Random Test Data Generation

Project Report



Course: Bachelor of Science (Honours) in Software Development

Name: Kyle Kinsella

Date when document was started: 23/04/2024

Supervisor: Chris Meudec

**All of the issues I have had throughout this entire project.**

Throughout this entire project, I've encountered numerous challenges, but that being said I have learned a lot in the process of making this project, but I have also made lots of mistakes, but most importantly I have learned from them. Below are most of the obstacles that I encountered during this project.

1. Making the right click functionality.
2. Reading one or many method’s specifications.
3. Learning a different type of Java code.
4. Generating a JUnit test.
5. Fixing the speed of JUnit.
6. My laptop has broken.
7. Revisiting overwriting and appending to a file.
8. Boundary values.
9. Installation issues.

**Making the right click functionality** - At the start of my project my first feature that I set out to make was the right click functionality as this is one of the core features of my project [1], to summarise it in a short sentence, it took me a while to get it to work but I eventually got it to work after many hours.

**Reading one or many method’s specifications** - After I got the right click functionality to work, I wrote some code to read a method's specification with the Java programming language [2]. At the start of the project I was only able to read one method specification for a method called “add”, but as I progressed throughout the project I was able to find any name of a method, types of the parameters and how many parameters there are.

**Learning a different type of Java code** - At the start of this project when I was doing some research on how to make a right click in Java, I was exposed to a new type of Java code, initially this was quite a curveball but after about a day or two I was starting to understand what was happening [3].

**Generating a JUnit test** - Once the first iteration was over, I was quite confident that I was ready to make my project more robust. At the start of the second iteration my main aim was to be able to generate a Java file with one or many JUnit tests within the generated Java file [4]. Before the second iteration was completed, I got this feature implemented and many more features.

**Fixing the speed of JUnit** - After I was able to generate a Java file that contained JUnit tests I was running them with JUnit jar files that I had download from the internet and put them onto my class path, but for some reason it was ridiculously slow, so to fix this issue I downloaded a JUnit plugin from the eclipse marketplace and this fixed the speed of the JUnit runtime [5].

**My laptop has broken** - One evening I was just working away on my project and completely randomly my laptop started to show coloured lines on the screen and it also started to freeze. To this day my laptop still doesn’t turn on. But I was saved by my parents. They had a laptop that I was able to use for the duration of the project.

**Revisiting overwriting and appending to a file** - During the second iteration I tried to implement a feature to overwrite a file or append a test to the end of the file, if you didn’t want to overwrite a file. And I was not able to get it to work, but during the third iteration I got it to work for overwriting a file and making a new Java file with new tests in that file, I couldn’t get the appending to work 100% right.

**Boundary values** - Due to me being very confused about boundary values and not really understanding what I had to do, until there was two weeks left in the project, but eventually I got a good understanding of them and I was able to input them into my project.

**Installation issues** - Recently I have been trying to install my software that I have developed onto my machine, but due to the lack of help on the internet, I was not able to do this and I resolved to do something different, see my read-me file in my GitHub [6].

**Accomplishments and Setbacks**

During this project I have achieved many things but I have also not achieved some things that I wish I could have achieved. I will explain the things I have achieved first, then I will talk about some of the things that I didn’t achieve.

Below are my achievements from constructing this project.

* Learning how to write technical documents on my own.
* Making my right click work, after all of the difficulties.
* Learning about JUnit and what it is used for.
* Being able to generate a Java file.
* How to write to a file
* Writing my own code to generate one or many JUnit tests.
* Writing more code to be able to make different types of JUnit tests.
* Re-trying to make a feature that didn’t work before, but I persevered and I did get it to work.
* I didn’t understand boundary values at all, due to what I found on the internet and what I was told by my supervisor and my understanding of boundary values due to my research. But again I didn’t give up and I got it to work.

Below are my setbacks from constructing this project.

* My laptop broke out of nowhere. This was quite a shock to me because I used my laptop everyday and it wouldn't ever do that, so for it to just completely break randomly was very weird to me, and my laptop broke at a really bad time, it broke with about two weeks left in the project, but I didn’t dwell on it, I tried my very best to move on and forget about it.
* Trying to install my software. For the last few days I have being trying to install my software onto my machine, I actually accidentally installed it, but when I tried to see if it worked it gave me a big red error, I spent days and hours on trying to fix it, but I didn’t succeed, so I just threw in the towel and thought of a different idea.
* Package and Java file name. At some point during the second iteration I was trying to allow the tester (user) to work in any package name and any Java file, but again I didn’t succeed in making this happen.

