Change request log – je3

# Team

Binh Doan - developer

# Change Request

In jEdit, the HyperSearch feature should list all occurrences of the search string (different Search>Find, which only

locates the next match). However, the list of results of a HyperSearch only highlights the first occurrence of the

search string in any given line, as shown in Figure 1 for the search string "the". Note that lines 522, 533, and 534,

among others, contain more than one occurrence of this string, but only the first one is highlighted (in purple). The

request is to fix the HyperSearch feature, so that its list of results highlights all occurrences of a search string.

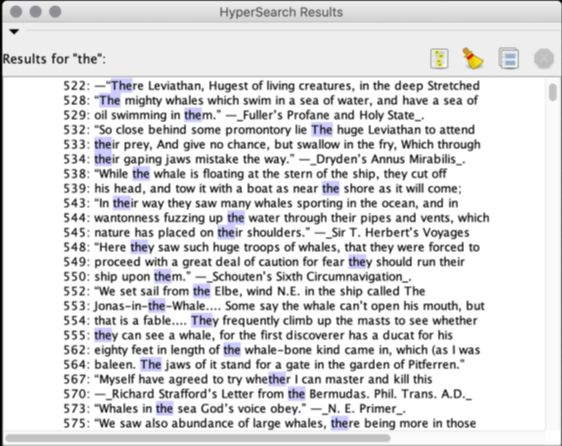


Figure . List of results of a HyperSearch in jEdit. Only the first match is highlighted (in purple) in each line

# Concept Location

The table below describes each step I perform the concept location for this change request.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | I executed the command $ant retrieve build run to start the jEdit application. | Ant build tool is used for this Java project. |
| 2 | “HyperSearch” and “highlight” are the concepts extracted from the change request. |  |
| 3 | Doing a file search with “hypersearch\* with no case sensitive and no whole word features, I found 1047 matches. However, from the previous 2 tasks, I know the location of the source code is under org/gjt/sp/jedit. In this path, there is a directory named “search”. Within this directory, there are some HyperSearch\*.java files. | HyperSearch is a popular feature that should be implemented in a separate class/file under search feature. It should be spotted somewhere in the source code without difficulty. |
| 4 | Doing another file search for “highlight” within the “search” directory and “HyperSearch” concept, I have only 1 result: HyperSearchResults. Marked the HyperSearchResults class as “located”. | The previous result is still broad. Another search will narrow it down. |
| 5 | Within this class, I see there are some “highlight” objects:   * “highlight” of type RolloverButton -> discard * HIGHLIGHT\_PROP of type String -> discard * HighlightingTree class -> probably * parseHighlightStyle() -> probably * highlightString() -> probably | Even though the HyperSearchResults class is marked as located, it is better to locate a more specific area within the class. |
| 6 | Within the “located” HyperSearchResults class, temporarily marked HighlightingTree class, parseHighlightStyle() and highlightString() methods as “located”. |  |

**Time spent (in minutes):** 30

# Impact Analysis

The table below describes each step I follow when performing impact analysis for this change request.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | From the jRipples’s Impact Analysis, we have a list of classes from the located class HyperSearchResults. | To track the classes that could be impacted by the change. |
| 2 | The list is long, and it will take a long time if I go over all classes in the list. | I went over some classes and I am not certain if these classes are subjected to be changed or not. |
| 3 | All classes in the list are discarded without going over every single next classes. | As we can see, the highlighting feature is implemented within the class. Therefore, there should be no impact to or from other classes. For the “HtmlUtilities.highlightString()”, the utility HtmlUtilities class should be discarded. Our change should not cause an impact to a utility class, otherwise it will cause a lot of other impacts. |

**Time spent (in minutes):** 10

# Prefactoring (optional)

Not implemented.

**Time spent (in minutes):** 0

# Actualization

The table below describes each step I followed when changing the code.

