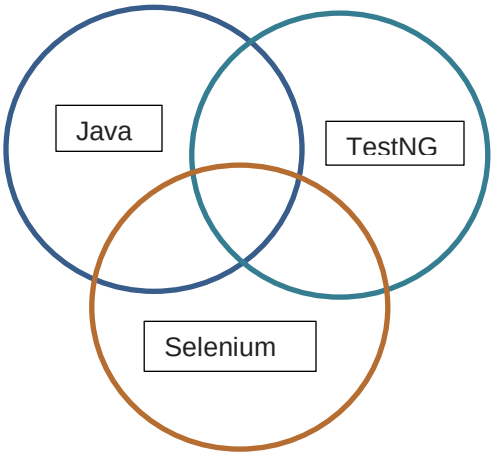
TestNG, Selenium, and **Java** (all freeware products) can work in concert together to create the website automation. Each product plays a specific role. There are many advantages to using this trio, since they work seamlessly together.

TestNG is used to run the whole suite of tests, groups of tests, or a single test. It provides the user great flexibility as to which tests or test that can be run at a given time. TestNG allows the user to create special methods that are run automatically upon start up and shut down of the test(s). TestNG provides the perfect structure to run automation tests. TestNG supports both Selenium Classes written in Java and thus Java itself.

Selenium is used to control the web browser as it accesses the website, which includes the navigating from page to page, entering data, checking for the presence of certain elements that should be contained on a page, and so many more things that should be tested for a given website to validate that it is ready for primetime.

Java is the language that the test code is written in. The tests are executed by Java and in conjunction with TestNG. It leverages the Selenium Classes to perform the necessary steps to complete the desired task being defined in the test cases for testing the website.

The trio provides the complete set of tools for beginners to write web automation that works in days, instead of months or years.



Free Ware Trio 1

This diagram shows the intersection of the trio in which they work in concert together to create the website automation. Let's look more in-depth at what makes each one a powerful part of doing web automation.

3 TestNG

TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers). [1]

At Simplifile we are using TestNG as a framework to run our web automation tests in conjunction with Selenium and Java. Here are the TestNG building blocks that we use.

3.1 Annotations

Annotations are used to denote a method for a given purpose. Annotations also support parameters that enhance how they are interrupted during the running of the test. The following Annotations are only a few of the ones available, but provide the needed functionality required to write effective tests.

3.1.1 Test

```
@Test (groups = { "Nav" })
```

Test is used to denote that a method as an actually test and not a regular method. If the Test Annotation includes the Groups parameter then that test can be executed as a part of that group. There can be single test or multiple tests for that Group. The Groups parameter are used to organize the tests in a given class or multiple classes.

3.1.2 Setup (Before…)

```
@BeforeGroups  (groups = {"LiveColumn1","LiveColumn2","LiveColumn3",”Nav”})
```

The BeforeClass, BeforeMethod, and BeforeGroup provide for different types of methods to be created for doing specific setup for tests.

3.1.2.1 The method annotated with “BeforeClass “will be invoked after the test class is built and before any test method that is defined within the Class is run. This allows those objects that need to be shared across other tests to be created once before any test are executed.

3.1.3 Teardown (After…)

```
@AfterGroups   (groups = {"LiveColumn1","LiveColumn2","LiveColumn3","Nav"})
```

The AfterClass, AfterMethod, and AfterGroup provide for different types of methods to be created for doing specific teardown for tests.

3.2 Asserts

There are many different types of Asserts to choose from when writing tests, as to which one you use will depend on what is to be tested. We have settled on using three of them, which are **assertTrue, assertEquals**, and **assertNotEqual**.

The **assertTrue** is used to verify that Web Elements exist, and for cases when something should contain a certain value. For example a text field should contain “joh” plus “nny” or “n” then the assertTrue will pass.

The **assertEquals** is used when the desired outcome is for the two objects are the same, such as the expected URL and the current URL.

The **assertNotEqual** is used when the desired outcome is for the two objects to not be the same.

How many asserts should be used in each test is based on what is defined in the test case for passing. The test should follow exactly the requirements that are defined in the test case. Since the Beginners aren’t that familiar with what makes a good automation test, this is the principle we follow at Simplifile.

