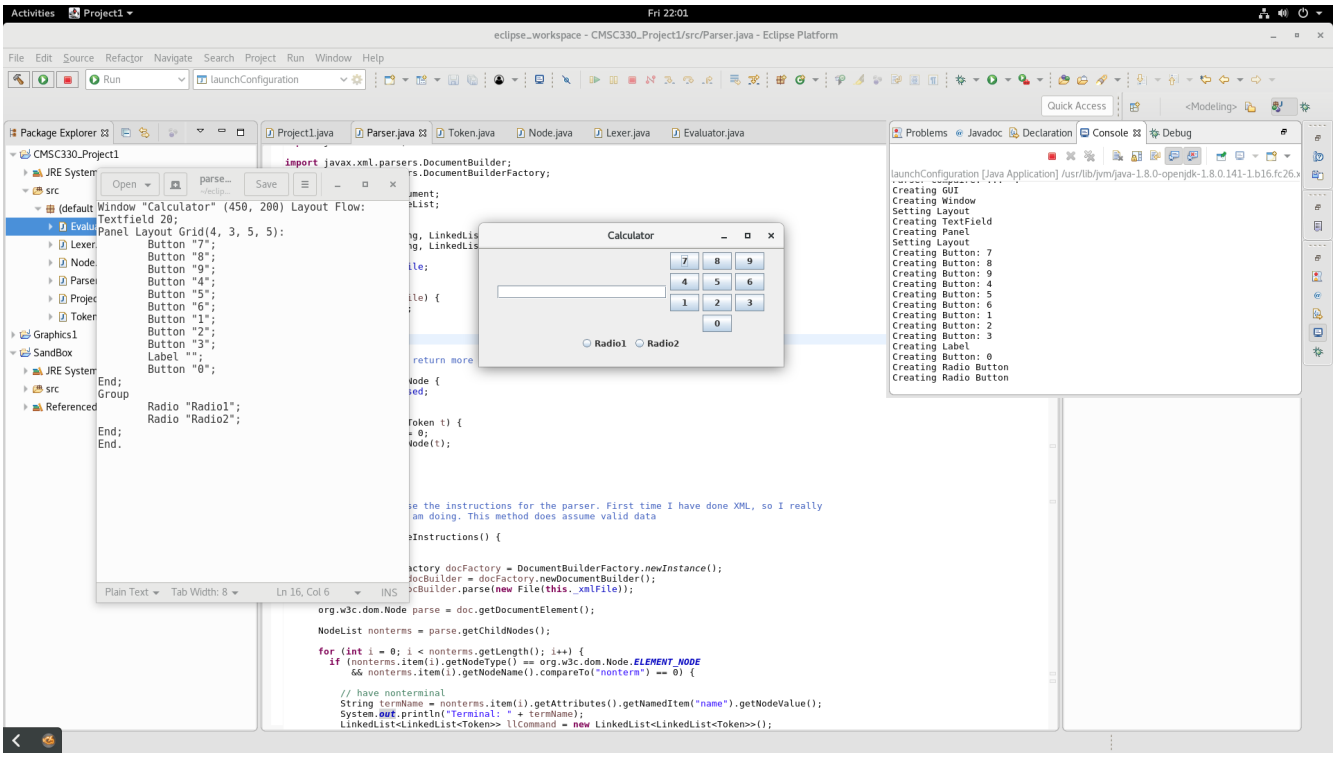
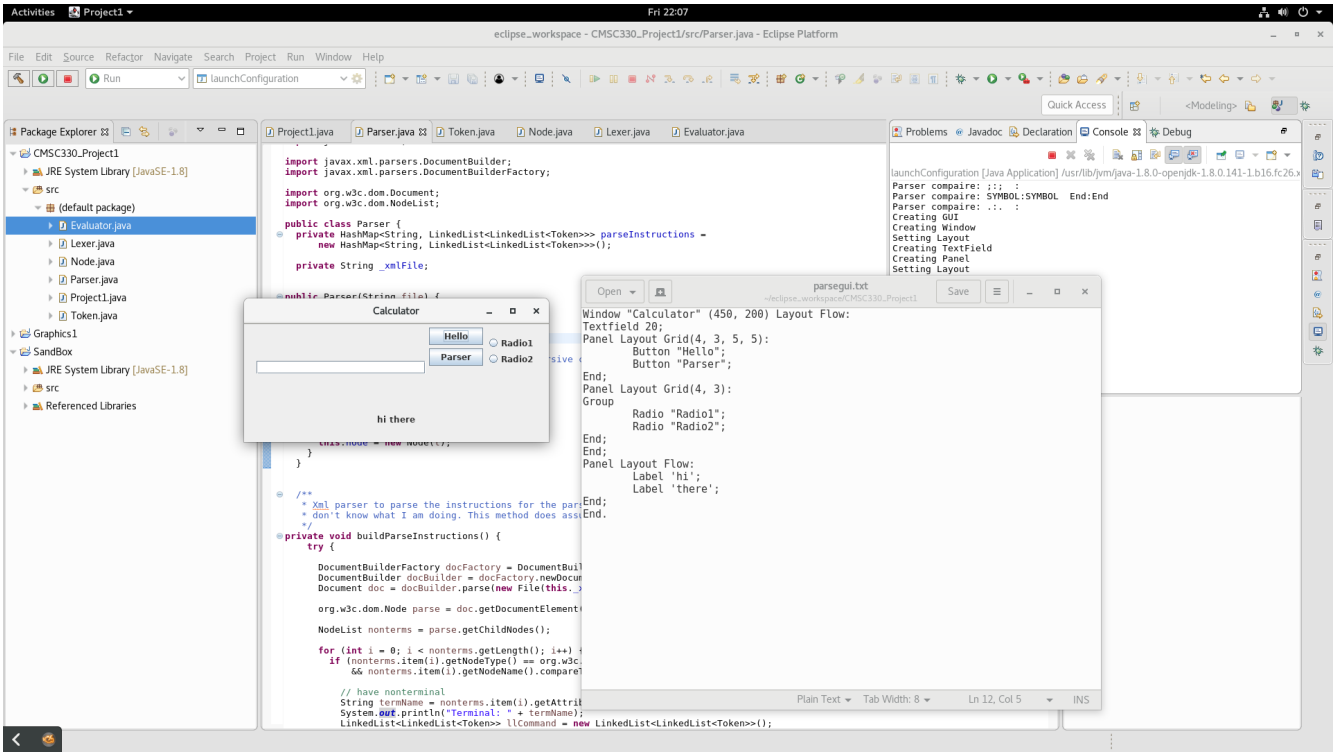
William Kendall
CMSC 330
Project 1

