# CiviCRM Design Note – Regression Tests

Tony Maynard-Smith 15/10/2020

This note gives an outline of the automated regression tests which have been implemented using Rspec and Watir. The motivation for setting this up was the occurrence of several regression errors, both in upgrading CiviCRM software, and in making changes to the LALG configuration.

## References

The following were found useful in installing the necessary software, and understanding some of the basic elements of the libraries used:

<http://watir.com/guides/>

<https://www.tutorialspoint.com/rspec/rspec_introduction.htm>

<http://testerstories.com/2011/10/automated-testing-with-watir-and-rspec-part-1/>

<http://testerstories.com/2011/10/automated-testing-with-watir-and-rspec-part-2/>

<https://www.tutorialspoint.com/watir/index.htm>

<https://www.rubyguides.com/2018/07/rspec-tutorial/>

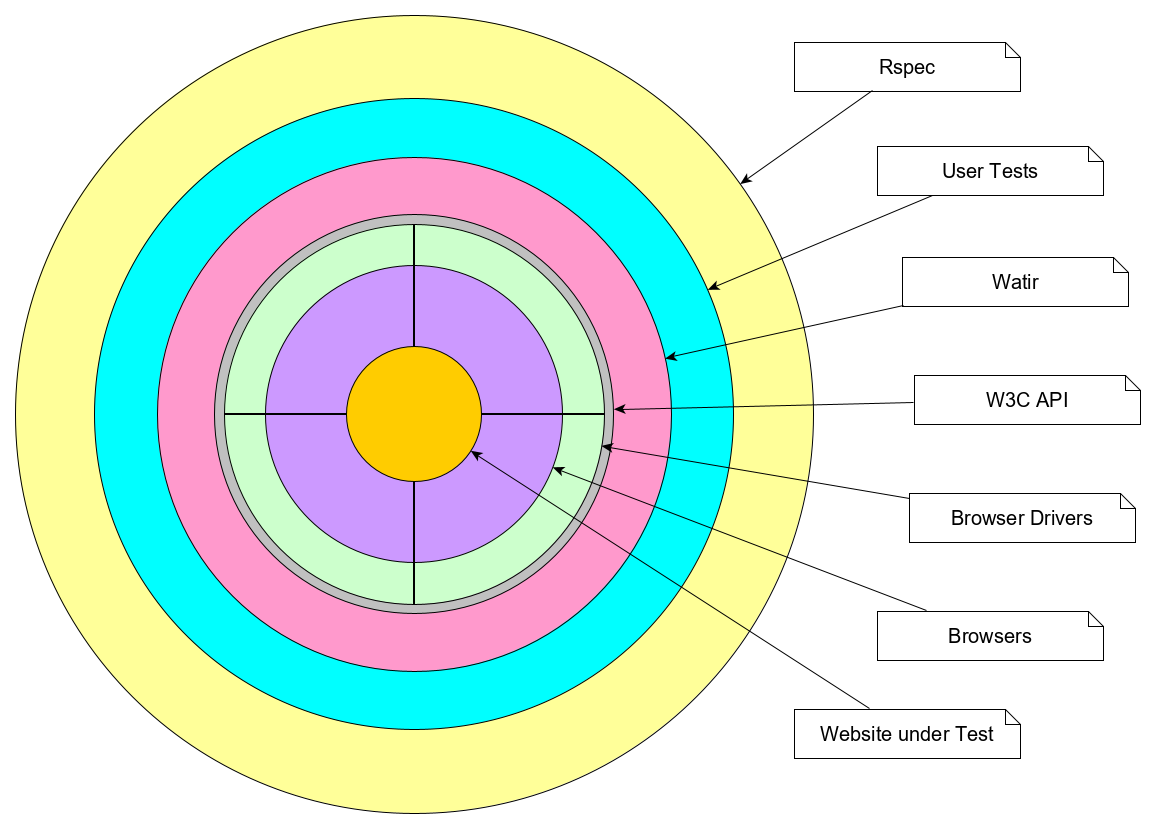
<https://ruby-doc.com/docs/ProgrammingRuby/>

<https://relishapp.com/rspec/rspec-core/v/3-9/docs>

## Architecture

The test suite resides on a client PC and executes tests on the website using an API built into common browsers to enable such automation. The tests use the Ruby language, and two Ruby libraries:

* Watir: Abstracts the browser API and adds many useful features, such as locating elements and waiting for elements to become available.
* Rspec: Structures a set of tests, provides facilities for checking various conditions, and running selected tests.



This very schematic diagram shows:

* Overall control of running the test processes is exercised by Rspec.
* The test processes are defined by the user written test specifications. These are written in Ruby using elements from the Rspec and Watir libraries (also written in Ruby).
* All communication between the User Test Specifications and the target website are through a stack consisting of:
  + The Watir library functions.
  + A standard API defined by the W3C
  + A Browser Driver which maps the common W3C interface onto the API provided by a particular browser.
  + A standard Browser. Chrome, Firefox, Edge and Safari are documented as supported, but so far I have only found the driver for Chrome.
* The LALG website under test.

