Change request log

1. Team

Driver: Michael Petrey

Navigator: Matthew Starks

2. Change Request

Change request #4 – Create an option on the recent directories menu that will clear the recent directories in the menu.

3. 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 program.*	
2	*We interacted with the program with the intention of identifying what we needed to search for in the explorer*	*This was to become familiar with the changes we had to make as well as to gain a basic understanding of how each one of them operated.*
3	*We found the two submenus in the program and started messing with them to see how they worked before we jumped into the coding.*	*Because we needed a basic understanding of what we were changing*
4	*We searched the source directories to find which one to look for. We determined from a previous change request that the source files that we were looking for were in org/gjt/sp/jedit.*	*Previous experience with the codebase helped locate the files.*
5	*We searched the files in this directory for what would be relevant to the change we needed to make. We knew we needed to find the code for the menu that related to the directories, so we opened the directory "menu."*	*Messing with the program showed us that it the changes we needed to make were in the top menus. The menu directory was the best choice out of the options.*
6	*In the menu directory selected the classes RecentFilesProvider and RecentDirectoriesProvider because we identified that these classes related to the recent menu that we had to edit*	*We selected RecentFilesProvider because it is the menu for the recent files, and it contained the code that deleted the recent files. We selected the RecentDirectoriesProvider class because it was the recent files equivalent for the directories.*

Time spent (in minutes): 60

4. 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	*We looked the code menuItem.addActionListener(e -> BufferHistory.clear()) in the RecentFilesProvider class.*	*This line executes the button that then deletes the recent file history.*
2	*We noticed that the RecentDirectoriesProvider class utilized a HistoryModel vs the BufferHistory class. We then inspected the HistoryModel class.*	*We assumed that RecentDirectoriesProvider would be structured similarly to RecentFilesProvider. It was so we looked at how the directory class stored the recent history.*
3	*We noticed that the HistoryModel class contained the function removeAllElements(). This would be called whenever the button is pressed.*	*We needed to see how to clear data in the History model class*
4	*We saw that the clear recent files button interacted with a prop called "clear-recent-files.label". We found that label by searching in GitHub search. The results came back as in the jedit_en.props which is in the "org/jedit/localization" directory.*	*We wanted to figure out how the labels worked in jEdit. We determined that if we added a prop to the props folder, it should have a small impact on the remainder of the code.*
5	*We figured it should work for the directories if we implemented the clear button in a similar method to how it is implemented in RecentFilesProvider.*	*We finished the impact analysis by finding that if we change the code from the RecentFilesProvider to use the local objects of the RecentDirectoriesProvider, the impact should be minimal.*

Time spent (in minutes): 50

5. 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	*The first change we made was to copy the code from RecentFilesProvider into RecentDirectoriesProvider*	*This is the primary component behind this change request. Next, we would have to restructure this code to fit with the RecentDirectoriesProvider*
2	*We changed BufferHistory.clear() to model.removeAllElements()*	*Model is the HistoryModel object in the RecentDirectoriesProvider class. This code will clear the recent directories when the button is pressed.*

3	*We added a prop to the props folder for the clear recent directories label that is located directly under the prop for the clear recent files label. For this prop we followed the same naming scheme as the other props to ensure consistency.*	*We did this because we needed a new label and we wanted to be consistent with the entire project.*

Time spent (in minutes): 35

6. 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	*Test case defined: The user will open a file on his computer and clear the directories.* *Expected output There are no more recent directories.*	*This is to test clearing one directory to make sure this works.* *The test case passed.*
2	*Test case defined: The user will open multiple files in different folders on his computer and clear the directories.* *Expected output There are no more recent directories.*	*This is to test clearing multiple directories at once.* *The test case passed.*
3	*Test case defined: After clearing the directories, the user will try to clear the directories again* *Expected output: There will be no button.*	*This is to test if there are no current recent directories.* *The test case passed.*

Time spent (in minutes): 20

7. Timing

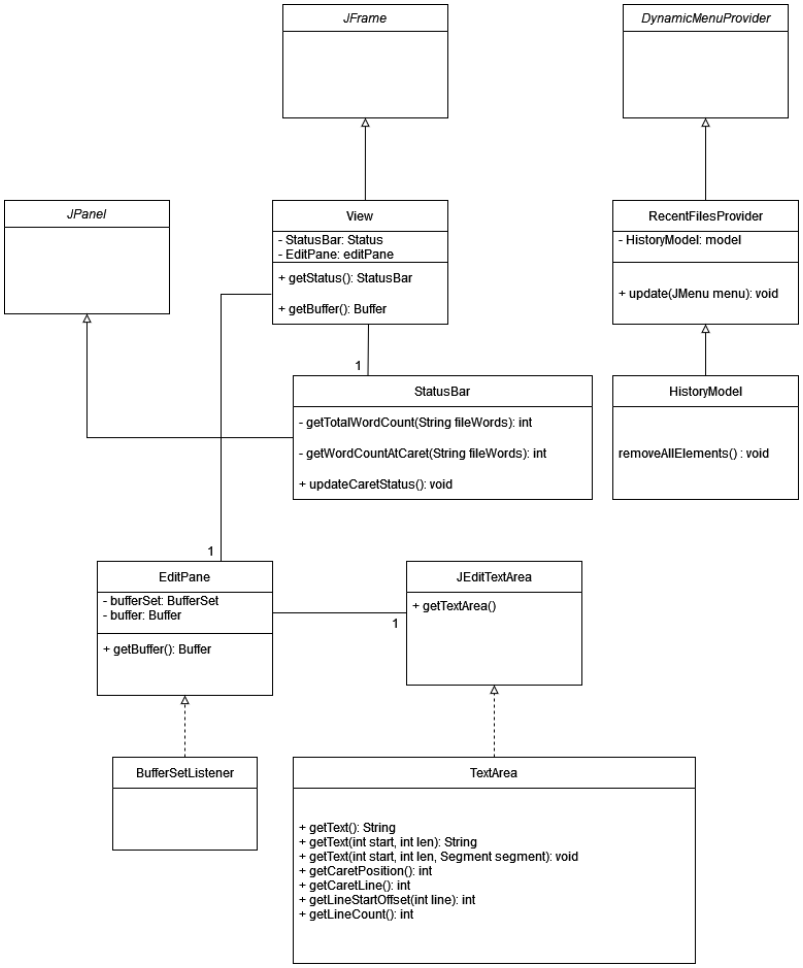
Summarize the time spent on each phase.

Phase Name	Time (in minutes)
Concept location	60
Impact Analysis	50
Actualization	35
Verification	20
Total	165

8. 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.**



9. 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.

This change was simple. The hardest part was to reverse engineer the part of the program that cleared the recent files. Once we understood the underlying structure of the code, we were able to implement the necessary changes in the structure to fit with the objects in recent directories.

Classes changed:

- org/gjt/sp/jedit/menu/RecentDirectoriesProvider.java
    - added code at the very end of the update function
- org/jedit/localization/jedit_en.props

- added a prop for a label