Screen Shots:





Test cases:

1: GUI provided
Using the GUI definition provided after the program was built and running worked flawless.

2: Adding radio buttons
The GUI from the project instructions did not provide a case for radio button, after the gui file was modified to add radio buttons, armageddon broke out. The evaluator would skip any extra panels. After searching the node graph, End instructions are on a higher level node than the objects. This is correct since the end function is for things like Window End; but the evaluator was looking at the panel end symbol. The graph is not complex, thus the End symbol was not needed by the evaluator and removed.

2: empty file or missing
I am not sure on error handling of xml files, however, the gui file crashes on empty. This was patched

3: Bad data
XML file, this file is part of the program and does not change. I will assume that this resource is correct and available. (Would be included in a program at distribution, such as inside a jar file)

gui file, the parser does not complete no matter how hard I tried to fool it.

Lessons learned:

The Bad:
Planning! I was taking this program one step at a time. This was the wrong thing to do, because once I would start working on the next part of the program, more often that not, I would have to return to the previous part and change it. This was a battle of back and forth.

Design: I had the ideas in my head of what I wanted to do, but do not take the time to really work out how it would be done. I started writing way too many system print calls and looking over the output to find out what the program was doing to fix it.

In all, this program took me way to long to finish that it could have, if I had a clear plan.

The Good:
Lexers are amazing. Along with the parser. This program could have been really easy to write just using a scanner and reading the gui data. But using a lexer/parser design allows the program to be more robust. I made a xml file for the parser instructions that can be changed out and the parser and lexer could be reused; as I see myself updating and reusing this code in the future for anything from text input validation to creating video games where levels are created in a text file.