Matthew Oakley

CMPT 220L-201

Dr. Pablo Rivas

8 May 2017

Java Style Application

Abstract

This paper is an overview of a Styling software application which checks Java source code for styling guidelines. Some of these styling guidelines are spaces, two space indent, and proper styling blocks. This program outputs one text file which tells the user how many of each error they have. This program also outputs the information on each line and if it passes styling guidelines or not. This program's goal is to improve Java programmers style and improve code readability.

Introduction

The main motivation for for making a styling program in java is to improve the readability of code. With people learning how to code and not having proper styling techniques costs time and money. The added cost of time is because it will take longer to understand badly written code, while the money is the time spent. This paper will cover what the system does and how users interact with it. It will also contain a UML for each class and the physical requirements of the system resources. The paper will also review other similar products and what makes this one different. It will also contain a user manual which will guide the user how to operate the program and common errors that they can avoid. At the end this paper will look at the significance this program provides to the programming community.

Detailed System Description

This program takes in an input file from the user and checks the file for proper styling techniques. Currently the program only works with .java files. Once the user uses the correct console commands to start the program, the program will check if the user included the .java extension in naming the file. If the user did not include it the program will automatically include it when searching for the file. The program will then check if there is a next line to check. The program will keep doing this until it reaches the end of the file. Each time it takes in input the program looks for tabs at each character location. If a tab is found the program automatically adds that to the total tab count. Next the program will use a function to check the total length of the line. If the line is greater than or equal to 100 characters the program will return '2' which means the line is too long. If the lines if greater than or equal to 80 the program will return '1' which means that the line is close to getting too long. If the line is just fine the program will return '0'. Next the program checks for proper indentation if the program does not have proper indentation then the line will be considered bad. After it checks for proper indentation the program will look at the next line of code and determine if it needs more indents or not based on Java keywords and curly braces. After this is all done the program will output a file called <file name>Comments.txt this is where the user can get quick feedback from the program.

Style
inComments : boolean
tabReader(lineText : String) : int
charCount(lineText : String) : int

isSpace(spot : char) : boolean
indenter(indents : int[], lineText : String) : void
indentKeeper(inDent : int[], lineText : String) : boolean

Requirements

This program has some basic system requirements. The system needs to have Java 8 so it can run the program. This program can run on linux, windows, and mac. This program needs the command line to be able to function. This program needs 10 MB of storage space which contains the file that will be run and the file that will be outputted. This program needs 1 GB of RAM so it will operate at an efficient level. The user needs to be able to use a keyboard so they can type in the file name.

Literature Survey

There are many products which are similar to this one. One example is the online cloud9 IDE which has a style50 check. This check does not work with java and can only be used in this online IDE. The program also has an output file which is not included in the style50 check. This program currently works with most programming languages and will be more user friendly than others. Some other style programs like beautify will style the code for the user. These programs are decent but do not work in an offline setting easily for the user. The Style application works perfectly in an offline setting and is catered towards Java.

User Manual

The user must type in “java Style fileName” fileName is the file the user wants to be checked for style. The user does not have to include the “.java” extension because the program will automatically include it. If the user does include that extension the program will still accept the input. If the user forgets to include the fileName the program will warn the user that they are using the program incorrectly. The program will then show how to correctly use the application. The user also has the option to use the “help” command which can give the user some basic commands that the program has. Another problem is if the file does not exist the program will encounter a problem. The user can get around this by making sure the file exists in that directory. Some minor problems currently with the Style application is that is does not understand double indents for text that is wrapped around.

Conclusion

The significance this project will have is making code more readable and having a standard. If the user uses this correctly it will increase the quality of code in the workplace. This program is made to be easy to use so a new programmer can use it to make sure their code is formatted correctly. It also can be used offline so that the user does not have to be connected to the internet to use it. The significance of this project revolves around its ability for the user to improve on their own styling skills in programming.