Renaldo Musto

April 17, 2022

CS-320

**Project Two**

The mobile app that I developed for my customer consisted of three main services. We had a contact, task and appointment service. All three services had similar requirements mainly involving adding, removing and editing their respective components. Because of this I took effectively the same approach to testing the functionality for all three services. First I would test the constructor to make sure that the program could actually create a contact, task or appointment with the proper parameters. I would use assertTrue statements to make sure that the data I passed to the constructor actually got assigned to a variable. Then I needed to test the setters and getters of the program to make sure that the program could not only change existing data but retrieve data as necessary. From there I created a specific unit test for each requirement outlined by the customer. For example, in the appointment service I needed to ensure that the details of the appointment remained under 50 characters so I had an assertions test run a string longer than 50 characters to make sure that an illegal argument exception was thrown by the program. I think that my tests were fairly effective as I got a relatively decent coverage percentage. While I definitely could have done better in the service classes I still kept the average coverage between 60% and 70% and in the constructor classes I always hit above 90% usually close to 100% coverage. This basically tells me that my program should be perfectly fine when it comes to constructing the contacts, tasks and appointments but there could potentially be some issues when it comes to updating them even though I think my coverage percentage makes this unlikely.

One of the main ways that I made sure my code was technically sound was to separate all the different functionality of my code into separate methods that way they could all be tested individually and a problem with one aspect of the program shouldn’t affect any other parts. For example in the Task Service class I have addTask on line 19, deleteTask on line 32 and updateTask on line 51. Setting things up this way I can have all three aspects of the functionality of TaskService function independently of one another and therefore test and troubleshoot them independently. In code I think that it is always best practice to have one function do as little as possible and have it focus on one task only and that is what I stuck to here. I made my code efficient by using as few lines and functions as possible to achieve my goals. For example all of the functionality required to add a contact would be in the addContact method on line 19 of the ContactService class with the small exception of the setters but that is unavoidable. This way my program doesn't have to run separate processes to achieve one simple goal. I also used proper naming conventions and commented my code thoroughly so that I could easily identify what does what later on if I or someone else were to ever need to make changes to the program.

For this project I simply needed standard assertions tests to make sure that my code met the functionality required by the customer. This testing method essentially works by passing some information to whatever method or constructor you are testing that you know should cause a specific behavior and seeing if that actually happens. For example if you wanted to make sure that a constructor actually worked you would create assertTrue statements that checked to see if it was true that the variable value equaled what you sent it to be set as. Another use case for this standard type of assertions testing would be something like a database. You could use tests like these to make sure that a database actually responded to changes as expected. Other testing methods that I chose not to employ for this project would be dynamic and parameter testing. Dynamic testing works by making something called a TestFactory where you would run multiple tests simultaneously that all ran after the code was already initialized. This could be useful for testing something that needs to be tested after a program has already launched like its connection to the internet or other users as well as maybe the program's ability to interact with and update a database. Parameterized testing works by reusing the logic of a test through multiple inputs. For example I could have a test that was checking the input on a variable from a user where it had a character limit or some other type of constraint and run many different inputs through it to make sure that no matter what I threw at the program it was able to understand. This might come in handy to make sure that a program can handle numbers, characters and symbols all equally well and in any combination especially for something like password creation where that would be very important.

In order to stay cautious while developing my code I made sure to run it regularly while I was programming it so that if something went wrong I would be able to tell exactly what line or lines caused the error. I also made sure not to overcomplicate things and add functionality that wasn’t required of me. I also made sure to be clear in my variable naming so that by the time I was done working with four different files per service everything would be consistent across the program and make sense. I think it makes perfect sense to assume that I was a bit biased in testing my own code. I knew exactly how the code worked so I suppose I might have only tested for what I expected to happen. It would probably make more sense to have someone who had never seen my code before write tests for it because they would think of things that I did not. In a small project like this I don’t think it ended up being too much of a problem but I can imagine on an enterprise level you would want the developers and testers to be two separate teams so that bias can be mostly eliminated. Being disciplined is extremely important when it comes to being a professional in the field of software development. In the actual workforce I will almost always be working with a team so it is imperative that I discipline myself and meet my deadlines so that I don’t drag my team behind. It is also important to stick to the standard that your company is working with. Usually every company has a slightly different naming convention or coding practice that I will have to learn and it will be important to remain consistent with that company's methods across all my code so that my colleagues can work with my code easily and efficiently.

Reference

García, B. (2017). *Mastering Software Testing with JUnit 5*. Packt Publishing.