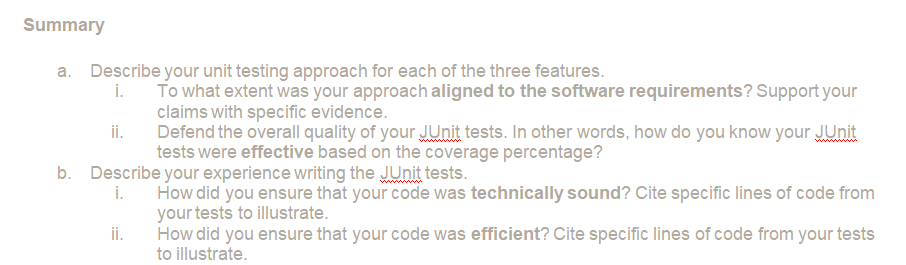
**CS 320**

**Project Two**

**Dr. Bryant Moscon**

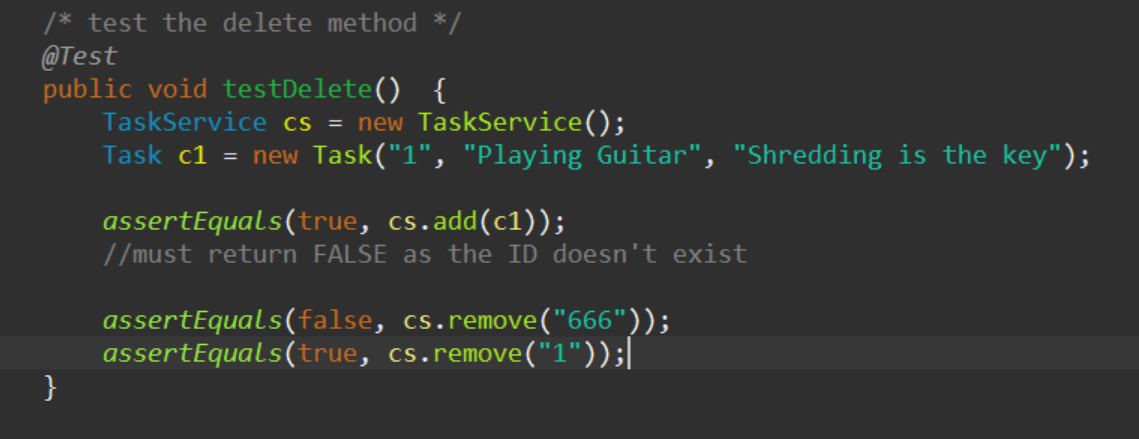
Taro Serigano

12/12/2020



In order to make sure to cover all the requirements that are necessary for the project, first I made a clear list of the requirements for each class in the project. This greatly helped me to construct the code without having to miss any of the requirements as I could constantly check and compare my code with the requirement list. Also, I paid great attention on testing classes as well as they can do such efficient work of ensuring all the requirements are covered and do the job. As a matter of fact, when you create testing classes that work properly, you wouldn’t really miss a requirement to cover. With that in mind, I’ve carefully crafted my testing code to test upon each function and constructor. This time, the constructor requirements became a quite key point in this whole project. This is because requirements around the constructors were quite heavy, to speak the least. Some of the examples were like character limit, unique ID, had to be certain number of numbers for setting up phone numbers etc. To overcome these tricky requirements, I knocked down each of them by repeatedly trying different approach and somehow managed to overcome them all. Therefore, overall I had great experience with testing in this respect. When you create robust code with classes, your testing can become much easier to perform. I’ve constructed various testing cases that could cover all the possible patterns of results from the code. Using JUnit’s default testing methods, assertTrue, assertEqual, assertFalse can effectively indicate whether you have constructed your code properly or not. If your code had some errors and testing revealed your flaws, it would mercilessly give you a red with error info such as which test method showed negative results. In my case, I’ve received quite a bit of errors as you can expect, so I repeated the cycle of testing… dah errors, testing… oh that one worked.

Particularly,



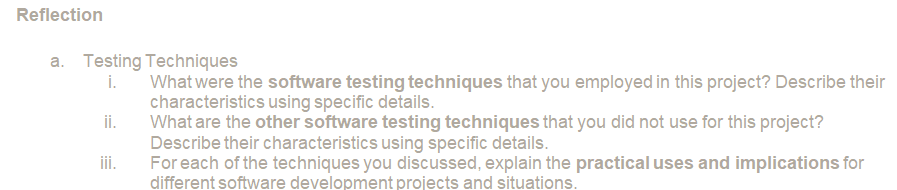
my TestTaskService.java class could test the following cases;

1. if the constructor properly works with each parameters?

