#### Homework 3 – Implementation of an advanced maintenance task in an unfamiliar Open Source Software (jEdit)

# Software system and version: jEdit 5.4.0

# Change Request*:*

Add a new feature to jEdit which allows lines to be drawn beneath the text to simulate the look of a paper notepad. Also, there should be an option to allow the user to disable/enable this feature. The location of this control should be under “Utilities->Global Options…->jEdit->Text Area”.

# Detailed Development Environment Description*:*

IntelliJ IDEA 2018.2.4 (Community Edition)

Build #IC-182.4505.22, built on September 18, 2018

JRE: 1.8.0\_152-release-1248-b8 x86\_64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o macOS 10.13.6

OS/Machine: Mac OS High Sierra (10.13.6) / MacBook Air

Working Directory: /Users/jguevara/CS-471/assignments/HW3/jEdit

Apache Ant(TM) version 1.10.5 compiled on July 10 2018

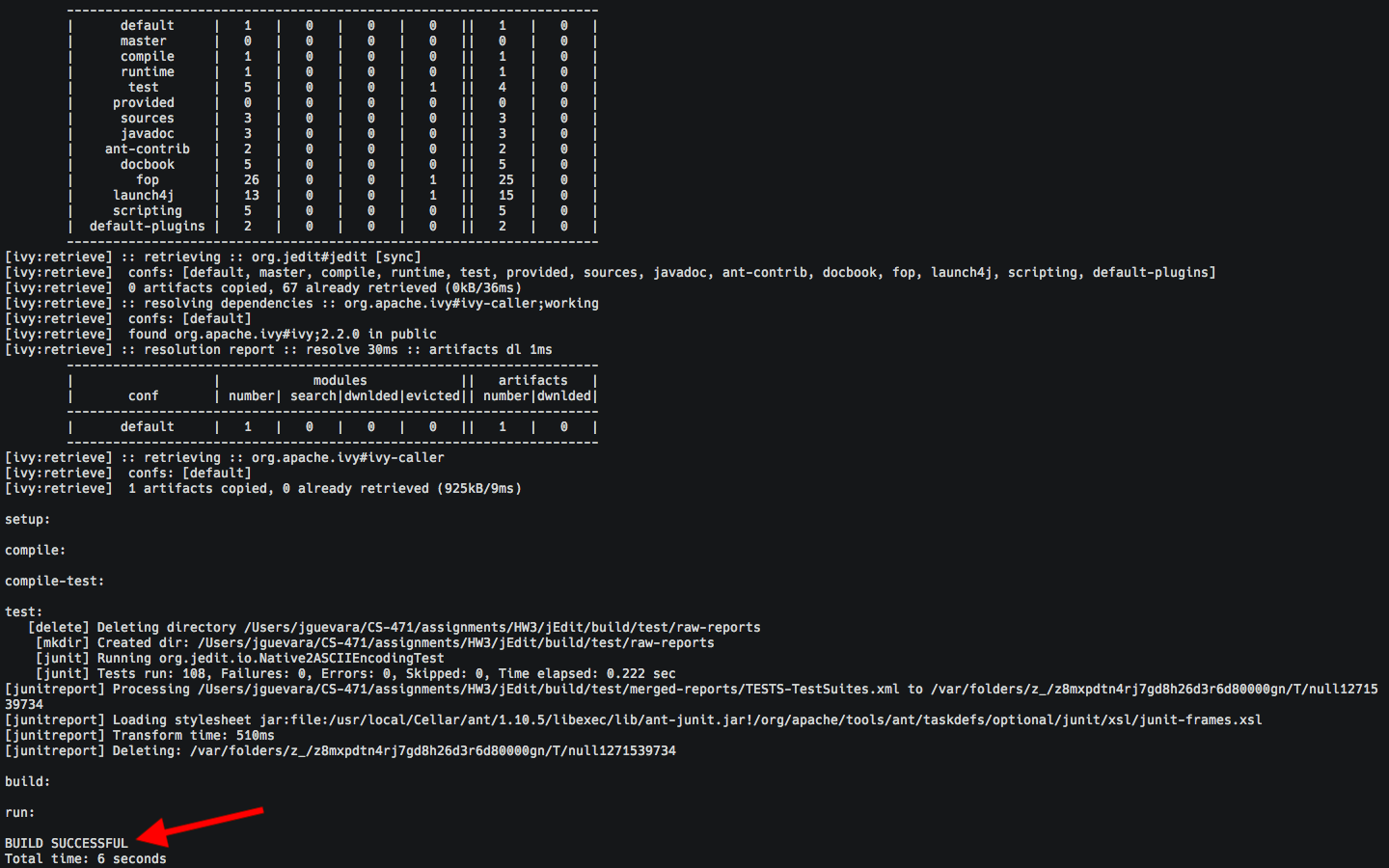
Java Version: Java JDK 1.8.0

Java Home: /Library/Java/JavaVirtualMachines/jdk1.8.0\_161.jdk/Contents/Home

# Description of Problems Encountered during Compiling/Building and Running:

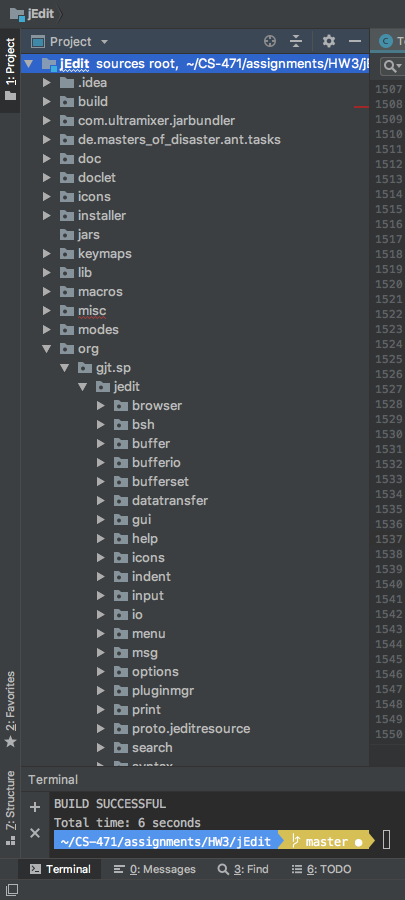
No compilation issues were encountered. However there were a few issues when running because I originally installed jEdit from a disk image for MacOS, so in order to make sure everything would work properly I uninstalled the disk image jEdit, and reinstalled from source (after installing Apache Ant using Homebrew). No problems were encountered when compiling/building/running from command. Importing the project then to IntelliJ was also no problem and getting it to work from the IDE was also no problem. In fact, a lot of my issues came from the fact that I decided to try out working with IntelliJ instead of Eclipse (partially because my project team decided to use IntelliJ in our project and I thought this would be a good way to learn, and get used to, using this IDE).

# Screenshot of successful compilation:



**Figure 1** Screenshot of successful compilation of jEdit 5.4.0 through command-line

MacOS High Sierra (10.13.6)



**Figure 2** Screenshot of successful compilation of jEdit 5.4.0 in IntelliJ IDEA CE 2018.2.4 (Community Edition)

on MacOS High Sierra (10.13.6)

(No compilation errors are highlighted in the Package Explorer)

# Identifying the Relevant Code (i.e., Concept Location):

This feature request was definitely more difficult than the one from muCommander. Unfortunately there was no decisive image, or source, that was useful in determining the location of the area needed to implement the feature request in. However, I was still able to use a partial trick from last time to do this: going to the driver of the whole project. While this one wasn’t labeled IntelliJ is good enough to let me which classes can be run by displaying green triangle over the ones that have a **main** function. I also figured that the one that would be the main driver for this would be the one named **jEdit**.

