Team 3's Automatic Testing Framework for OpenMRS CSCI 362

Meg Krawczyk, Mitch Suzara, Itzayana Carrillo De Los Santos, Jim Bowring

Introduction

As part, our Software Engineering class students are required to pick an open-source F/HOSS project and create an automatic testing software to test 5 methods from the project, each with five test cases. Choosing a project to test was difficult for our team, we looked through, downloaded, and attempted to compile seven to eight projects before finally settling on OpenMRS.

OpenMRS is a program that provides an electronic medical record system and its mission is to improve healthcare in areas of the world that need it the most. In addition to being a great cause, OpenMrs has been kept up to date and had great documentation which is why we decided to choose it.

Methods

Creating Test Cases and Drivers

Creating the test cases was a simple process, each test case consisted of an ID, method name, class name, requirements, input, expected output, and the driver name. The diver was also a simple process, we simply created a file that contained the main method, took the input, and called the function we wanted to test. Then the script would take the output from the driver and compare it to the expected output from the test case.

Creating Automated Script

Once we had our drives and test cases we needed to create a bash script that would automatically compile and run the drivers. Once all the drivers were done running, the script would then open a browser tab and display the results of each test in a table, seen in figure 2. We have a graph of our framework in figure 1 that demonstrates the flow of our framework.

Implementing Errors

We implemented a few errors in our program and see whether our automated testing framework would catch these errors. The way we decided to implement these errors was through comments as described above. In each method, there is a comment that contains a snippet of code with an error in it. To run the code with the error in it simply uncomment that code segment and comment out the segment above it.

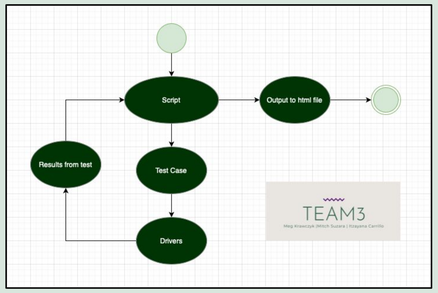

Figure 1: This is a diagram of our testing framework. The script uses the test case files to access the appropriate driver, then the driver runs the test. The script then takes those results and outputs them to a HTML file.

Purpose

The purpose of this project was to get hands-on experience working on an active open source project. At the end of this project, our team should have learned valuable skills such as bash scripting, creating test cases, creating drivers, and a better overall understanding of how creating software works. In addition, this project facilitated self-driven learning, a skill valuable in the software engineering industry.

Results

Team 3 | Carrillo, Krawczyk, Suzara

Test Results

Test ID	Class Name	Summary	Method Type	Inputs	Expected Outputs	Driver	Result	Pass/Fail
01	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1969/12/31/1/1/1	Wed Dec 31 23:59:59 EST 1969	getLastMomentOfDayDriver	Wed Dec 31 23:59:59 EST 1969	Passed
02	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1800/9/7/19/21/0	Sun Sep 07 23:59:59 EST 1800	getLastMomentOfDayDriver	Sun Sep 07 23:59:59 EST 1800	Passed
03	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1969/12/31/4/4/4	Wed Dec 31 23:59:59 EST 1969	getLastMomentOfDayDriver	Wed Dec 31 23:59:59 EST 1969	Passed
04	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	2001/9/8/6/56/0	Sat Sep 08 23:59:59 EDT 2001	getLastMomentOfDayDriver	Sat Sep 08 23:59:59 EDT 2001	Passed
05	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1998/12/20/20/45/30	Sun Dec 20 23:59:59 EST 1998	getLastMomentOfDayDriver	Sun Dec 20 23:59:59 EST 1998	Passed
06	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/31/19/20/34	Wed Dec 31 19:20:34 EST 1969	DateUtilDriver	Wed Dec 31 19:20:34 EST 1969	Passed
07	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	3930/38/10/28/13 /14/15	Sun Oct 28 13:14:15 EDT 393038	DateUtilDriver	Sun Oct 28 13:14:15 EDT 393038	Passed
08	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/31/19/00/00	Wed Dec 31 19:00:00 EST 1969	DateUtilDriver	Wed Dec 31 19:00:00 EST 1969	Passed
09	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/21/23/37/30	Sun Dec 21 23:37:30 EST 1969	DateUtilDriver	Sun Dec 21 23:37:30 EST 1969	Passed
10	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1800/11/30/12/01/36	Sun Nov 30 12:01:36 EST 1800	DateUtilDriver	Sun Nov 30 12:01:36 EST 1800	Passed
11	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	808760	true	containsOnlyDigitsDriver	true	Passed
12	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	-842	false	containsOnlyDigitsDriver	false	Passed
13	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	8940	true	containsOnlyDigitsDriver	true	Passed
14	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	34hi80	false	containsOnlyDigitsDriver	false	Passed
15	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	60 78	false	containsOnlyDigitsDriver	false	Passed
16	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	600	600	convertToIntegerDriver	600	Passed
17	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	6000000000	null	convertToIntegerDriver	null	Passed
18	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	-8387852740 2	null	convertToIntegerDriver	null	Passed
19	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	2147483647	2147483647	convertToIntegerDriver	2147483647	Passed
20	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	-2147483648	-2147483648	convertToIntegerDriver	-2147483648	Passed
21	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	HELLO THERE	false	containsUpperAndLowerCaseDriver	false	Passed
22	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	how are you today	false	containsUpperAndLowerCaseDriver	false	Passed
23	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	I'm well how are YOU?	true	containsUpperAndLowerCaseDriver	true	Passed
24	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	Hi12	true	containsUpperAndLowerCaseDriver	true	Passed
25	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	hejfbjsd83eh	false	containsUpperAndLowerCaseDriver	false	Passed

