CSC 413 Project Documentation

Muzaffar Sharapov

921548703 *CSC413*

Table of Contents

1 Introduction

1.1 Project Overview

This calculator app can solve basic operations. For example, it can add, subtract, multiply, and divide. Also, the calculator app has a power button and parentheses.

1.2 Technical Overview

In this project we use stack, we push and pop operators and operands.

1.3 Summary of Work Completed

I've completed and passed most of tests. The only tests failed are related to parenthesis. I wish I had more time to solve it.

2 Development Environment

I used visual studio and IntelliJ.

3 How to Build/Import your Project

1. To build/import the project, we need to select SSH, HTTP. Copy a link and then open your terminal. If we want to use SSH key we need to make sure we set up a token already. Then open Terminal, command line on our device. In our terminal type: git clone ssh/hhtp. Once the repo is cloned, follow the steps below to import your project into IntelliJ IDEA. Then build project.

4 How to Run your Project

To run our project, we need to go to EvaluatorUI and click a run green button next to main.

To run/compile the project on visual studio via terminal we could use these commands: javac Evaluator.java javac EvaluatorUI.java

5 Assumption Made

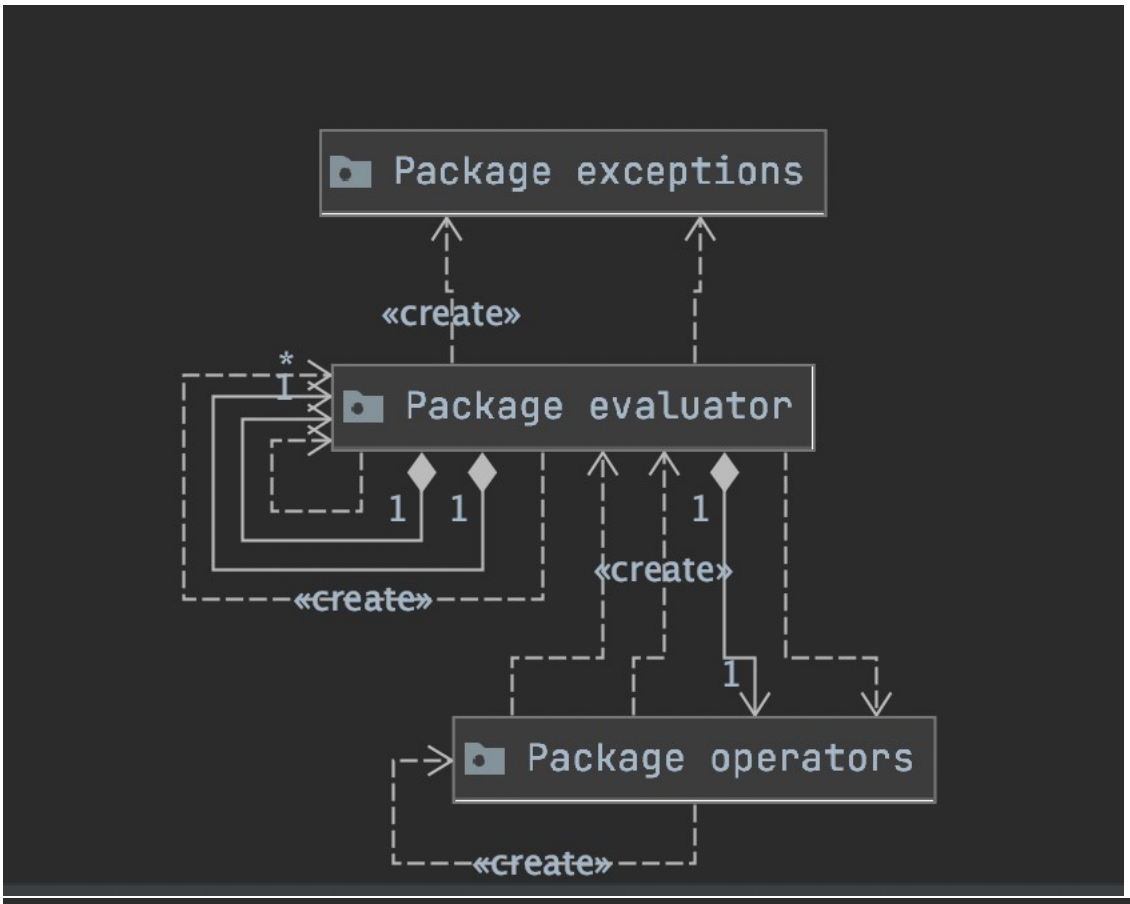
I assumed the skeleton that we were provided was correct and not misleading. Thus, I assumed there were no specific test cases that would break my app from skeleton code.

