BugQuery

Review by Christina Russo
December 11, 2017

Background and User interface

When it's late at night and the project is due the next morning, where do you go when you run into a weird error? For me, I would venture towards stackoverflow or any other website I can find, but what if I was able to remove the searching part of the process? What if there was some way to cut the searching time in half? This can be obtained through BugQuery, a search engine for stack traces. It helps programmers solve bugs in his or her code in a quick and efficient way, using an index of questions and answers extracted from dedicated web services (like StackOverflow). BugQuery includes, along with the Q/A index, a website and a plug-in for Eclipse. Even though it is still in development, BugQuery is commercially available and can be obtained through the Eclipse Marketplace, as seen in Figure 1. Currently there are only 81 downloads, with a total of 8 in the last month. Generally speaking, it is not widely used within the testing community yet since it is still a work in progress. It is being developed at the Technion as a yearly project but if one wanted to contribute to BugQuery, the developer guide, which includes details about the installation process and the code, can be found on the BugQuery homepage.

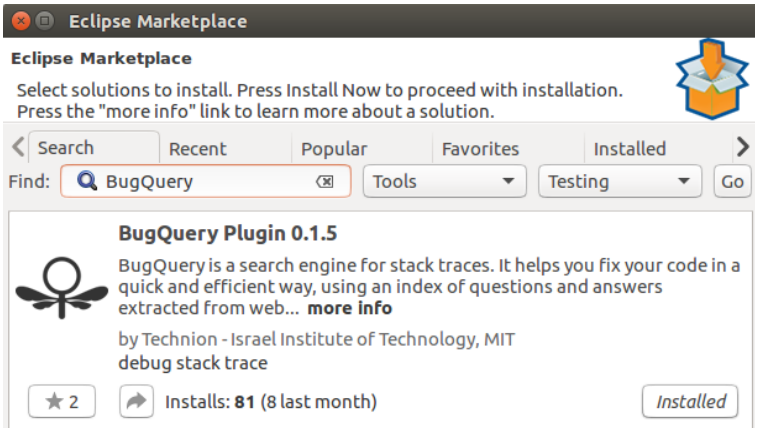

Figure 1

How to Use

One of the many perks of BugQuery is how easy it is to use. After installation, click on the BugQuery tab on the toolbar, select the project you wish you test, then continue to program until an exception is thrown during execution. Once this happens, press Ctrl 6 and a new web browser tab will pop up displaying your current stack trace and 10 similar questions/answers pertaining to your specific issue. The output for an ArrayIndexOutOfBoundsException can be seen in Figure 2.

Figure 2

The following results are from sites such as StackOverflow and have been organized and outline in a Question and Answer format. The original question can be viewed along with the accepted answer. To view each, click on each individual tab until a helpful resource is found. An example following question 6 can be seen in Figure 3 and Figure 4.

