Recommendation Report on Debugging and Error Checking video game program code

in the more efficient and timely manner

Cameron Windsor Fox

EGR3350:01

Fall 2017

Mr. Pestak

Problem: Unable to debug code in a video game program in a timely manner without repeatedly having to rerun the debug program multiple times.

Problem Summary:

Company X is a new Indie Video Game company working to build a new RPG (Role-Playing-Game) to release in the next two years. As the game is being developed, the program coders are having issues with testing the code using the debugger that is a part of the Game Deployment Program they are using. The team is getting worried that they won’t get the game built on time if they can’t figure out the problem with the debugger. They are trying to test and check through the code but with debug software only able to move forward in code, it takes time to test repeatedly to track and hunt down errors that the debugger may not be able to find. The company needs to have the game built and finished within 2.5 at max. The main deadline is 2 years with only an additional half year for extra testing if needed.

Possible Solutions

1. Work with unit testing to track your code and look at where errors are happening more closely
2. Work with a different program coding software with a different debug tool that could provide more useful debugging information for fixing errors.
3. Other (Possible extra solution to the problem)

 Scope:

This report will be taking the same program code used for creating a small video game using two different software and debugging tools; Unity’s built-in Mono-Develop and Microsoft Visual Studios. I will be finding the solutions of doing unit testing to test the code piece by piece and then comparing both tools and software to see if one is better than the other. I will be doing this using the criteria: Reliability, Unit Testing, and Cost.

Evaluation Criteria

Reliability (Does this software not only detect an error in your code but also has a way of notifying the developer in a way to get the developer’s attention)

As I was building my small Game on my laptop, I noticed that whenever I made a syntax error (A misspelled word or forgetting to have an ending ‘;’ when creating a new variable or adding in new data). The Unity code software (Mono-Develop) did very little to let me know that I had that issue right away. All that happened was I got a small red line for the syntax error. It is good that Unity could recognize there was an error. Except it couldn’t present it to me in a way that got my attention. The only way to check for errors right away as I type out my code is by going to the “View” tab and click Debug. This then starts the program to check for errors or mistakes in my code. Unity does have a way to make up for this, if you save your code and try to test your game, if there are any errors you didn’t see or forgot to solve, the program won’t let you ply your game and give you a popup saying all errors must be fixed before you can play your game. The downside if you have done multiple code segments and more than one file, you will have to go searching through each one to check and where the errors are from.

By comparison, when I check my code for errors using Visual Studios, this is a much faster process. When I do the same test with my code in Visual Studios if I make the same error and look to see how visual studios handle the problem it is done in a way that can grab my attention much easier. If I make a syntax error or an error in general in my code, I get a small red line as I did in Mono-Develop, but not only that in the bottom of the screen there is an errors tab that is shown that automatically updates if a syntax error comes up while you are in the middle of coding.

Unit Testing (When testing for errors using variables, can the debugger look at the variables and see if they are being used as needed for the specified function)

When going over Unit Testing with both software options using either the built-in Mono-Develop and Microsoft Visual Studios, I can see that when working with the code and creating tests in this software, unit testing works very well. I can see when I make any syntax errors as they appear and when I run to test my code, if there is an issue while I have multiple functions of code running, this software and debugger will pinpoint the function where I am having the problem by looking at the variables and seeing if they change correctly or not while debugging the code line by line. This means that it will take less time to build the code and check for errors allowing more time to continue working on the game and finish it.

When testing with Mono-Develop however, it is different. For example, if when I am working in Visual Studios, if I am using print statements to test a certain function to make sure things are working without errors, Unity does not work the same way with any print statement like visual studios. So, testing each function of code with Mono-Develop will be a completely different setup. If not trained, it will mean using up more time for learning the program to do testing needed to check for errors.

Cost (Work hours used for testing code instead of making sure to finish the game on time)

With using cost in the number of work hours spent with testing and debugging code for errors, by looking at how the coding software Mono-Develop, and Visual Studios handles syntax errors it can be said that at least for the start of error checking and testing Visual Studios will be able to check for errors much faster and alert the developer sooner by showing a pop up indicating where and when an error in the code is made. Where with Unity’s built-in software, if you don’t see the red line, you won’t know you have an error until you run and test your code. What makes that even more time consuming is that Unity will not tell you where the error is located, within a specific function of the code. then working with the Mono-Develop software built into Unity.

Conclusion:

By comparing the two-different software and there debugging tools, each one has it's own pros and cons. Except, Visual Studios, is able to allow an easier test and more convenient error checking and testing while informing the developer when errors occur and the location where they originate from.

Source for Report

I will be the source of information with this project. I am building a game in Unity (A Game Development Program) I am testing the debugging software with Unity’s built-in code software called Mono-Develop and comparing how well its debugger works when I test the same code in Visual Studios. Because unity is compatible with creating program code using either software.