Change Request Log

1. **Team**

This change request was worked on by Chris Zeller and Kristina Kepic.

1. **Change Request**

This change request is change request 5 for jedit software. It is meant to change the find function to highlight all the found cases of the string instead of only highlighting the one that is currently selected. This will allow all occurrences to be highlighted without moving the caret position of the search.

1. **Concept Location**

| **Step #** | **Description** | **Rationale** |
| --- | --- | --- |
| **1** | *We ran jEdit* |  |
| **2** | *We opened up a file and used the find feature* | *To familiarize ourselves with how the feature worked* |
| **3** | *We looked up the search bar API tool documentation to find where the files related to search are.* | *This would give us a better idea of how the search bar works and where the classes that control it could be found.* |
| **4** | *We added a breakpoint to the SearchAndReplace class* | *We selected this class because we knew for sure it would run based on familiarizing ourselves with the search function and assumed it would be integral to the search function.* |
| **5** | *We inspected the searchDialog class.* | *We wanted to see how the string being searched for was stored in the program.* |
| **6** | *We marked the SearchDialog class as located* | *We could not find out what class was in charge of creating the highlighting based on just using the debugger.* |
| **7** | *We inspected the TextAreaPainter class* | *We could not find out what class was in charge of creating the highlighting based on just using the debugger.* |
| **8** | *We ran the program in debugger and decided we could likely modify both of these classes to make every iteration of the word highlighted.* | *The two classes seemed to have methods and variables already in them that could be helpful.* |

**Time spent (in minutes):** 120

1. **Impact Analysis**

| **Step #** | **Description** | **Rationale** |
| --- | --- | --- |
| **1** | *We made a list of classes and methods that were used while searching and replacing* | *This was so we had a list of all the classes and methods in one location and could mark them off the list as we eliminated them from the list of possibly impacted code.* |
| **2** | *We marked the TextAreaPainter class and the SearchDialog class as to change* | *We had decided that these would likely have new methods added to them.* |
| **3** | *We discarded classes and methods that dealt with replacing the words and hyper search* | *We would not need to deal with replacing the words just with searching for them so we decided to mark that off. We also had used hyper search and discovered it already had a form of highlighting that highlighted the string while getting rid of the lines around said strings.* |
| **4** | *We added the text area as a class that could be impacted* | *The highlighting will appear in the text area and could possibly impact other aspects of it.* |
| **5** | *The file SearchBar was discarded from the set* | *The search bar would be unaffected by highlighting that occurs after it is used.* |
| **6** | *Files related to the search buffer, searchDialog, SearchMatch, TextArea, and TextAreaPainter were kept in the set and marked as the most likely to be impacted* | *These were the ones that we would purposefully use and therefore should be checked the most thoroughly for impact analysis. Other files might be impacted but these ones are the ones that we are purposefully impacting.* |

**Time spent (in minutes):** 45

1. **Actualization**

| **Step #** | **Description** | **Rationale** |
| --- | --- | --- |
| **1** | *We created the method highlightAll() inside the searchDialog class* | *We wanted to do it in this class because it was easier for us to understand how the method would work if we had immediate access to the string put into the search bar, but putting it in the search bar class made no sense.* |
| **2** | *We created a new method in the TextAreaPainter class that was meant to paint the lines without getting rid of previously painted lines. We did this by making a method similar to the setBounds method but without the calls that get rid of the previously highlighted lines.* | *By creating a new method based off of an old method we hoped to save time.* |
| **3** | *We ran the code, it did not seem to work correctly and nothing had seemed to change inside the original application.* | *To test to make sure everything worked.* |
| **4** | *We began debugging to try and figure out what is wrong.* | *By going through the debugger we hoped to better understand the behavior of the application with our new methods added.* |
| **5** | *Despite debugging we did not have the time to fix the methods and had to turn in not working code.* | *Due to starting late for exams and one of us having a long flight tomorrow we did not have time to fully figure out what was wrong.* |