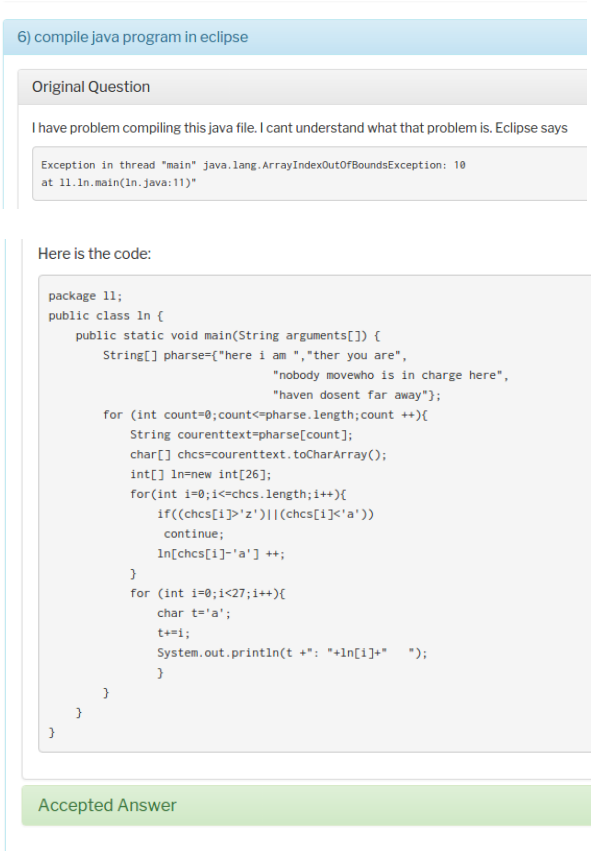

Figure 3

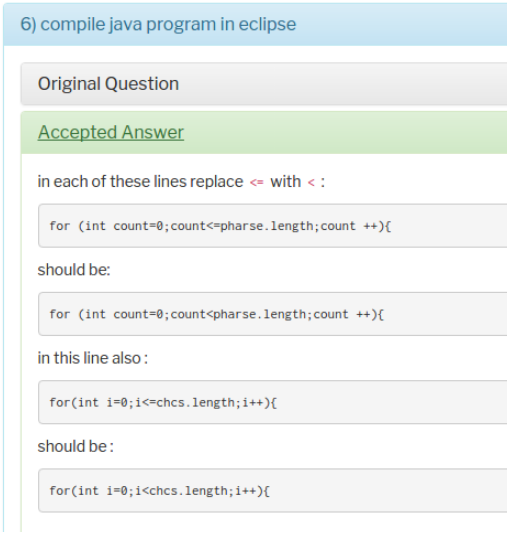

Figure 4

This allows programmers to review other samples of code which encounter the same exception in hopes of finding a similar answer which would assist them in their overall troubles.

Selection of Software Under Test or Program Under Test

To evaluate BugQuery, I made a random set of programs that were built specifically to fail and throw exceptions. These test programs were designed and crafted to test how well BugQuery can produce correct and helpful sources for the faulty code. Table 1 below outlines the exceptions with their results. Correctness is measured in whether the given answer for a quetion is correct and helpfulness is determined if the given answer is sufficient for my program's issue. If N/A is written for any column then a correct answer was never found.

Exception Under Test	Question Number	Correct/Incorrect	Helpful/Not Helpful
ArraryIndexOutOfBounds	6	Correct	Helpful
NullPointerException	N/A	N/A	Not Helpful
FileNotFoundException	N/A	N/A	Not Helpful
NumberFormatException	2	Correct	Helpful
InputMismatchException	1	Correct	Helpful
ArithmeticException	4	Correct	Helpful
NoSuchElementException	N/A	N/A	Not Helpful

BugQuery Helpfulness and Correctness Results

Table 1

Evaluation Process

This level or type of testing is considered Uniting testing, specifically in relation to Exceptions. To evaluate BugQuery, helpfulness and correctness were measured since the goal is to have as many useful and accurate answers when using this tool. The above results are split in half since 4 test cases were considered substantial and 4 were not. This product is still in its development phase thus having these kind of results do not surprise me. It was only able to produce helpful and correct answers for only half of my tests programs but hopefully with time

this testing tool will be more useful. Overall, I would not consider BugQuery to be a quality product just yet.

Under the Hood

BugQuery, as a whole, does not utilize Black or White box testing which causes this concept to be irrelevant for this tool. BugQuery instead tries to help the programmer figure out what exactly went wrong by gathering examples of similar errors and answers that match the stack trace. This tool does not test code in particular but instead assists the programmer to learn and understand from other people's mistakes. The fundamental approach in using this tool is to gather more insight through a researching aid in hopes of figuring out how to fix their own separate issue. So far, since this product is still in development, I would say this tool is best applicable for software or situations that have already been properly outlined by the development team themselves. Since I only obtained results that were half helpful and correct I would hope that once BugQuery furthers their development this would be a much more useful and polished product.

