Change request log

# Team

Youssef Donia, Ryan Lea

# Change Request

Change request #2. Need to make the highlighter highlight a file name if the string in the search matches any part of the file name, not just the beginning.

# Concept Location

Use the table below to describe each step you follow when performing concept location for this change request. In your description, include the following information when appropriate:

* IDE Features used (e.g., searching tool, dependency navigator, debugging, etc.)
* Queries used when searching
* System executions and input to the system
* Interactions with the system (e.g., pages visited)
* Classes visited
* The first class found to be changed (this is when concept location ends)

When there is a major decision/step in the process, include its rationale, i.e., why that decision/step was taken.

Make sure you time yourselves when going through this process and provide the total time spent below.

The following is an example of a concept location process for the change request "Color student schedule":

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We ran the system* |  |
| 2 | *We interacted with the system: after logging in we entered the schedule screen.* | *To get familiar with some of the features of the system, and identify the screens or graphical elements we had to change.* |
| 3 | *We searched for "Highlighter class" using the regular expression feature of the IDE's search tool.* | *Because we noticed a keyword in the change request instructions was “highlight”* |
| 4 | *After getting 5 results, we found the Highlight class in the StructureMatcher.java file. We then navigated to the file.* | *We selected this class because we assumed that to change the way things are highlighted, we need to make a change in the methods to do with the highlighter class.* |
| 5 | *We inspected the class method getOffsets in the Highlight class. We went to this class using the dependency navigator of the IDE editor.* | *It looked like the getOffsets method was responsible for where to highlight.* |
| 6 | *We put a break point at line 155 to check if we were correct.* |  |
| 7 | *We then examined the getOffsets method. We then put a break point there and ran the program in debug mode.* | *It seems that the method that paints (highlights) uses the getOffsets method to determine where the line starts and where it ends.* |
| 8 | *We were not able to figure out whether this method was the root of the problem or not, but we tried making changes anyways.* | *While we were not sure if changing the method would help us successfully implement the change, it seemed logical to us that the offsets determined where the highlighting would be done.* |
| 9 | *We could not fully confirm whether or not this class would lead to the solution or not.* | *We could not confirm this class had to be modified.* |

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

# Impact Analysis

Use the table below to describe each step you follow when performing impact analysis for this change request. Include as many details as possible, including why classes are visited or why they are discarded from the estimated impact set.

Do not take the impact analysis of your changes lightly. Remember that any small change in the code could lead to large changes in the behavior of the system. Follow the impact analysis process covered in the class. Describe in details how you followed this process in the change request log. Provide details on how and why you finished the impact analysis process.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *Tried to make a loop at first to loop through the entire file title before determining the offset for highlighting.* | *We thought the problem was the offset* |
| 2 | *Realized that maybe the offset was not the problem, but that maybe we needed to make a change to the way the words match, so we investigated the match class.* | *We thought maybe the offset is purely for creating text Areas and creating the highlighting feature, so we needed to change the match class instead.* |
| 3 | *We could not find anything in the match class, so we instead entered “search” in the search query as a last resort to find a way to change the search feature itself, but we could not find anything other than the searchChildren ColumnBlock.* |  |
| 4 | *We discarded all of the above classes as classes that will help us make the change request* | *Because the classes all seemed to deal with something else, to the extent of our knowledge of course. We did not know how to identify this.* |

**Time spent (in minutes):** 60

# Actualization

Use the table below to describe each step you followed when changing the code. Include as many details as possible, including why classes/methods were modified, added, removed, renamed, etc.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We tried implementing loops in the offset method in the highlighter class, but to no avail.* | *We thought at first that that would help us change the matching, but realized later that we were mistaken.* |
| 2 | *We then tried to make changes to the match class, but could not determine what there needed to be changed.* | *We have a lack of knowledge on what to search for and could not identify anywhere in the match class we could change or add to make it match differently. The code was confusing.* |
| 3 | *We did not have enough time OR knowledge to implement the test cases .* | *We have not seen anything like this before, and looked every place we could to identify possible changes to be made, but could not find anything, and we are not familiar enough with what the software even does to be comfortable enough to add our own changes, and we did not have enough time to get familiar with the software.* |
| 4 | *...* |  |

**Time spent (in minutes):** 35

# Validation

Use the table below to describe any validation activity (e.g., testing, code inspections, etc.) you performed for this change request. Include the description of each test case, the result (pass/fail) and its rationale.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We were unable to create meaningful test cases for the code because we were not able to implement the change properly.* | *We tried, but were unable to make code to test.* |
| 2 | *We now believe that the change may have something to do with regex strings, but all we found was the go to line function, and we failed to make one for the recent files box, or find one.* |  |
| 3 |  |  |
| 4 | *...* |  |

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

# Timing

Summarize the time spent on each phase.

|  |  |
| --- | --- |
| Phase Name | Time (in minutes) |
| Concept location | 120 |
| Impact Analysis | 60 |
| Actualization | 35 |
| Verification | 20 |
| Total | 235 |

# Reverse engineering

Create a UML sequence diagram (or more if needed) corresponding to the main object interactions affected by your change.

Create a partial UML class diagram of the classes visited while navigating through the code. Include the associations between classes (e.g., inheritance, aggregations, compositions, etc.), as well as the important fields and methods of each class that you learn about. The diagram may have disconnected components. Use the UML tool of your preference. When a significant fact about a class or method is learned, indicate it via annotations on the diagram. **For each change request, start with the diagram produced in the previous change request. For the first, you will start from scratch.**

Recent files search

TextArea class

Highlighter class

# Conclusions

Perform and analysis of the change requests and the change process. List the major challenges this change request posed.

List all the classes and methods you have changed.

For example:

*For this change, concept location was relatively easy because the system is small and its architecture and code are not complicated. Concept location, impact analysis, actualization (and change propagation) was done using JRipples, which was very useful. Testing was performed using JUnit and Abbot. It took me long time to make Abbot work, mainly because I had to learn how to use the tool.*

*Classes and methods changed:*

* *org/gjt/sp/jedit/search/HyperSearchResults.java/HyperSearchResults*
  + *void foo(p:integer)*
* *org/gjt/sp/jedit/search/HyperSearchResults.java/HyperSearchFind*
* *etc.*
* For this change, concept location was very hard, as we were not familiar with the software, and most of us are not familiar with the structure either. We looked through many different parts of the code to try and determine whether or not they changed the code, but were unable to properly figure it out. We determined at the end that the change could possibly be made in JeditTextArea.java file as a method possibly similar to the showGoToDialog method, but we had troubles implementing it. We tried changing the Highlight class, the Match class, and the JeditTextArea class.
* - /Users/ydonia/GIT/cs3354-f21-team11/org/gjt/sp/jedit/textarea/JeditTextArea.java