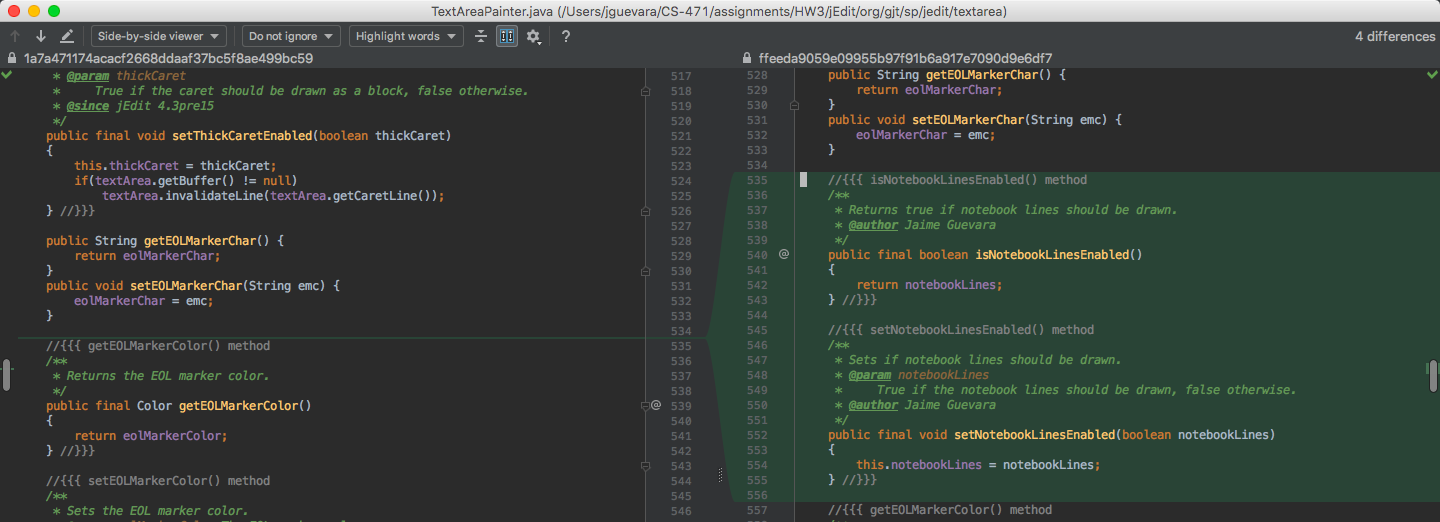
Once the driver for the program was found, all I did was find anything that may have held the name **‘textarea’** (with no case matching) as the feature request asked for us to place the option to toggle this on/off within an area called **Text Area** in the settings menu. I figured it would be called this as its nomenclature would be indicative of what area it controlled, and also because from experience Java programmers tend to have use some form of camel-case when naming their identifiers (therefore searching with no case matching would just search for this identifier regardless of the casing used by the programmers of jEdit). After this, I also asked IntelliJ to show me the very first instance of this as (by the previous logic) the odds that the class where I’d need to make the change would also be named **textarea** (regardless of casing). Luckily this paid off and from the imported package I was able to trace where in the project the directories I needed were found. From there it was just looking for where GUI elements were being called. As for the checkbox, while I was searching for the text area I stumbled upon the option panes and subsequently where and how to implement my checkbox.