Updated on 12/01/2020 03:56:11 PM EST by suzaram

Figure 2: The chart above contains the results of the testing framework without errors implemented, All tests pass. In this chart, but on one of our team member's machines test case 9 and 10 fail due to being one second behind

Test ID	Class Name	Summary	Method Type	Inputs	Expected Outputs	Driver	Result	Pass/Fail
01	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1969/12/31/1/1/1	Wed Dec 31 23:59:59 EST 1969	getLastMomentOfDayDriver	Wed Dec 31 23:59:58 EST 1969	Failed
02	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1800/9/7/19/21/0	Sun Sep 07 23:59:59 EST 1800	getLastMomentOfDayDriver	Sun Sep 07 23:59:58 EST 1800	Failed
03	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1969/12/31/4/4/4	Wed Dec 31 23:59:59 EST 1969	getLastMomentOfDayDriver	Wed Dec 31 23:59:58 EST 1969	Failed
04	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	2001/9/8/6/56/0	Sat Sep 08 23:59:59 EDT 2001	getLastMomentOfDayDriver	Sat Sep 08 23:59:58 EDT 2001	Failed
05	OpenmrsUtil	This method will take a date and time, then shift the time to be the last second of the day on that date.	getLastMomentOfDay	1998/12/20/20 /45/30	Sun Dec 20 23:59:59 EST 1998	getLastMomentOfDayDriver	Sun Dec 20 23:59:58 EST 1998	Failed
06	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/31/19 /20/34	Wed Dec 31 19:20:34 EST 1969	DateUtilDriver	Wed Dec 31 19:00:00 EST 1969	Failed
07	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	3930/38/10/28/13 /14/15	Sun Oct 28 13:14:15 EDT 393038	DateUtilDriver	Sun Oct 28 13:00:00 EDT 393038	Failed
08	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/31/19 /00/00	Wed Dec 31 19:00:00 EST 1969	DateUtilDriver	Wed Dec 31 19:00:00 EST 1969	Passed
09	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1969/12/21/23 /37/30	Sun Dec 21 23:37:30 EST 1969	DateUtilDriver	Mon Dec 22 23:00:00 EST 1969	Failed
10	DateUtil	This method will return a Date and time with the milliseconds truncated	truncateToSeconds	1800/11/30/12 /01/36	Sun Nov 30 12:01:36 EST 1800	DateUtilDriver	Sun Nov 30 13:00:00 EST 1800	Failed
11	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	808760	true	containsOnlyDigitsDriver	false	Failed
12	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	-842	false	containsOnlyDigitsDriver	false	Passed
13	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	8940	true	containsOnlyDigitsDriver	false	Failed
14	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	34hi80	false	containsOnlyDigitsDriver	false	Passed
15	OpenmrsUtil	This method returns true if a string only contains digits	containsOnlyDigits	60 78	false	containsOnlyDigitsDriver	false	Passed
16	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	600	600	convertToIntegerDriver	null	Failed
17	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	6000000000	null	convertToIntegerDriver	null	Passed
18	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	-8387852740 2	null	convertToIntegerDriver	null	Passed
19	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	2147483647	2147483647	convertToIntegerDriver	null	Failed
20	OpenmrsUtil	This method turns a long into an integer, as long as it does not exceed the int memory limit	convertToInteger	-2147483648	-2147483648	convertToIntegerDriver	null	Failed
21	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	HELLO THERE	false	containsUpperAndLowerCaseDriver	false	Passed
22	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	how are you today	false	containsUpperAndLowerCaseDriver	true	Failed
23	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	I'm well how are YOU?	true	containsUpperAndLowerCaseDriver	true	Passed
24	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	Hi12	true	containsUpperAndLowerCaseDriver	true	Passed
25	OpenmrsUtil	This method returns true if a string has upper and lower case letters	containsUpperAndLowerCase	hejfbjsd83eh	false	containsUpperAndLowerCaseDriver	true	Failed

Figure 3: The chart above contains the results of the testing framework with errors in the code. The point of this chart is to see if our tests would catch the error.

Analysis

Basic Testing

When you run our automated testing framework a firefox browser should open up and display these results. As you can see in figure 2 all the tests passed. However, one weird quirk we found was that with certain dates Mitch's actual output would be one second before the expected output, causing it to fail but on the other team member's computers, the test passed without any issue.

Testing with Errors

About four out of five of our tests failed when an error was implemented into the code being tested, as you can see in figure 3. We found that when an error is hardcoded more likely than not all cases will fail, like in tests 1-5. While in many of the other cases the error was dependant on the input given, so sometimes the error could affect it and cause it to fail, while other inputs would be unaffected by the error, and as a result, those tests would still pass.

Overall Experience

Overall this group project was a positive learning experience. The team collectively learned a great deal about the importance of testing. Taking the time to test code while it is being developed saves time, resources, and headaches in the long run. We learned about one of the most powerful tools in the computing world; the command line. Having a graphical interface is convenient for visualization purposes, but the power that comes with the command line is far superior. In the end, we enjoyed the semester-long project and look forward to learning more in the continuation of CSCI 462.

Acknowledgments

Special thanks to OpenMrs for opening your project to the public.

Special thanks to Jim Bowring for leading us through this project this semester.

Special thanks to the College of Charleston Department of Computer Science for providing the opportunity for us to work on a challenging and rewarding project