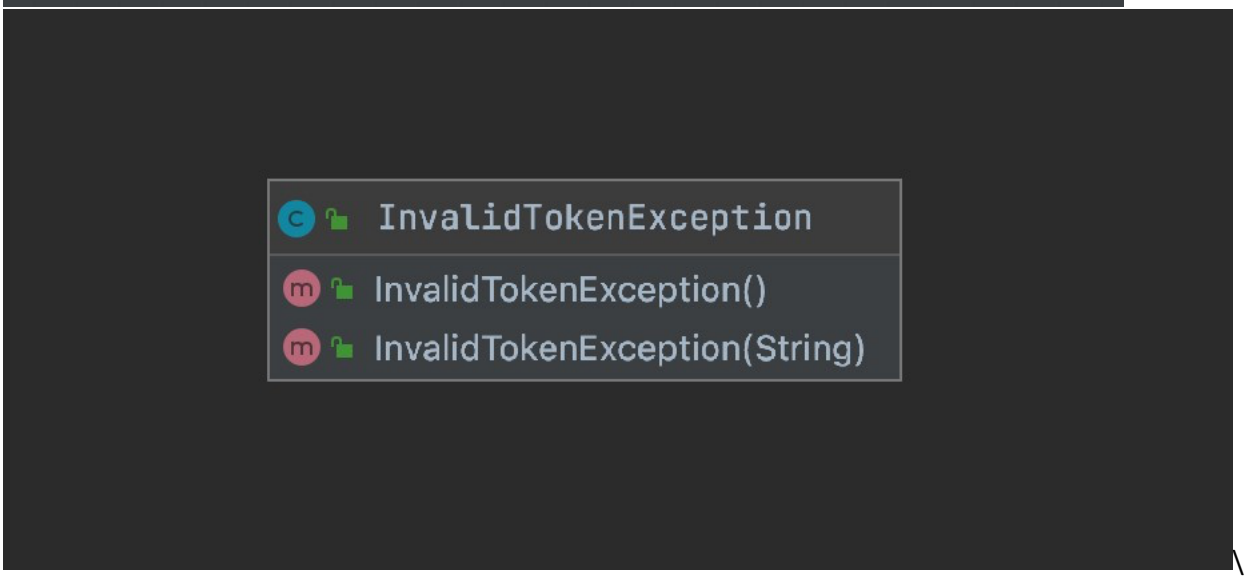
6 Implementation Discussion

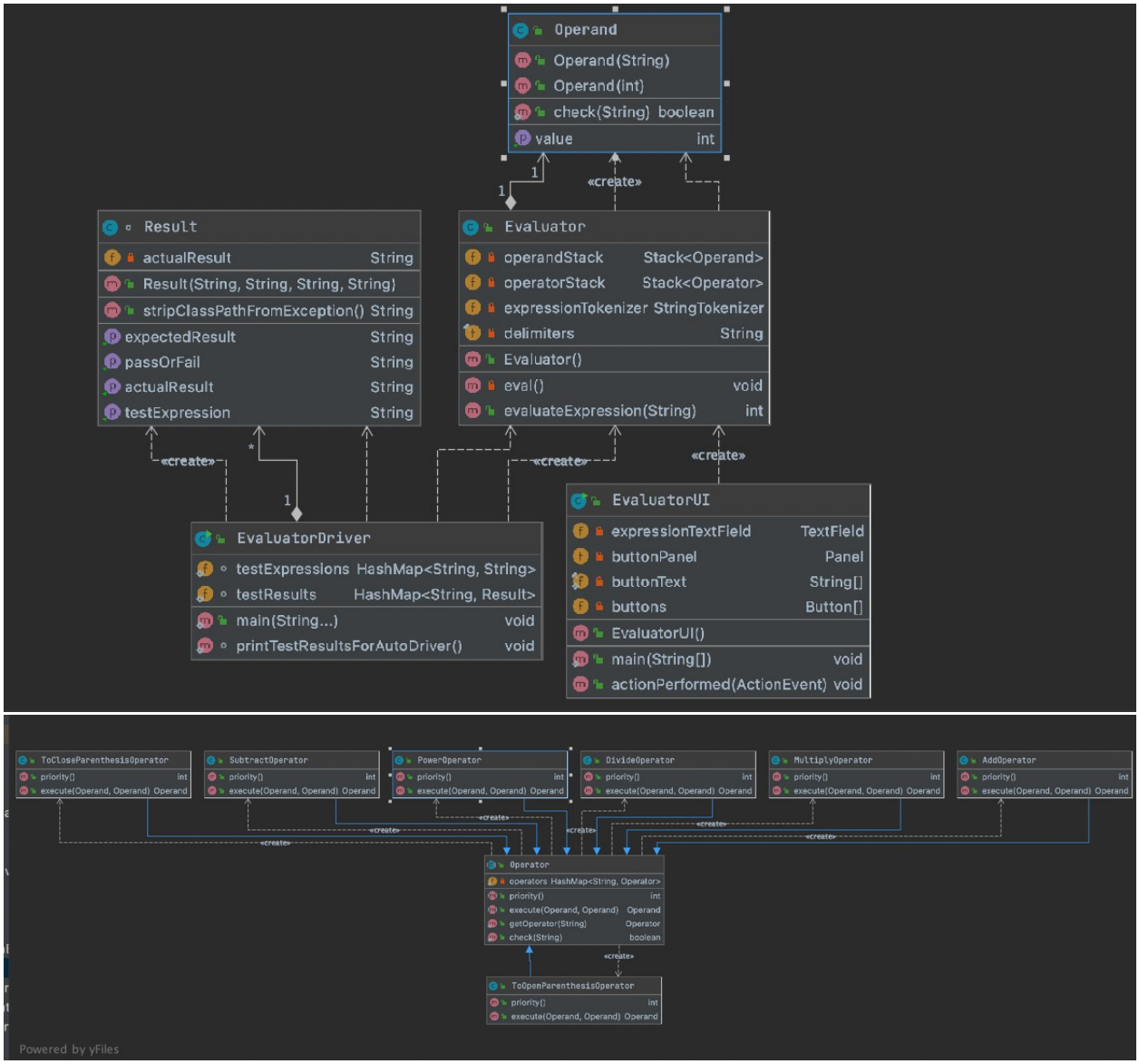
1. In this project, we mainly focus on Evaluator, EvaluatorUI, Operand and Operators classes. Operand class stores the operand tokens of the expression as operand objects. The class contains two constructors, one that takes a string and one that takes an integer, a getter, and a function that checks tokens to ensure the strings are numbers.
2. In Evaluator we mostly focus on GUI.

3 The operator class is an abstract class. Since we're practicing OOP, it’s better to create many subclasses that extend to the Operator class. All the child references are stored in a HashMap as HashMap look-up is the fastest.
In the Expression Evaluator, we do all the heavy work. Strings gets tokenized. We store operands stack for storing numbers and an operator stack for storing mathematical operators.

6.1 Class Diagram







7 Project Reflection

At first it was challenging to follow the skeleton that we were provided. It is harder for me to follow a skeleton than to start from a scratch. The main challenge of the app was the Evaluator Class.

8 Project Conclusion/Results

I wish I had more time. I believe I need to improve parenthesis functions. I know there is a better way to push and pop parenthesis in Evaluator class. At the end of this project I relearned, practiced, and tested my knowledge. The project taught me how to develop and implement object-oriented design, Strategy pattern design properly.