3.3 Results

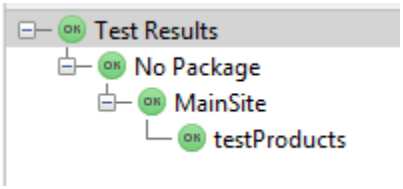
TestNG provides results for each tests run, as to whether it passed or failed. The overall result information will look like this:

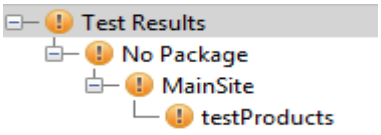
```
===============================================
```

Custom suite

Total tests run: 1, Failures: 0, Skips: 0

The Test Results information regarding the actual test run will look like this, green is for passing and orange for failing:





If the test fails there should be some type of output saying what the cause was of the failure. When using asserts it is a good practice to take the time to create the message that can be a part of the assert method. As shown below:

java.lang.AssertionError: https://simplifile.com/e-recording/ vs https://simplifile.com/sf/login/

Expected :true

Actual :false

There is always the possibility that either result could be a false positive depending how the code was written. It is good practice to have someone else do a peer review of both code and results until the beginners have demonstrated that they can determine the difference between a false positive and a real pass or failure.

Using **annotations, asserts**, and the **results as reports** that TestNG provides is the one of the major benefits provided by the Trio. The **annotations** provide the controls by which groups of tests can be run and what setup methods are called to allow the tests to function properly. **Asserts** provide the way to determine if the test passes or fails. The **results** provide the means to determine if a single test, a group of tests, or the suite of tests have passed or failed. The **results** can contain lots of clues for failing tests that will assist in resolving the issue, especially if the time was taken to add messages to the **asserts** in that test.

4 Selenium

Selenium automates browsers. That's it! What you do with that power is entirely up to you. Primarily, it is for automating web applications for testing purposes, but is certainly not limited to just that. Boring web-based administration tasks can (and should!) also be automated as well.[2]

Selenium WebDriver; a collection of language specific bindings to drive a browser -- the way it is meant to be driven.[2]

In other words Selenium WebDriver allows the Website or WebApp to be navigated within tests that incorporate it. The navigation is done very smoothly and in a logical way. There are many methods that are object based that leverage language specific bindings that the WebDriver provides. These are the ones that we use to assist our beginners in writing tests.

4.1 FindElement

```
WebElement we = driver.findElement(By.cssSelector("#logo > a > img"));
```

The FindElement is used to locate the First WebElement that matches the descriptor provided by the By.cssSelector(<WebElement Name>). This method returns a WebElement that can be acted upon, with actions like getText or Click.

String value = we.getText;

The String “value” now contains the text from that WebElement.

we.click();

That WebElement received that click and if the onClick method was available it ran the javascript that was a part of that method.

4.2 FindElements

```
if (driver.findElements(By.cssSelector("#menuDiv > iframe")).size()>0) {
```

The FindElements is used to locate a list of WebElements that matches the descriptor provided by the By.cssSelector(<WebElement Name>). This method returns a list of WebElements that can be acted upon. It can also be used to validate if a WebElement is present or not by checking the size of the list. To determine if the WebElement is present the size will be greater than zero.

4.3 Get

driver.get(HTTPS_SIMPLIFILE_COM_SF_LOGIN);

The get is used to load up the browser with the URL passed into the method. The driver is now using that URL to perform any commands requested through the driver methods.

4.4 GetCurrentUrl

String stateUrl = driver.getCurrentUrl();

The getCurrentUrl retrieves the URL for the current page. The driver is currently using that URL to perform any commands requested through the driver methods.

4.5 SendKeys

textWebElement.sendKeys("AutoMation");

The sendKeys will act like it is typing those characters. Text Field WebElements allow for the sendKeys method to be used to fill out forms.

These are the minimal five methods of the Selenium WebDriver that provide the capability to write tests to perform the test cases in a Step by Step fashion. They are used to find WebElements for validation purposes, navigate to or retrieve the URLs, and to provide the means replicate any action a Keyboard could do. This allows the test to perform the actions of a user of the website without actually doing it manually.

5 Java

Java is a programming language that is used by many businesses. It is used because it is very versatile and has a strong structure. Through its wide acceptance there are many examples that can be found on the internet to assist in writing test. Beginners can google the desired functionality they want and quite often find the java code they need. We use java because it is used by the Development team and it is rather easy to learn. It supports all the basic organization style of programing described in the next section.

6 Basic Organization Style of Programming

There are several ways to design the tests: by using Page Objects or straight forward Step By Step. Both have advantages and disadvantages, but for our purposes the Step By Step method is what will be discussed in this paper. It is really easy to create tests with this method. The tests can be quickly written without much thought as to how to organize them, since it follows the natural progression of a test case. Test cases are written step by step order. At a later time, they can be organized by using the Groups as a part of the Test Annotation. Whereas the Page Objects method requires a lot of planning as classes will need to be created for each web pages, in-order to be written correctly. Each class that is created for each page should contain methods to match each function that can be performed on the page. It is a lot of up front work. Since we are looking to produce test more rapidly this method doesn't offer us this ability. Plus there is a high learning curve to write classes for each page.