## Installation

You need to install Ruby, libraries (Gems) and drivers for the relevant browsers on your PC. See the first reference document above.

[AMS. On my PC they are in c:\Ruby26-x64.]

## Test Implementation

At the time of writing, tests are concentrated on the CiviCRM Membership system, with the objective of providing a level of confidence against regression faults when updating the CiviCRM software or the LALG configuration, such as webforms.

Tests can be broadly divided into the following groups:

* Utility Functions
  + Test Admin Login
  + Test Create User (i.e. End User)
  + Test User Login (i.e. End User)
  + Test Clean Data
  + …
* Administrator Operations
  + Admin New Member
  + Admin New Member NoEmail
  + Admin Renew Membership
  + …
* End User Operations
  + User New Member Stripe
  + User New Member Pay Later
  + …

## Test Operation

Tests are run from a Windows Command Line (cmd) window, and it is assumed that the standard software has been installed with the Path environment variable set up so that ruby and rspec may be invoked directly from a command prompt.

Tests are (currently) contained in a single file named CiviTestVn.rb; the first step being to cd to the directory containing this file. (Currently on my PC,   
 C:\Users\Tony\Dropbox\LALG Website development\Watir Regression Tests

Point the script at the correct website with the command: set RspecDomain=xxx  
Where xxx = www, dev or tmp as appropriate (defaults to dev if not set).

Run one or more tests with the command:   
 rspec CiviTest-Tests.rb -e 'Test Name' -f html -o 'results.html'

-o directs the test results to the specified file, which may be altered as required.

-e specifies the test(s) to be executed. The quoted phrase is searched for in the title of all tests, with those matching selected. So that for example:

‘Pay Later’ will run the one test specified.

‘New Member’ will run all the New Member tests.

Omitted entirely will run all the tests.

If -e is repeated two or more specific tests may be selected.

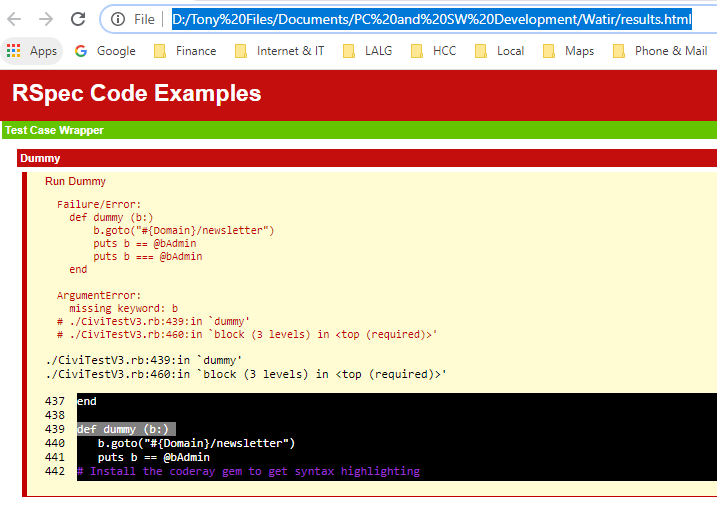
**NOTES:**

* The website must be set up with a Drupal User having Membership Administrator privilege and:  
   Username: watir  
   Password WatirTesting987  
   First Name Joe  
   Last Name WatirAdmin  
   Postcode JW1 1JW
* To allow an automated login you need to turn off CAPTCHA on the user\_login and user\_register forms (Drupal Configuration >> People >> CAPTCHA menu).   
  And turn on again afterwards if appropriate.
* To include the User Tests, which use STRIPE, the patch PR373 for Webform\_Civicrm must be installed. This allows Payment Test mode using argument ‘?payment=test’, so the live operation is not (should not be) interfered with.
* Most tests clear the Print Cards flag on the test Contacts, but again should leave any flags on live contacts alone. But be aware.
* Test data from the last test to be executed is left in the system and not cleaned out. This is deliberate to assist diagnosis of possible failures.

Following completion of a test run the results may be displayed by opening the HTML file quoted in the initial command line. A successful result looks like the following:



An error report looks something like this – though this an artificial example.



## Test Structure

The Test Specification source file is divided into two files containing the following main sections:

### Code File

* Global Constants.  
  Setting up a few constants, for the website Domain and other conveniences. Could be extended to include which browser to use (when more than one demonstrated).
* Utility Setup Methods  
  Including Opening the Browser, Logging in, Logout, Cleaning old test data, etc.
* Test Action Methods  
  Routines to perform the different parts of the business processes being tested. Including Create a New Member, Make Payment, Renew Membership, etc.
* Test Verification Methods  
  Routines to carry out the verification of the system state following the Action part of a test. Checks for Contacts, Relationships, Memberships, Contributions etc.

### Tests File

* Test Specifications  
  The actual definition of the various tests listed above. Typically, will call Utility Setup Methods to prepare the environment, one or more Action Methods to perform the required process, and one or more Verification Methods to check the results.