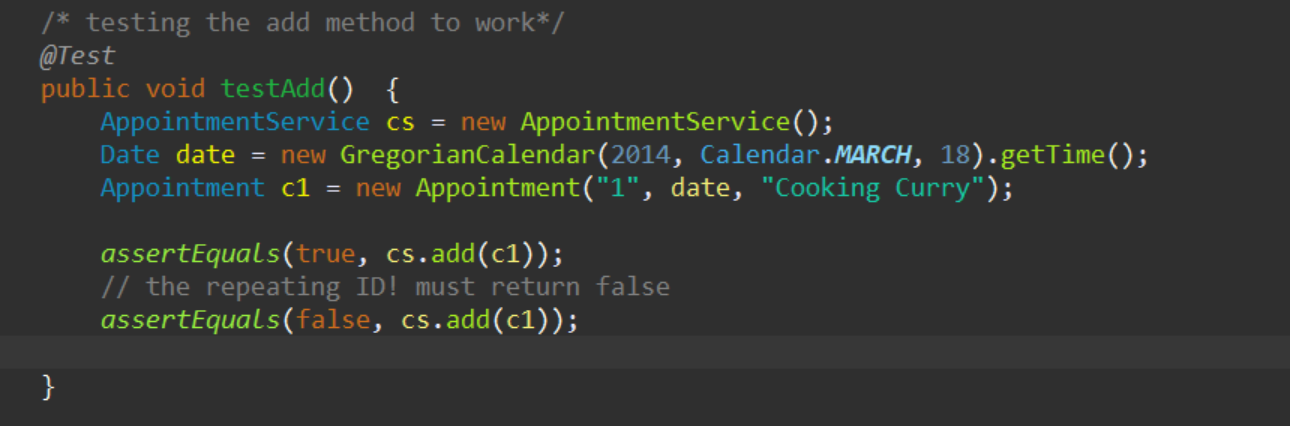
2. Can the add() method properly work and add that instance to the List? (as I made an ArrayList).

3. And then what if I put an non-existing ID and what the program would show?

I could test that by using assetEquals with “false” expected value. And what if the code actually works well and gets through the test, then the assertEquals with true value would prove the triumph to you. As you see, my test case was sort of like All-in-One, One does it all kind of testing that can validate multiple types of results.

****

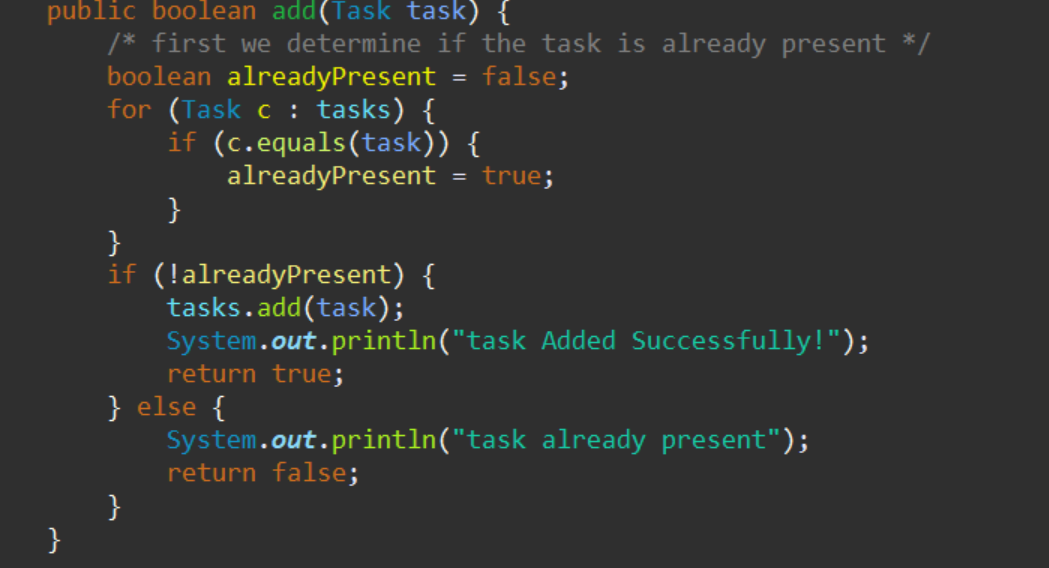
When creating test cases, I took a step-by-step approach to test upon each method. It was actually my first experience to use this JUnit and do some testing. It was important for me to take slow but steady steps because I was so sure I wouldn’t be able to handle a bunch of errors if that happens. Let’s take testAdd() testing for instance:



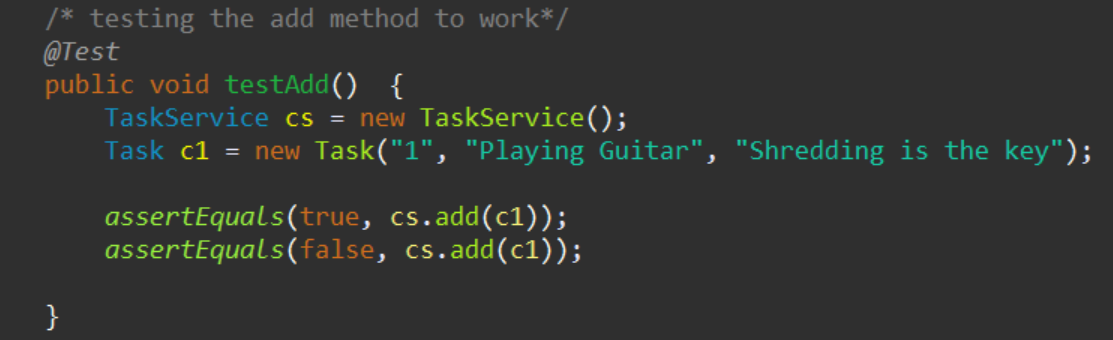
I guess our testing plan was pretty well designed. As you can see, you would have to construct an instance in order to test the methods. If the constructor is well written, then an instance gets created without any problem, but it wouldn’t get created if you didn’t make the constructor properly in the class so the testing fails at that point. By the way, the week 5 assignment was particularly tricky as you can’t just enter the “date” when you create an instance. You would have to create the date by using the library and put that into a variable, and then you could use that variable as valid enter. Just a piece of my experience here.

As I’ve mentioned some, I didn’t really see that it was necessary to use testing classes for constructors, like Test Contact class. In my case, I just left them blank. The reason is that our requirement clearly show that, for an instance, the constructor needs to have unique ID, or the phone numbers have to be exact 10 digits! And those are all well implemented inside the constructor cases, you wouldn’t really have to separately test them in Contact Test class, you can test that in Test Contact Service class if you like.

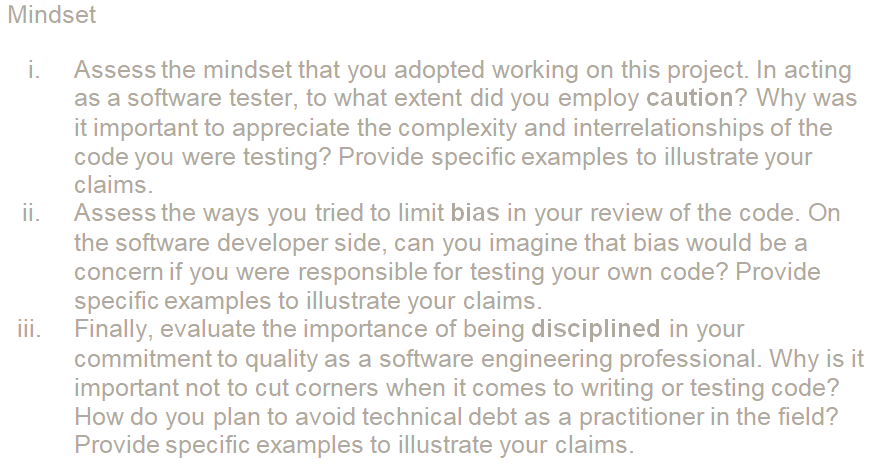
So again the actual testing part wasn’t that bad. This is because I’ve created my method carefully and error-free.



If the method can efficiently add() new instance to the list when ONLY the unique ID is entered, then you can only test that so easily by entering the same IDs TWICE and then you would pick up an error return by inserting assertEquals with expected value of “false” as my docile add() method wouldn’t let you add another instance with the same ID.



.



Testing is extremely critical to successful development. This phase is imperative in order to catch or minimize any possible errors and bugs that could be present inside the application. Again, I have made test cases in order to validate every possible outcome and output from each method, and this is pretty important. As a nature of science, one could only predict so little about how the work you are working on gets so close to perfection, no matter how the development team work so hard, it becomes extremely challenging to prevent any errors from occurring and develop a completely error-free products. You would want to test not only how methods produce positive results but also how they behave and output negative results too. That is because you would never know what kind of side effects the negative results trigger, which can lead to another bug or error in design as well. These are also the reasons it is a best practice to separate development team and QA team. While developing an application, it is somewhat inevitable to grow a certain angle of viewpoint over your own development. Your insight could be slightly sided and objective, it would be highly more encouraged to have the QA team test the application for the capacity to evaluate the products with subjective point of view.

Writing code is extremely unforgiving. You put one typo somewhere in the code and then it affects the whole application. In my past experience, there were so many cases when I tried to cut the corner and then the whole project simply failed. Again, programming is not forgiving to you unlike humans. You could try cutting corners especially if you are good at it, and oftentimes, some projects have extremely tight deadlines and you are left without other option to do so, but you have to be aware of how risky it can be and still you should take best cautions in order to minimize critical errors to occur when you have to cut corners. I am saying I worship the idea of cutting corners but it is a realistic nature of software development field that we always have to battle the time restrictions. Having said that, having discipline and be disciplined when you work on programming is very important. For instance, you seriously need to be aware of how careful you should be handling the data. You would always want to have some ways of backup data, you would always want to keep each different version records and snapshots, and also data recovery systems when the data gets lost somehow. Nowadays it has become quite norm for any application to automatically create duplicates of data simultaneously as you save. This discipline is heavily considered in AWS, as most of their services focus on how users can choose options for how they would desire to make duplications of data.

Above all are the disciplines so far I have learned through my experience and something that I would like to keep in my mind for I dream someday I’d be a rock star programmer, well possibly.

*FIN♬*