Change Request Log

Concept Location

Step #	Description	Rationale
1	After installation, we ran the code to identify and understand what changes were to be made.	
2	After Running the application, we played around with the different Functionality of the application such as File Browser, File Editing like Undo, Redo, Cut, Copy and paste.	
3	We found the Recent Files Search bar which was located inside the File TAB on top bar.	This was carried out to see what changes occur in the user interface to find any clue to identify what the search in File >> Recent Files >>
4	We searched inside the code base of the key word **Recent Files as** that was what we were dealing with.	Which lead us to jedit_en.props file where constants are stored.
5	In the jedit_en.props file we got to know the property **no-recent-files.label** being used.	From the GUI it was clear that RecentFilesProvider.java was the file which was responsible for rendering the UI for **Recent Files Search**
6	We searched the entire code for the property being used, which was being used in **RecentFilesProvider.java**	
7	JTextField is used as input for search and **addKeyListener** is used to load appropriate UI elements (higlight, etc)	Usually while we are searching, we usually type inside a Text Field
8	Based on regex match UI elements are being updated. But the regex `regex + "*" only does starts with match.`	Usually, regex is used for string match, hence this was easier to notice.
9	Now only thing that was left was to change regex to `"*" + regex + "*" which does sub-string match.`	We confirmed this change does sub-string match to highlight if the string is contained anywhere in the file name

Time spent (in minutes): 25

Classes and Methods inspected

1. org/jedit/localization/jedit_en.props
2. Org/gjt/sp/jedit/menu/RecentFilesProvider.java
   - Update(menu)
     - `text.addKeyListener`

Impact Analysis

Step #	Description	Rationale
1	Using the search all files feature in IntelliJ IDE, we searched where RecentFiles keyword is being used.	We listed all the functions in order to see if there is any function that affects the change we made.
2	We analyzed each of the functions to understand the effect of our change.	We concluded that the change did not affect any other part of the code and there was no need to change the source code any further as a result of our change.

Time spent (in minutes): 9

1. org/gjt/sp/jedit/menu/RecentFilesProvider.java
   - Update(menu)
     - `text.addKeyListener`

Actualization

Step #	Description	Rationale
1	We changed the value regex to "*" + regex + "*"	As the usage of this variable is very limited and in very few places, the value of the variable could be changed without affecting other functionalities.
2	After our change, we ran the code and made a few manual tests.	We made sure the changes we made did not affect other functionalities.
3	Our changed code was pushed into the respective branch.	We thought it was a good idea to push the code to a separate branch just in case.

Time spent (in minutes): 5

1. org/gjt/sp/jedit/menu/RecentFilesProvider.java
   - Update(menu)
     - `text.addKeyListener`
       - `regex = "*" + regex + "*"`

Validation

Step #	Description	Rationale
1	Inspection Case: After the change, we manually checked by opening multiple files on our PC and	The idea behind this is to check if all the change is reflected in different scenarios (i.e.

	tried to sub-string match, making the sb-string from various parts of the filename Expected Output: Sub-String match worked, and respective Files were highlighted.	not only match file names but otherwise also). The test passed.
2	Test case: opening a file with a name having more than 1 word. Inputs: search for the file using different words in the file name. Expected output: the file should be selected if the search value is present anywhere in the file name.	This case was to make sure that the change made works irrespective of where the search term value is. The test passed.
3	Test case: opening multiple files with similar names. Inputs: search value like that of the file name. Expected output: as we have multiple files with the same name all the files with the names should be highlighted.	This case was to make sure that the change made works for multiple files as well. The test passed.
4	Test Case: search for a file that is not recent. Inputs: search value dissimilar to that of file names in recent. Output: None of the files should be highlighted.	This case was to check if the functionality that was present earlier is still unaffected due to our change The test passed

Time spent (in minutes): 17

Summary of the change request

Phase	Time (minutes)	No. of classes inspected	No. of classes changed	No. of methods inspected	No. of methods changes
Concept location	25	1	-	1	-
Impact Analysis	9	1 + 7(props)	-	1	1
Actualization	5	1	1	1	1
Verification	17	-	-	-	-
Total	54	3	1	3	2

Conclusions

The change request was easy when compared to the first one and the fact that we were familiar with the code organization helped us during concept location. As this was a simple change and related to only one **regex value change** to do a sub-string match. Impact analysis, actualization and testing were easier. Concept location, impact analysis and actualization were done using IntelliJ IDE followed by manual testing.