6.1 Constants

Declaring the most common URLS as a constant provides for easier validations and reuse of those URLS. It is also useful from the standpoint of doing updates, by having to update the URL in only one location.

6.2 Global and Local Variables

Declaring WebDrivers as global variables will allow them to be used by all the tests and can be set to the Local variable in a given test. The Local variables will get cleaned up upon the finishing of the method. The global variables will get cleaned up upon the finishing of the class.

6.3 Methods (Repeatable Code)

Taking code that is copied from test to test is a candidate for becoming a new method which in turn can be called by all those tests that contain that copied code. This is very basic coding principle that makes code more robust. It sometimes takes some re-engineering to create variables that match the different

WebElements that are needed in the newly created method. The re-engineering doesn't need to happen right away. It is important to let the beginners create lots of tests at first to get their feet wet so to speak. By doing so, they will be able to see on a grander scale the reasoning for the creation of methods that consolidate code. It may take time before the beginners can recognize what needs to be a candidate of this process. It is worth the effort to assist them in this process of determining when to construct a new method that replaces the copied code.

6.4 Logic Constructs and Data Types

Logic Constructs and Data Types should be leveraged always when writing code for the simple fact that they are very powerful. The power comes in that the logic constructs provide means to traverse the data objects, perform the same section of code multiple times, perform a certain action based on a value provided, and many more.

The logic constructs could be as simple as a For Loop for an object, For Loop by count, While, Do-While, If, If-Else and many more. These examples can give great structure to the writing of the tests, which ones that will be used should be represented in the test case in the Step by Step method as it defines what needs to be verified.

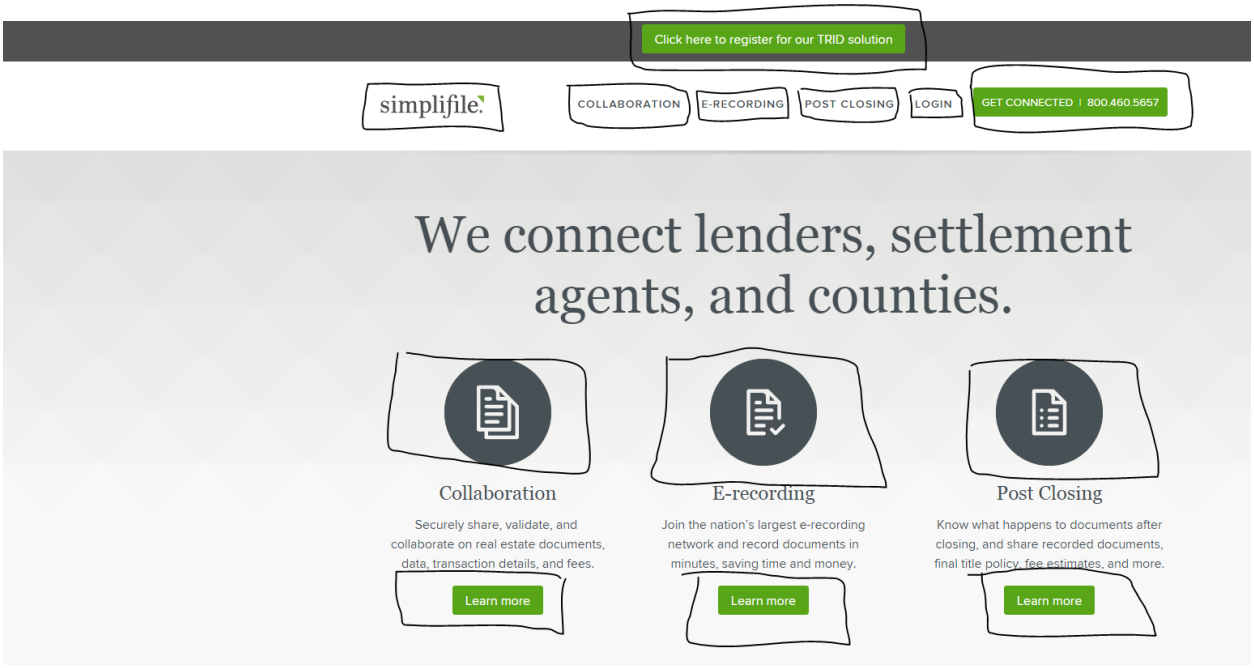
The Data Types could be as simple as a Hashmap, a List, or an Array to store data for comparison or that needs to be pasted from method to method. The use of data types is a good coding practice and should be implemented whenever possible. It provides the writer of the code the means to evaluate the data being used in the test, which is invaluable when trying to determine why a test is failing.

