Nick English, Nico Hillison, Leo Qian
CS361
Project 10
5/6/22

Overview

We implemented two main features in this project to convert it from a Java to a Bantam IDE. We updated our application from project 6 before working on the conversion. A check button was added to run the semantic analysis and log errors to the console and keyword highlighting was adapted. To improve the program, we first added a PrettyPrinter class to replace bantam programs with a nicely formatted version. This first generates a Program using the Parser and then utilizes a PrettyPrintVisitor to print the nodes in the AST. Next, we added a Converter to transpile Bantam programs into Java. To minimize code repetition, it extends PrettyPrintVisitor. This means it only has to override three visit methods. The Converter, like the PrettyPrinter, uses the Parser to create an AST before altering it to make a Java program. The output of the process is saved to a file, which can then be compiled and run. The IDE now runs the converter first before compiling and running the result.

Elegance

The main elegance of our project comes from the visitor pattern. Using the pattern vastly reduces the amount of code needed and makes the classes much easier to understand. It also means that our solutions are extremely modular since they only rely on the Visitor to traverse the AST. Changing the compile and run features to utilize the Converter only requires calling a method and altering the input file. The PrettyPrinter is even more modular since it is called by a new menu item, Preferences -> PrettyPrint.

The most problematic part of our new features is probably the comment handling for the PrettyPrinter. While on one hand it only changes and adds a few lines of code in the Scanner and Parser, making it relatively elegant, it sacrifices some things. We store comments in a queue of Pairs with the line number and text. When the PrettyPrinter advances to a new line of nodes, it polls the comments and inserts them. This works well for programs that are on multiple lines. However, if a program is one line all the comments will be printed at the beginning of the new pretty version. The solution to this would be to couple comments to nodes, but that would require a lot more code.

Some improvements

There are still some issues with the semantic analyzer that was pointed out in project 9 report. We still didn't find time to fix them. At the same time, it looks like there's an issue with using the compile and run button after running pretty print, which we need to find the cause of. In the end, the additional features we implemented in the previous few projects still need some

improvements such as adding tabs for multiple lines of code or toggle comments to make them more intuitive.

Group Work

Nick and Leo worked together on updating the project from projects 6 and 9. Nick implemented the Check button and the PrettyPrint classes. Leo created the transpiler to convert Bantam code to compilable Java as well as the logic to compile and run the new code. Both also performed bug testing.