# Successful Implementation Video:

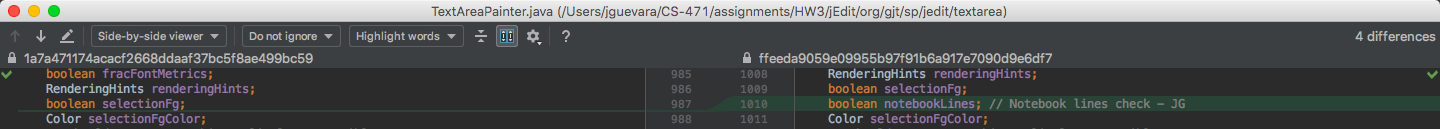
<https://youtu.be/x4JWjiiCYWM>

# Highlighted Source Code of the Implementation in a Side-by-Side readable format:

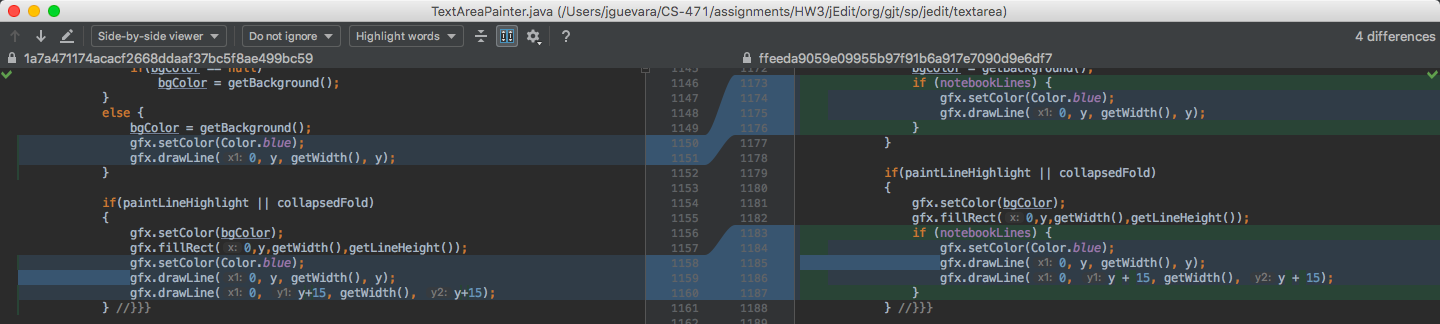
**TextAreaPainter:**



**Figure 3.1** TextAreaPainter – Diff 1

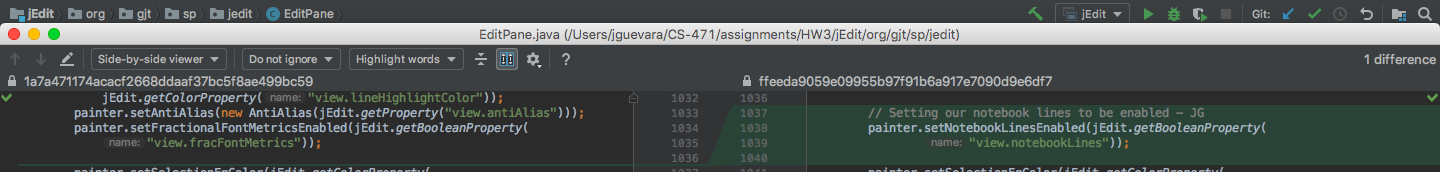


**Figure 3.2** TextAreaPainter – Diff 2



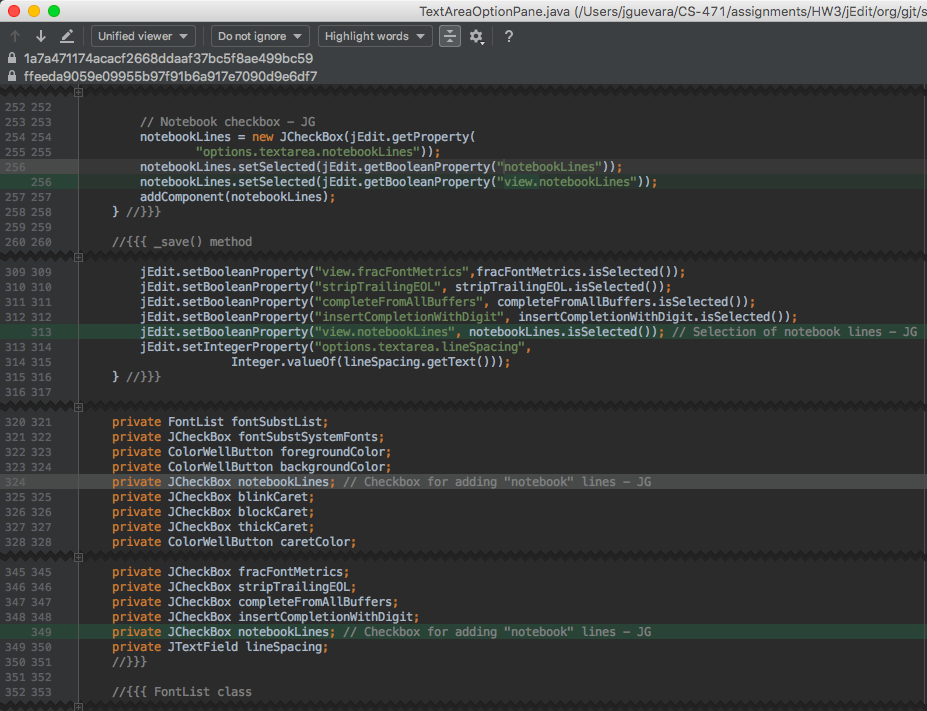
**Figure 3.3** TextAreaPainter – Diff 3

**EditPane:**



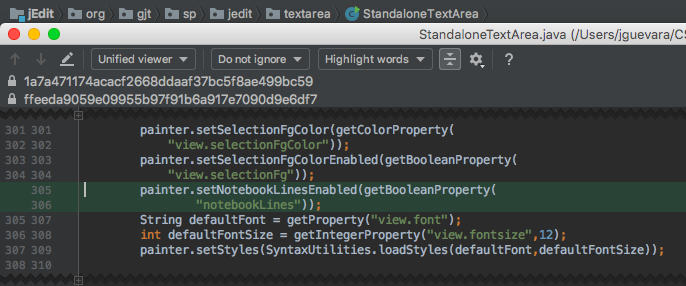
**Figure 4** EditPane – Diff 1

**TextAreaOptionPane:**

****

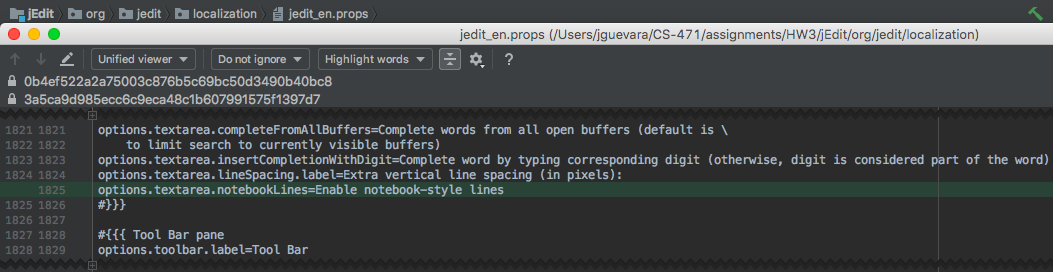
**Figure 5** TextAreaOptionPane – Diffs shown unified for all commit changes

**StandaloneTextArea:**

****

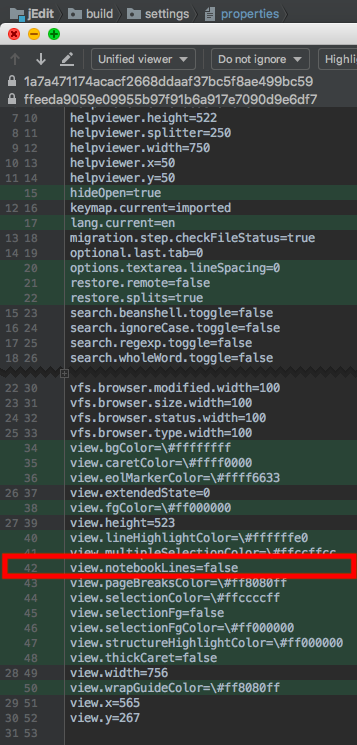
**Figure 6** StandaloneTextArea - Diffs shown unified for all commit changes

**Jedit\_en.props:**



**Figure 7** jedit\_en.props - Diffs shown unified for all commit changes

**Properties:**



**Figure 8** jEdit properties list, there were some additions due to building/compiling so my manual addition was highlighted the red box

# Time Required for Completing this Assignment:

|  |  |
| --- | --- |
| **Activity** | **Approximate Time Required (in minutes)** |
| How much time did it take you to figure out how to compile and run the software (i.e., total amount of time it took to read documentation, download the software, install Ant, etc.), **and NOT how much time it took the builder to build the software**. | ~30 minutes |
| How much time did it take you to understand the code that you need to modify, to implement the change request, and to test it? | ~840 minutes (~14 hours) |