Compare and Contrast

BugQuery may search through a program's stack trace but Samebug recommends solutions and enables collaboration while working on a programmer's code. Both testing tools are available for Eclipse and any other programing language but they are still very different in how they asses error checking. The main contrast between these two tools is that Samebug will only give the programmer possible answers instead of returning specific questions and answers that were caused by the same error, like BugQuery. Samebug actually generalizes each answer and returns

a simple one to two sentence solution to what may have caused your problem. Below Figure 5 demonstrates how Samebug handles its error checking.

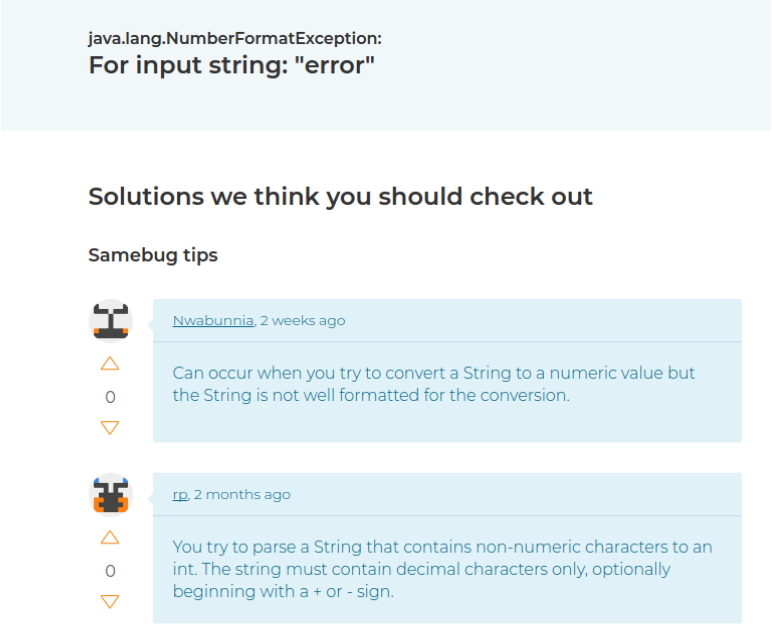

Figure 5

Samebug also is a fully documented and finished product, unlike BugQuery. Developers built an automated “Wikipedia” of software bugs where they collected all the bugs of the world in one place, creating an aggregate knowledge base. This allows them to identify and connect programmers facing the same bug, empowering them to fix it in a collaborative way. This is vastly different from BugQuery since this testing tool only produces similar questions and answers in hopes of inspiring the programer to fix their own error. SameBug goes the extra mile in aiding the troubled programmer by analyzing and compressing possible answers. To further test to see if SameBug is in fact a better product, I used my test cases to measure its helpfulness and correctness. These results can be found in Table 2 below. For consistency, the measurement procedure was kept the same.

Exception Under Test	Question Number	Correct/Incorrect	Helpful/Not Helpful
ArraryIndexOutOfBounds	4	Correct	Helpful
NullPointerException	N/A	N/A	Not Helpful
FileNotFoundException	1	Correct	Helpful
NumberFormatException	1	Correct	Helpful
InputMismatchException	8	Correct	Helpful
ArithmeticException	N/A	N/A	Not Helpful
NoSuchElementException	7	Correct	Helpful

SameBug Helpfulness and Correctness Results

Table 2

Even though SameBug looked more promising its results were not that different from BugQuery. The stack traces that did not produce an answer have a N/A in place since it could not locate a solution Figure 6 displays this kind of failure. Overall, after taking the time to look into Samebug, which has a similar function to BugQuery, I would say its finished product is only a little more helpful and efficient than BugQuery. Samebug was only more useful in one more exception but is there actually enough to be considered the better testing tool? Can one really test

and measure usefulness or accuracy? Only time will tell if BugQuery can truly pass these test cases after it is finished being developed.