This basic organization style of programming lends itself to beginners in assisting them with concert concepts to follow while writing automation tests. By creating a template class that contains tests in the sample in the next section our beginners have a guide to writing test within days of starting.

7 Sample

This section of the paper will further show the code that actually tests the Simplifile.com Main Page, along with an image of the page.

7.1 Main Web Site Page Layout



Black triangles are to show the links that need to be present for the test to pass and then tested to verify that they work in another test.

7.2 Test Code for If Links are Present

```
@Test    (priority = 4,groups = { "Nav","SAMPLE" })

public void testMainPageLinksRPresent(){

    //validate that links are present

    assertTrue(driver.findElements(By.cssSelector("#post-27 > section > div.home-banner > div > div > div > div.home-banner-collaboration.fourcol.first > div.home-banner-person > a > img")).size() > 0);

    assertTrue(driver.findElements(By.cssSelector("#post-27 > section > div.home-banner > div > div > div > div.home-banner-erecording.fourcol > div.home-banner-person > a > img")).size()>0);

   // There are more links that are being tested in this method, shown by the yellow highlighted links, but these two lines are enough to get the idea across.

}
```

This is the best way to see if the links are present on the page that is outlined by the test case and then report the result. The actual testing for the link is done by the checking the size returned by the findElements for that WebElement Link. In order to return true, the size must be 1 or greater. There

should only be one link found by that selector, otherwise there could be duplicate links on that page. The assertTrue is used to report whether the link was found or not.

We chose to tests all the links of this page at once, instead of creating 12 additional tests that just do one link at a time. It was a savings on the number of tests that needed to be written and executed, with little or no cost to debug any issues with missing links.

7.3 Test Code for If Link Redirects Work

```
@Test    (groups = { "Nav" })

public void testProducts(){

    //validate that links go to correct location

    goToPage("#menu-item-34 > a",HTTPS_SIMPLIFILE_COM_E_RECORDING);

    goToPage("#menu-item-135365 > a",HTTPS_SIMPLIFILE_COM_COLLABORATION);

    goToPage("#menu-item-135364 > a",HTTPS_SIMPLIFILE_COM_POST_CLOSING);

}
```

Calls method goToPage() for each link and passes in the desired redirected page by URL. This method performs the same test like “If Links are Present”, but actually clicks on the link and then verifies that the new URL matches what the newly re-directed URL Then reports if all these checks passed.

```
private void goToPage(String link, String newURL) {

    assertTrue(driver.findElements(By.cssSelector(link)).size()>0);

    driver.findElement(By.cssSelector(link)).click();

    assertEquals(driver.getCurrentUrl(),newURL,driver.getCurrentUrl()+" vs. "+newURL);

    home();

}
```

Calls method home(), which will always redirect to the Home Page by URL. This method contains a special case check for a page that doesn’t follow the rest of the website format style and has no Home link in order to navigate back to the Home Page.

```
private void home(){

    if(!driver.getCurrentUrl().equals(HTTPS_SIMPLIFILE_COM_SF_LOGIN)) {

        driver.findElement(By.cssSelector("#logo > a > img")).click();

    }else if (!driver.getCurrentUrl().equals(HTTPS_SIMPLIFILE_COM)) {
```

```
        driver.get(HTTPS_SIMPLIFILE_COM);
    }
}
```

These are just two samples of test code that have been written to test links on the home page for Simplifile.com. These tests embody the Freeware Trio along with Basic Organization Style of Programming which have been outlined up to this point. Each item outline from 2 through 6 has been addressed in these two tests at least once. By using these two test as a pattern in our template class, our beginners can write their own tests very quickly.

8 Summary

The Freeware Trio are widely accepted in the Testing Community as being standard tools. They all work well together in building a highly functioning Automation Testing framework as outlined in this paper. There is no financial obligation in adopting these as a part the testing arsenal. There is a low learning curve. Through leveraging TestNG, Selenium, Java, and basic organization style of programming the most novice QA team member can use a Java Class Template as a pattern to write automation tests within days. Taking a mostly manual QA team to a new productivity level through writing automation using Freeware is now possible by using these practices that were outlined in this paper.

9 References

[1] TestNG, Cédric Beust (cedric at beust.com) “Introduction” Current version: 6.9.4, Created: April 27th, 2004, Last Modified: May 9th, 2015 http://testng.org/doc/documentation-main.html#introduction (accessed June 9,, 2015)

[2] Selenium, http://www.seleniumhq.org/ (accessed June 9,, 2015)