**How I tested my software.**

The way I tested my software was I would have a specific goal / feature that I was trying to make, I would write some code and hope it would work and then I would run it manually and see if the code that I wrote did what I thought it would do. If my code did what I programmed it to do, I just continued to code out more functionality. But if my code didn’t work as expected or it gave me an error I would spend a lot of time debugging my code to try and fix it to work the way that it should.

**What I learned.**

From the beginning to the end of this project, I have done nothing but learn absolutely loads. Before I started this project one of my biggest concerns was making a project all on my own, but in actual fact, it has been lots of fun.I am going to now tell you what I have learned overall from this “Random Test Data Generation” project. I think everything that I have learned will at some point be used at some stage in the future.

* At the very start of this project, I had to write technical documents, and at the start I was not very good at it but gradually I got much better and more confident in writing technical documents. And this is something I will need going forward.
* I learned how to make a right click within the eclipse IDE, in order to be able to make a right click you must have a xml file with some code in that file, you must have dependencies and extensions downloaded into your plug-in project that you have made and your Java file that contains your Java code that you want to run.
* I learned how to write code to be able to read a method specification in Java. This involves being able to get the name of the method, the return type of the method and how many parameters the method has, you can get even more information if needed.
* I learned how to write some new type of Java code that I hadn’t seen before. It is still Java code but it is quite different to a normal Java code.
* I learned how to make a Java file by writing my own code to do so, I also learned how to write to a file, and I also learned how to generate a JUnit test or tests with writing my own code.
* I learned what Junit is and what it is used for. Unit is a testing tool for Java testers and Java developers to run tests on a specific unit of their code, it is also known as Unit testing.
* I learned how to write my own code to be able to overwrite a file if a file exists and I also learned how to make a new Java file if you don’t want to overwrite a previous Java file. I also learned how to append (add) to the end of a file, but I wasn’t able to implement this into my project as I couldn't get it to work properly.
* I learned what a boundary value is, a boundary value is when we have a range of numbers. For example if I set the range to be between 1 and 10 any number less than 1 is invalid and any numbers greater than 10 is invalid, but if we have a number that is in the range of 1 - 10 such as 5 it is a valid number. But this wasn’t the type of boundary values I had for my project, the type of boundary values I had for my project were “-1, 0, 1” for the types int, float etc.

**If I had the opportunity to undertake this project again, I would.**

If I had to remake this project again, I would like for the following to be in my favour. I would like Google and YouTube to be of more use to me, for everyday kind of stuff Google and YouTube are very good, but when you are trying to find certain information on Google to help you or you are trying to find a video on YouTube to also help you, it is very hard to know what to do and how to do a certain task. I would also like it if there was documentation on how to install a custom plugin into your eclipse IDE, this is for the process of installing your own software onto your machine. What I have mentioned might not ever happen, but in an ideal world if it did it would really assist in making this project much better and you would learn even more because there would be more resources to help you do what you need to do.

But from a programming point of view I don’t think I would change anything, because anything that I said I was going to make or I thought of making I did get it into my final project at the end of the final iteration.

**References**

1. GitHub. (2024). Code that I wrote for right click. Retrieved from: <https://github.com/KyleKinsella/Random-Test-Data-Generation/blob/main/I%20made%20a%20right%20click/SampleHandler.java>
2. GitHub. (2024). Code to read a methods specification. Retrieved from: <https://github.com/KyleKinsella/Random-Test-Data-Generation/blob/main/Read%20Method%20Spec%20Code/readMethodSpec.java>
3. GitHub. (2024). Learning a new type of Java code. Retrieved from: <https://github.com/KyleKinsella/Random-Test-Data-Generation/blob/main/Right%20Click%20works%20properly%20now!/selectText.java>
4. GitHub. (2024). Making my first generated test in JUnit format. Retrieved from: <https://github.com/KyleKinsella/Random-Test-Data-Generation/blob/main/Generate%20test/test.java>
5. Eclipse Marketplace. (2024). JUnit Plugin, to fix JUnit runtime. Retrieved from: <https://marketplace.eclipse.org/content/junit-tools-4-spring>
6. GitHub. (2024). Read my read-me file. Retrieved from: <https://github.com/KyleKinsella/Random-Test-Data-Generation/tree/main>