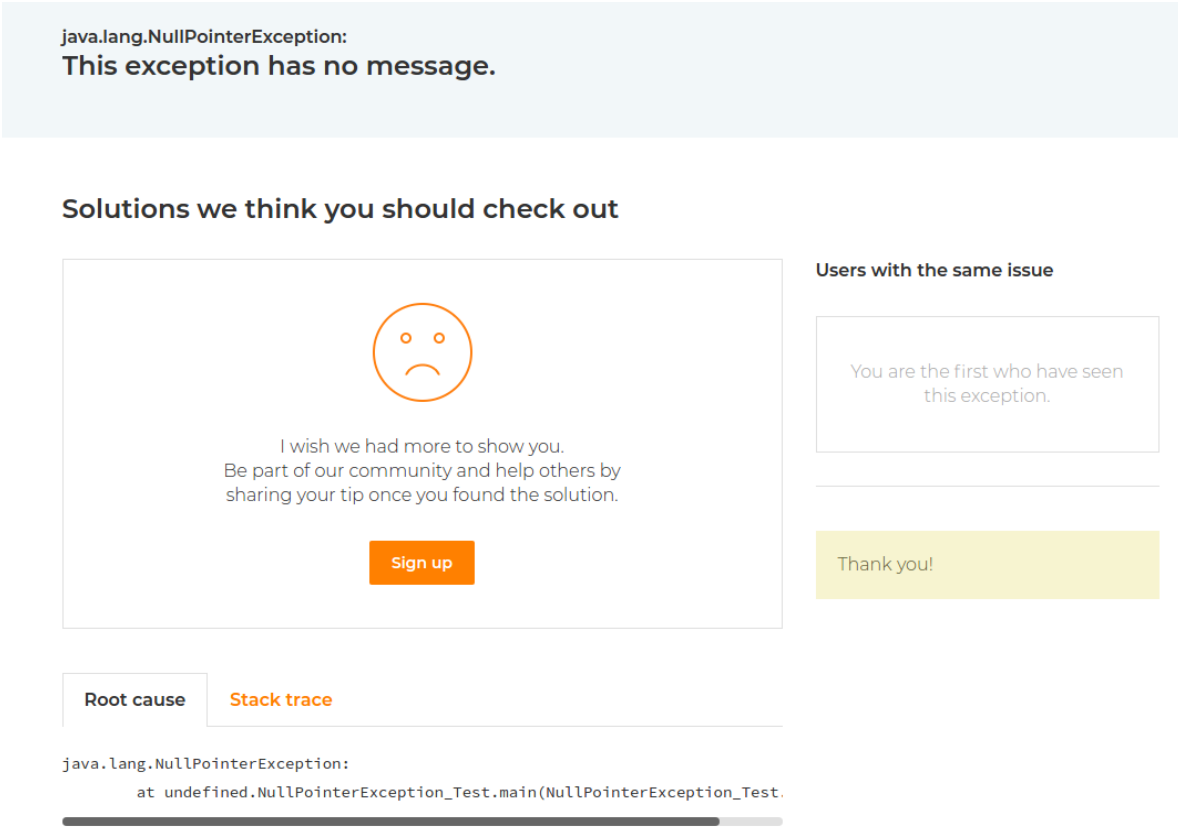

Figure 6

Overall Impression and Conclusion

BugQuery, as whole, demonstrates a wonderful and great concept for error checking. The act of following a full stack trace to find a specific answer cuts down research time and make overall error checking easier and quicker. It is simple enough to use and is just a hotkey away from results. BugQuery may not be the only one of its kind, since Samebug performs a similar service, but overall this testing tool makes error checking and stack tracing easier for the troubled programmer. It will try its best to provide similar questions and answers that previous

programmers have encountered in hopes of inspiring a possible solution for the faulty code. Overall, measuring and testing for qualitative data rather than quantitative made this research project both interesting and conceptually challenging.