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | When I opened the application, and tried the hyper search feature, I saw the result was shown as a tree list in a small separate window, so I added “tree” into the concept location list too. | Replicated the bug and experience the feature to get more familiar with the process. |
| 2 | I suspected that the results are contained within a node list. | Even though this thought is not specific, it still a good clue. |
| 3 | I debugged by writing to file everything related to the list described in Concept Location step 6. |  |
| 4 | I saw that HtmlUtilities.highlightString() is the method that highlight the result strings which are passed as one of its parameters. |  |
| 5 | In the logged file, the variable “s” of the string seems to miss some results. The number of missing results matches with the number of unhighlighted hypersearch results. |  |
| 6 | When debugging deeper into that section of code from line 587 to line 600, I saw that the Match m is assigned to 0 at the end of the while loop that stops it from getting to the next match. Removing it seems to fix the bug. |  |
| 7 | I tested and it worked perfectly as I expected. |  |

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

# Postfactoring (optional)

Not implemented.

**Time spent (in minutes):** 0

# Validation

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

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

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | Test method: test the GUI directly with the running application.  Test case defined: A regular hyper search  Inputs:   * Create the content by copying text from a book into jEdit. * Hyper Search for a letter “a”, so it will be found multiple times.   Expected output: All letter “a” are highlighted. | This is the regular expected behavior.  The test passed. |
| 2 | Test method: test the GUI directly with the running application.  Test case defined: A regular hyper search  Inputs:   * Create the content by copying text from a book into jEdit. * Insert some special words such as “Hello world” 2 times per line for 5 lines. * Hyper search for “Hello world”   Expected output: All 10 matches are highlighted | This is the regular expected behavior.  The test passed. |
| 3 | Test method: test the GUI directly with the running application.  Test case defined: No hyper search match found  Inputs:   * Full fill the text content by letter “a’ * Hyper search for letter “b”   Expected output: No match and nothing is highlighted. | This is the regular expected behavior.  The test passed. |

**Time spent (in minutes):** 10

# Timing

Summarize the time spent on each phase.

|  |  |
| --- | --- |
| Phase Name | Time (in minutes) |
| Concept location | 30 |
| Impact Analysis | 10 |
| Prefactoring | 0 |
| Actualization | 35 |
| Postfactoring | 0 |
| Validation | 10 |
| Total | 85 |

