Shawn Frye

Software Testing & Automation  
Southern New Hampshire University

**Summary**

For my Java classes, I used JUnit to create test classes for each one. My approach to unit testing each feature was thorough and aligned to the software requirements. I used the assertEquals and assertThrows methods to ensure that each aspect of the feature was working correctly, and that any exceptions were handled appropriately.

For example, in the validation testing, I created tests that verified the correct inputs were accepted, and that incorrect inputs resulted in an exception being thrown. I am confident in the overall quality of my JUnit tests because I believe I achieved good coverage. I made sure to ensure that all lines of code were tested, and that the tests were effective in identifying any bugs or errors in the code.

Writing the JUnit tests was a challenging but rewarding experience. I found that it required a great deal of attention to to detail. Also, the benefit of catching potential bugs and ensuring that the code was technically sound made it a valuable process. To ensure that my code was technically sound, I reviewed each line of code in the test cases and made sure that it covered all possible scenarios. For example, in testing the appointment creation, I made sure that the test cases covered all possible inputs, such as valid and invalid attributes.

I made sure to keep the test cases as efficient as possible. I avoided redundancies and made sure that each line of code had a specific purpose. For example, in testing the update contact feature, I used a single test case that covered all possible inputs, rather than creating multiple similar test cases.

Overall, I am confident that my JUnit tests were effective in ensuring the functionality and reliability of the software. I believe that my approach to unit testing was aligned with the software requirements, and that the quality of the tests was high without bugs or errors.

**Reflection**

**Testing Techniques**

During this project, I employed several software testing techniques to ensure the functionality and reliability of the software. One technique that I used was unit testing, which involved creating test cases for individual units of code to verify that they were working as expected (Jakubiak, 2022). I used JUnit to write test cases for each class in the software, and I employed methods like assertEquals and assertThrows to check for correctness and exception handling. The characteristics of unit testing include its focus on small, isolated units of code, and its ability to catch bugs early in the development process (Jakubiak, 2022). Unit testing is a practical testing technique for all types of software development projects, from small scripts to large-scale applications, and can help ensure that each unit of code is functioning correctly.

Other software testing techniques that were not used in this project include integration testing, system testing, and acceptance testing. Integration testing involves testing the interaction between different units of code to ensure they work together properly (Hamilton, 2022). System testing involves testing the entire system as a whole to verify that it meets the software requirements (2022). Acceptance testing involves testing the system with the end-users to ensure that it meets their needs and expectations (Natchimuthu, 2022). The characteristics of these testing techniques include their ability to catch bugs and issues that only arise when multiple units are working together or when tested in a real-world environment.

The practical uses and implications of these testing techniques depend on the software development project and its requirements. For example, integration testing is crucial for large-scale projects with many interconnected modules, while acceptance testing is important for user-facing software like websites and applications (Hamilton, 2022). System testing is useful for ensuring the entire system meets the requirements, while unit testing is valuable for catching errors early in the development process(2022). By using many testing techniques that are appropriate for the project, software developers can ensure that their software is of high quality, reliable, and meets the needs of its users.

**Mindset**

During this project, I adopted a cautious mindset when working as a software tester. I made sure to test each unit of code thoroughly and was mindful of the the consequences of introducing bugs into the system. It was important to try and understand how it all meshes together. For example, when adding new tasks, I had to consider how the method interacted with the HashMap structure and how it could potentially impact other parts of the.

To limit bias in my review of the code, I tried to approach each unit of code objectively and focus on its functionality rather than any personal preferences or opinions. I made sure to consider all possible scenarios and inputs, and I reviewed the code multiple times to catch any potential errors. Bias can be an issue if a software developer is testing their own code because they may be more likely to overlook errors or make assumptions about the code's functionality. For example, a developer might assume that a particular feature is working correctly because they know how it is supposed to work, rather than testing it thoroughly and objectively.

Being disciplined in my commitment to quality as a software engineering professional is essential because it ensures that the software is reliable, efficient, and meets the needs of its users. Cutting corners when it comes to writing or testing code can result in many issues, which can make it more difficult to maintain and update the code in the future. I plan to stay up-to-date with the latest software development practices and tools and to prioritize testing and quality assurance. For example, I will make sure to test each unit of code thoroughly, document the code and the testing process, and use code analysis tools to identify potential issues before they become larger problems.

References

Hamilton, T. (2022, December 31). What is software testing? definition. Guru99. Retrieved February 17, 2023, from https://www.guru99.com/software-testing-introduction-importance.html

Jakubiak, N. (2022, December 6). Junit tutorial with examples: Setting up, writing, and running Java Unit tests. Parasoft. Retrieved February 17, 2023, from https://www.parasoft.com/blog/junit-tutorial-setting-up-writing-and-running-java-unit-tests/

Natchimuthu, A. (2022, December 21). What is acceptance testing: Types and best practices. Disbug Blog. Retrieved February 17, 2023, from https://disbug.io/en/blog/acceptance-testing