Assignment HW5

Cover Page

Prepared for:
Dr. Mehra Borazjany
Trung Hieu Tran

Prepared by:
Alex Lundin
SE-4367.0U1-Testing

Assignment Choice:
N/A for this assignment

24 June, 2018

Proof of Working Software

GitHub link:

https://github.com/AlexLundinEducational/SE-4367-Testing

Branch Summary:

master – managed by Alex, only fully completed pulls allowed to make TA's life easy. Master only contains assignment material once they reach completed status.

working – flexible branch for team, ideally, this material should build without causing technical debt during the project.

Commit for grading:

HW5_Alex-Lundin Complete, Ready for Merge to master and Ready for Grading

Test Set

Data driven testing attempts to reduce the amount of hard coding into each test case in a set. So, the end goal, is to create a scenario where a tester can send a wide range of inputs through a fixture while observing the outputs. Data driven is much more focused on method arguments than other forms of testing.

I chose to use a parameterized test set, to achieve modular data driven test cases. The Junit package comes with a @Param keyword which allows the tester to define parameters to send into a test method. My design uses an iterable 2-dimensional array list with arguments as follows: month1, day2, month2, day2, year, expectedReturn. The second piece of my design is public Constructor. This testCalc constructor does all the instantiating that my third pieces require. The third piece is the AssertFixture. This method is the key to the whole design.

Here's an overview of the sequence of events. When the test starts, the method with @Params before it is called. In this case, that sets up the parameters list. Next, the constructor, piece 2, is called. The constructor creates an object to hold the test results and other setup tasks. The constructor also accepts the parameter values from the current index of the array list. Finally, the AssertFixture is called which actually tests the cal method and evaluates the results. Steps 2 and 3 repeat until the list is done.

For my test set, I focused on normal behaviors, meaning valid inputs based on the preconditions. First, I tried a few values where day1 and day2 crossed another as well as month1 and month2, those were test cases 1-4. Test cases 5-7 I tested the boundary values for all inputs. Test cases 8-10 I chose some randomized inputs to see how the method performed for a random set.

Two Faults

A – Impossible to find

This if statement added into the cal() method could never be found by any of my tests. This is mainly because the preconditions I used as assumptions limit my test set.

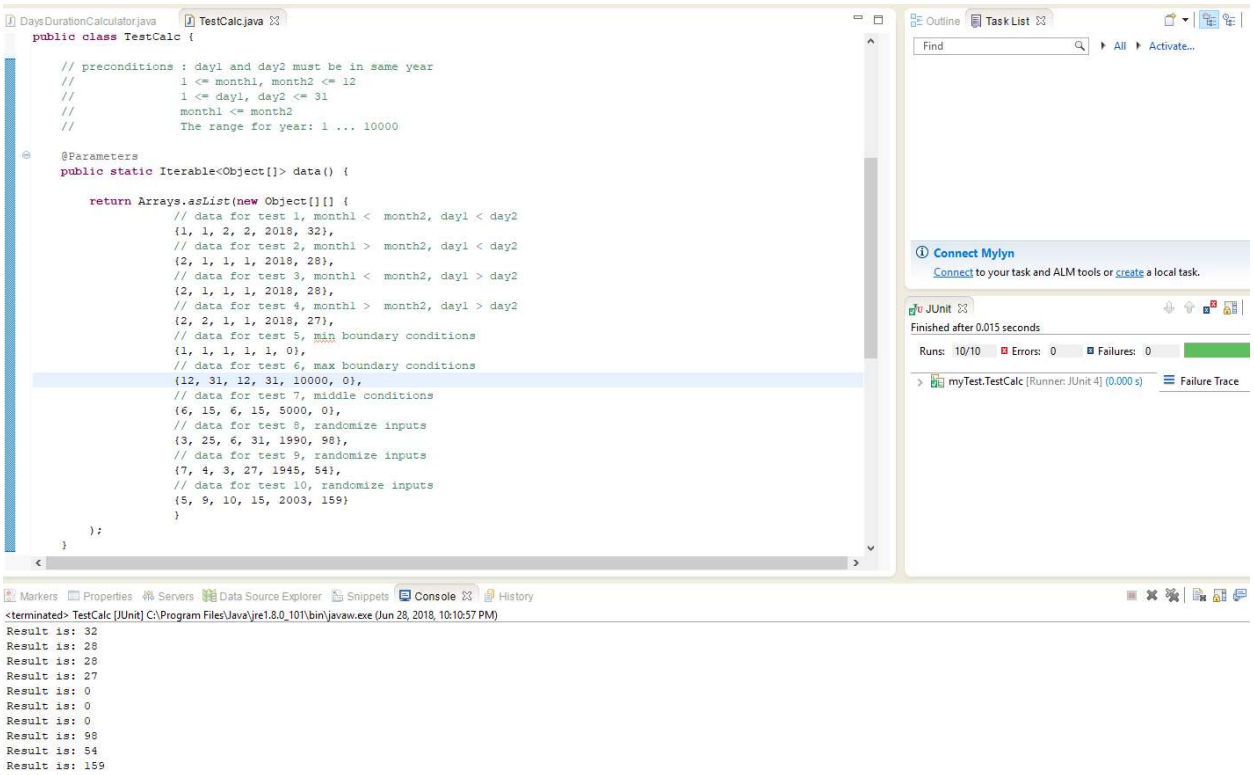
```
if(year > 10000){
	return -1;
}
```

B – Possible to find

This if statement added into the cal() method would be found by my tests. This is mainly because I did boundary checking on the assumption edges.

```
if(year == 10000){
	return -1;
}
```

Screenshot


DaysDurationCalculator.java
TestCalc.java
public class TestCalc {
// preconditions : day1 and day2 must be in same year
// 1 <= month1, month2 <= 12
// 1 <= day1, day2 <= 31
// month1 <= month2
// The range for year: 1 ... 10000
@Parameters
public static Iterable<Object[]> data() {
return Arrays.asList(new Object[][] {
// data for test 1, month1 < month2, day1 < day2
{1, 1, 2, 2, 2018, 32},
// data for test 2, month1 > month2, day1 < day2
{2, 1, 1, 1, 2018, 28},
// data for test 3, month1 < month2, day1 > day2
{2, 1, 1, 1, 2018, 28},
// data for test 4, month1 > month2, day1 > day2
{2, 2, 1, 1, 2018, 27},
// data for test 5, min boundary conditions
{1, 1, 1, 1, 1, 0},
// data for test 6, max boundary conditions
{12, 31, 12, 31, 10000, 0},
// data for test 7, middle conditions
{6, 15, 6, 15, 5000, 0},
// data for test 8, randomize inputs
{3, 25, 6, 31, 1990, 98},
// data for test 9, randomize inputs
{7, 4, 3, 27, 1945, 54},
// data for test 10, randomize inputs
{5, 9, 10, 15, 2003, 159}
}
);
}
Outline
Task List
Find
All
Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
JUnit
Finished after 0.015 seconds
Runs: 10/10
Errors: 0
Failures: 0
myTest.TestCalc [Runner: JUnit 4] (0.000 s)
Failure Trace
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
History
<terminated> TestCalc [JUnit] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (Jun 28, 2018, 10:10:57 PM)
Result is: 32
Result is: 28
Result is: 28
Result is: 27
Result is: 0
Result is: 0
Result is: 0
Result is: 98
Result is: 54
Result is: 159