# Reverse engineering

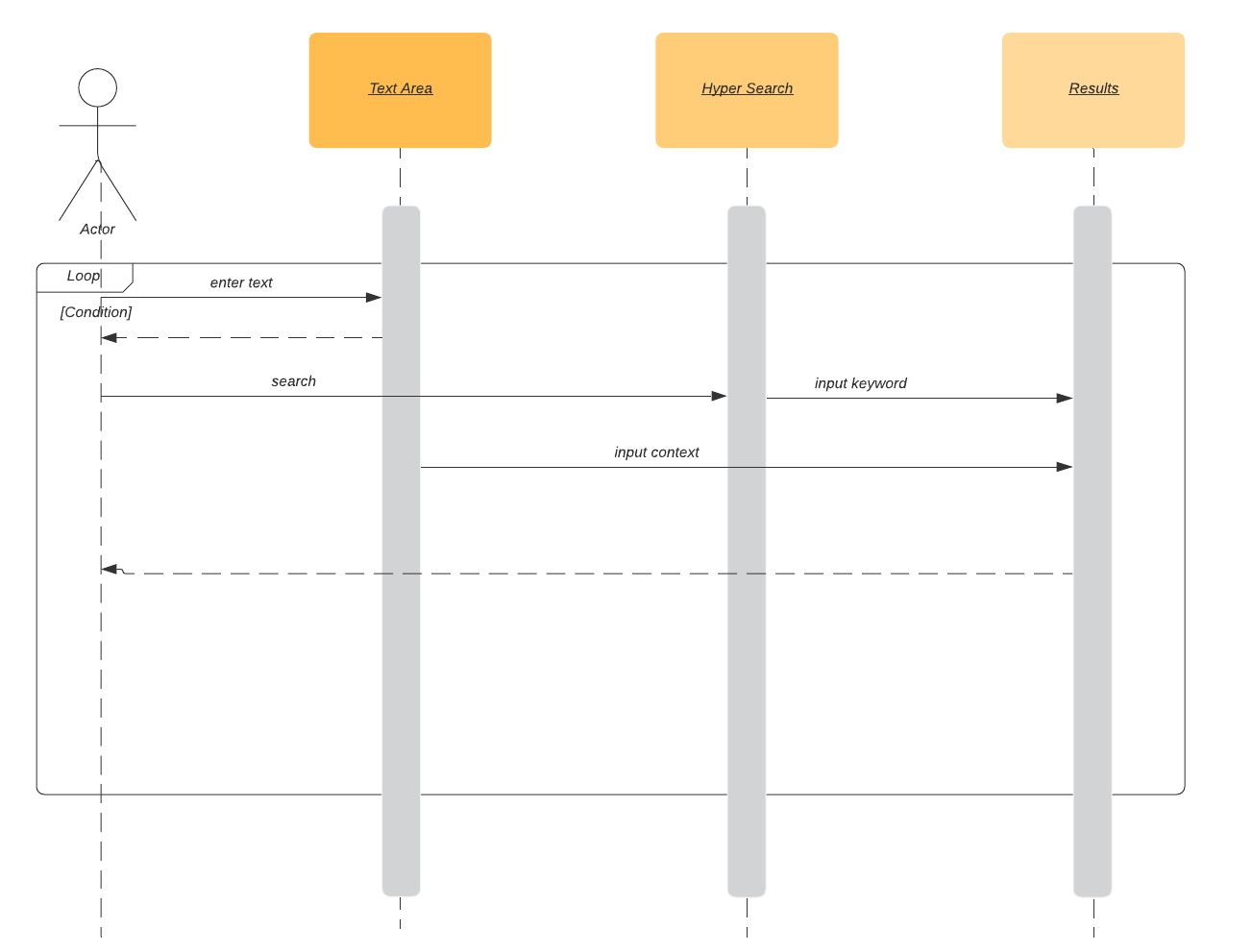


Figure . UML sequence diagram

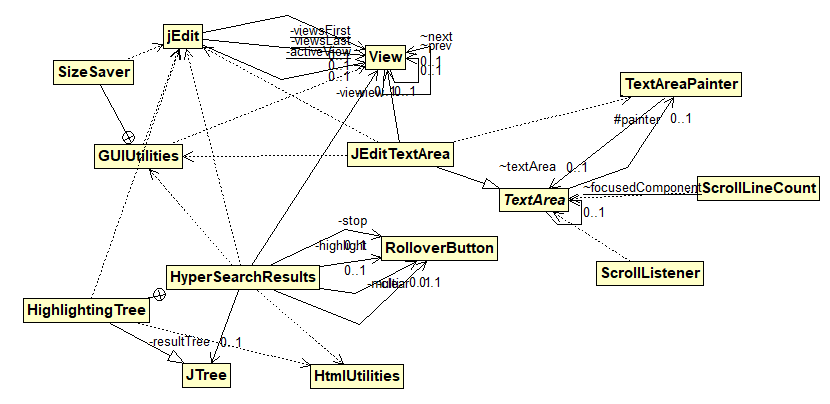


Figure 3. UML class diagram shows the visited classes connecting to the HyperSearchResults class

# Conclusions

The change process has been done within a short amount of time as following the change procedure. Concept location was done by using Eclipse IDE’s file search tool. Impact analysis was done with the support of both file search tool and jRipples. However, impact analysis was not fully accomplished due to the small impact of the marked concept location. The change is only in a local variable named “m” of type Match which is then passed to a utility method highlightString(). Utility method should be discarded.

The classes I have changed:

* inner class HighlightingTree within the HyperSearchResult class