**Time spent (in minutes):**140

1. **Validation**

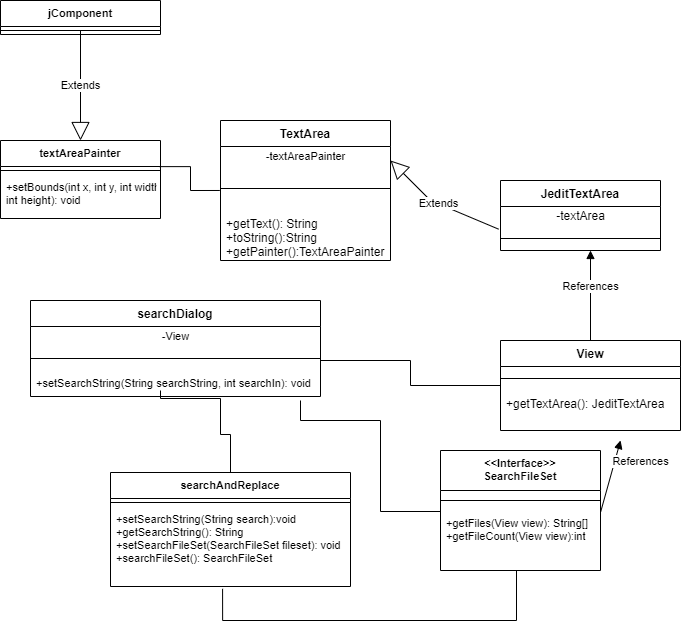
| **Step #** | **Description** | **Rationale** |
| --- | --- | --- |
| **1** | *Inputs: word that only occurs once in an opened document*  *Expected output: The word should be highlighted* | *This tests if it works for only one occurrence of the word. It passes the test.* |
| **2** | *Inputs: A string that does not occur in the document*  *Expected output: The application immediately says no more matches were found* | *This tests if the function properly throws an error message when the string it is looking for can not be found. The test passed.* |
| **3** | *Inputs: A string that occurs multiple times in the same file*  *Expected output: The application highlights each occurrence of the string* | *This tests if the application highlights a string multiple times at the same time. This test was failed.* |
| **4** | *Inputs: Nothing is put into the search bar*  *Expected output: Nothing happens when find is clicked* | *This tests to make sure a null string does not have any sort of result. This test was passed.* |

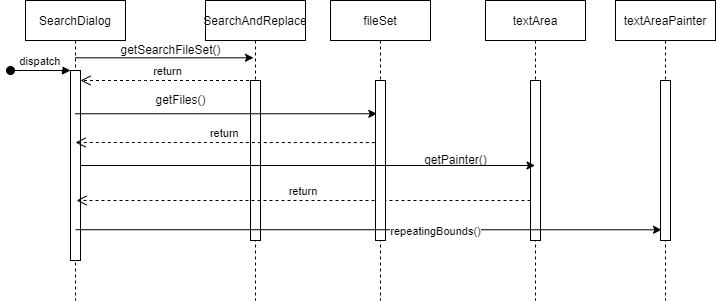
**Time spent (in minutes):**20

1. **Timing**

| **Phase Name** | **Time (in minutes)** |
| --- | --- |
| Concept location | 120 |
| Impact Analysis | 45 |
| Actualization | 140 |
| Verification | 20 |
| **Total** | 325 |

1. **Reverse Engineering**





1. **Conclusion**

This request was very difficult for us to implement. When reading through the possible change requests we underestimated the difficulty we would have throughout multiple stages of implementing the request. This lead us to spend more time on it then we originally expected. The concept location took a very long time and the whole function of the search was much more complicated then we expected. We also had trouble actualizing the code, likely due to our issues understanding the classes that made up the search function. All these factors combined lead to us having code that did not work despite us spending a very long time trying to get it implemented.

Classes and methods changed:

org/gjt/sp/jedit/search/SearchDialog.java/SearchDialog

* Void setSearchString(p:String,View)
* Void higlightAll(p:View,String)

org/gjt/sp/jedit/textarea/TextAreaPainter/TextAreaPainter

* Void repeatingBounds(p:int,int,int,int)