Kris Daigle

Project 2

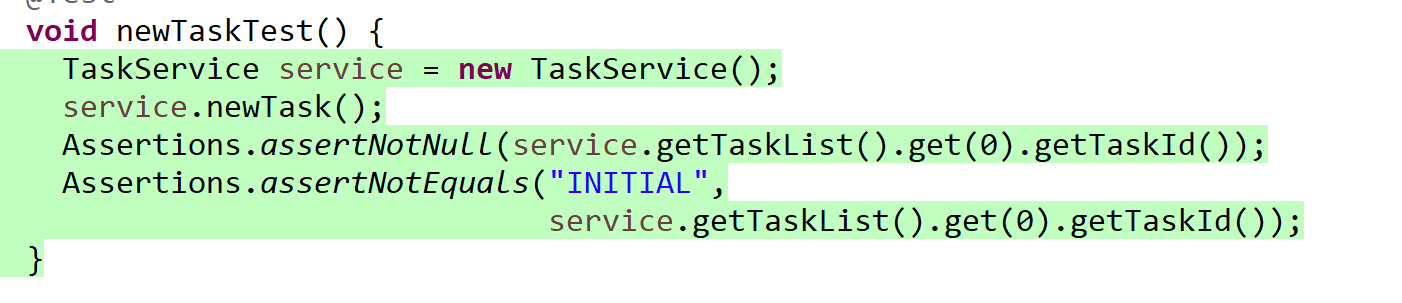
CS-320

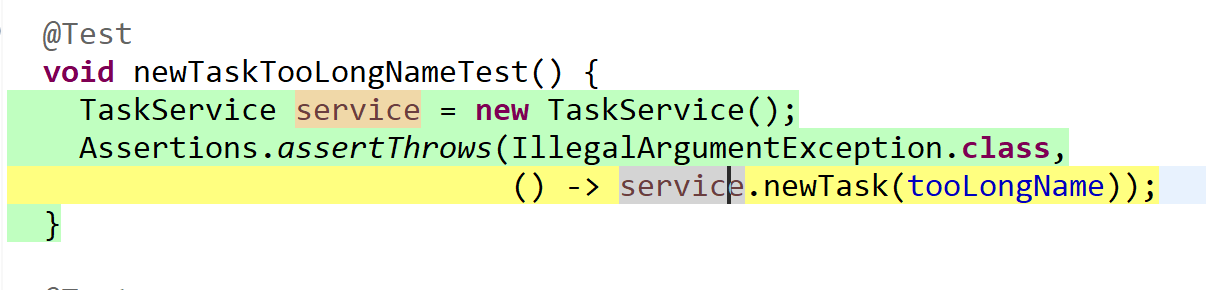
**Summary and Reflections Report**

Unit testing is a vital part of software development. In doing so, testing allows one to reduce the potential for program errors/bugs while in development. This cuts down the development time and makes the process more efficient. For the three features in project 1, I attempted my testing by how well it met the program requirements set by the prompt. Some of these requirements include the length of the entries tested, as well as the requirement that the entry shall not be null. The contact object, for example, could not have a last or first name greater than 10 characters. In order to pass the test unit, the requirements must be met.

The overall quality of my Junit tests were effective based upon the coverage shown after testing. The overall coverage for my project was 93.4%. This was well over the requirement percentage of 80%. Of all of my files, the lowest coverage was 83.3% for TaskTest.

To ensure that my code was technically sound, I tested to ensure that the entry was not null. The lines of code below depict an example of this.



I also used my test objects of strings that were too long to ensure that the code was sound as well. An example of this can be seen in the code below. 

This code checked the parameters for the TaskService class to check for a valid length. This is a good way to ensure efficient code.

**Reflection**

One of the techniques I utilized was dynamic testing. This tests dynamic code behaviors. This was used to test for objects that change within the code. The value in this stems from the vulnerability created by these items. Code that allows for inputs is vulnerability to hacking attempts and corruption. By testing for this, we can try to eliminate some of this vulnerability and protect our programs.

Other software testing techniques include static testing. This tests the code without executing it. I find that this does not find errors or vulnerabilities in the code as easily as dynamic testing. These could include development standard breaches, or inconsistencies.

The differences between these is based on the fact that dynamic testing runs the code. Static testing checks code based on requirement/design documents to find errors. This is to prevent defects. Dynamic testing rather, is to find and remedy defects. To me, static testing seems like a good planning tool in earlier stages of development.

In terms of my mindset throughout this project, I tried to ensure at each step that I was meeting the requirements. This required me to be detail oriented and thorough. By continuing to carefully assess code, it allowed me to attempt to avoid errors from the beginning. It’s important to appreciate the interrelationships of the code, because without taking that into account you may cause an opportunity for vulnerabilities.

I can see where bias can come into play when testing your own code. I think that’s why using testing tools is so important. You can’t lie about JUnit coverage. You need to be honest with yourself about when things don’t line up. I had an issue with my files that was so simple, but severely reduced my coverage percentage. I also think that’s why it’s also important to get another set of eyes on code when you’re able. At my current job where we work with different instrumentation, you cannot quality check an instrument you calibrated. This is to avoid bias. Something similar should be done with coding when its available.

It’s important to be disciplined in your commitment to quality. You shouldn’t cut corners because that will allow for poorly performing code that will later need to be fixed. As an employable person, it’s important to maintain quality code that meets the requirements of your job. If you’re lazy or unprofessional in running your code, then they’ll find someone with that level of discipline. If they require you to fix it, it will take more time. I plan to avoid this while in the field by asking questions, testing as thoroughly as possible and having others in my place of work double check me if I have concerns.

As an example of this, if I work on a team with folks more experienced, it’s important to take advantage of their experience to build my own. There are also online resources to ask questions from other programmers that may help. To be better at doing this, we must practice and gain experience. I will make sure I test to the best of my ability and take advantage of different styles of testing as well.

When thinking about this project, it’s important to ensure things are running effectively. The tests we did ensured that by testing that the contacts have unique IDs, or that the contacts cannot be updated. By keeping this in mind and being disciplined, we avoid future errors within other areas of the code.