COMP 587 Project Report

Thad House

Unit Testing:

Unit testing was added to the project, however this was also a requirement for 430. So instead of unit testing (which is there, but can’t fully count for this class), automated CI across all 3 platforms, along with code coverage reports on every build were done. The merge of the PR that added this feature can be found here. <https://dev.azure.com/thadhouse/COMP430/_build/results?buildId=688&view=results>

Automatic code formatting was also added as well. The test directory can be found at the following location, but I’m going to focus more on the automated testing. <https://github.com/ThadHouse/COMP430/tree/master/Compiler.Test>

One place that is interesting is I was able to get 100% code coverage on the tokenizer. That can be seen with the following build.

<https://dev.azure.com/thadhouse/COMP430/_build/results?buildId=697&view=results>

The tokenizer is Compiler.Tokenizer.SimpleTokenizer, and you can find all code coverage results on the Code Coverage tab.

During this, I found a bug in the compiler that resulted in unhittable branches; I created the following issue to track this in the C# compiler.

<https://github.com/dotnet/roslyn/issues/44109>

Automated Testing

Automated testing was added to the compiler to ensure large chunks of code could be easily tested. Here are some links to specific examples. There are many more I can provide if necessary. The tokenizer is where most of the focus was in, with some going into the parser.

* Testing all invalid escapes fail correctly (<https://github.com/ThadHouse/COMP430/blob/master/Compiler.Test/Tokenizer/EscapeSequencesTest.cs#L62>)
* Testing all valid escapes work (<https://github.com/ThadHouse/COMP430/blob/master/Compiler.Test/Tokenizer/EscapeSequencesTest.cs#L13>)
* Testing all unicode escape characters (<https://github.com/ThadHouse/COMP430/blob/master/Compiler.Test/Tokenizer/EscapeSequencesTest.cs#L112>)
* Testing variable length escape characters (<https://github.com/ThadHouse/COMP430/blob/master/Compiler.Test/Tokenizer/EscapeSequencesTest.cs#L127>)
* Type validation tests that ensure special tokens match their specified types. Using refection to test all at once (<https://github.com/ThadHouse/COMP430/blob/master/Compiler.Test/Tokenizer/TokenizerTypeValidations.cs#L11>)

PRs

I wanted to make sure PR’s made for 587 were explicit, so I have them listed here. I know I’m missing a few, I just didn’t get to them.

<https://github.com/ThadHouse/COMP430/pull/1>

<https://github.com/ThadHouse/COMP430/pull/3>

Lessons Learned

I’ve learned how powerful automated testing is, and would definitely start with it from the beginning. I would also do more testing while developing. I was so focused on getting the compiler working that I kind of skipped doing any testing for a while, which made me have to go back and add things.