Change request log

# Team

Anurag Kumar and Ahmar Aftab

# Change Request

jEdit Change Request #1: In the File » Recent Files main menu of jEdit, the text box on top of the recent files list allows to highlight recent files names that match with a given string. The string in the text box should match all the files that contain it anywhere in their name. However, the highlight works only when the string matches the beginning of a file name. You are requested to modify this feature so that the highlight occurs for the cases when the string is contained anywhere in the file name.

*Change Owner:* Implemented and documented by Anurag Kumar

# Concept Location

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We build the jEdit* |  |
| 2 | *We ran the system and opened some files to add to the list of recent files.* | *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 the keyword “Recent” using InstaSearch.* | *Because we were looking for class or methods or variables named “\*recent\*”.* |
| 4 | *We checked all the results one by one in decreasing order of the frequency of the keyword until we located the target class.* |  |
| 5 | *After going through the class GeneralOpationPane, EditPane, BufferHistory, etc. we finally located the target class named RecentFilesProvider.* | *The class RecentFilesProvider has access to the list of all recent files. The method keyReleased reads the string and then we create the regular expression to find the files matching the given string.* |
| 6 | *We inspected the class RecentFilesProvider.* | *We found that the variable “regex” is responsible for searching the list of recent files and producing the list of files starting with the given string and thus was relevant to our change request.* |
| 7 | *We added a break point at line 109 of the RecentFilesProvider and ran the program in debugging mode.* | *We did this just make sure that this is where we get the desired result from the list of recent files.* |
| 8 | *We marked the class RecentFilesProvider as “located”.*  *We marked the class StudentGraph as "located".* | *We confirmed this class had to be modified.* |

**Time spent (in minutes):** 150

# Impact Analysis

Since no new feature is being added, therefore it had minimal or no impact on the related systems. However, we did the following impact analysis:

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We looked for dependencies of the class RecentFilesProvider using JRipples.* | *To track the classes that could be impacted by the change.* |
| 2 | *No dependencies found* | *As we did not find any class dependent on the class RecentFilesProvider, thus we just checked the class itself for any ripple effects but did not find any impact.* |

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

# Actualization

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *We changed the regular expression “regex” in the class RecentFilesProvider.* | *We realized that by just changing the regular expression in the same code, we can search for all the files that matches the given string.* |

**Time spent (in minutes):** 15

# Validation

|  |  |  |
| --- | --- | --- |
| Step # | Description | Rationale |
| 1 | *Test Case 1: open output.txt, input.txt*  *Inputs: “put”, “in”, “out”*  *Expected outputs: output.txt, input.txt*  *input.txt*  *output.txt* | *This is the regular expected behavior.*  *The test passed.* |
| 2 | *Test Case 2: open Path planning algorithm using A\*, path planning algorithm using Dijsktra, Path planning using Mobile robots in 3D world*  *Inputs: “path”, “mobile”, “algo”*  *Expected outputs:*  *Path planning algorithm using A\*, path planning algorithm using Dijsktra, Path planning using Mobile robots in 3D world*    *Path planning using Mobile robots in 3D world*  *Path planning algorithm using A\*, path planning algorithm using Dijsktra* | *This is the regular expected behavior.*  *The test passed.* |

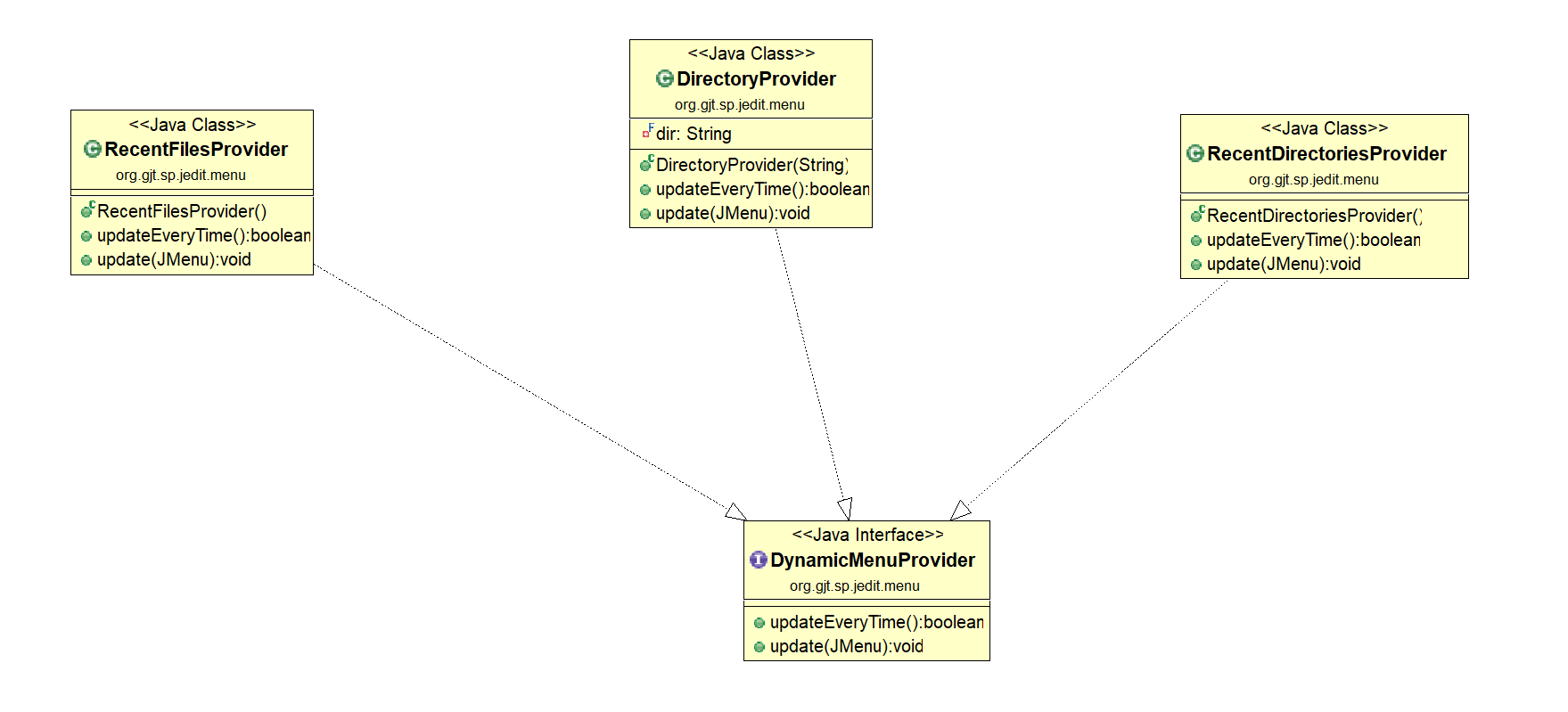
**Time spent (in minutes):** 20

# Timing

Summarize the time spent on each phase.

|  |  |
| --- | --- |
| Phase Name | Time (in minutes) |
| Concept location | 150 |
| Impact Analysis | 35 |
| Prefactoring | - |
| Actualization | 15 |
| Postfactoring | - |
| Verification | 20 |
| Total | 220 |

# Reverse engineering



A partial UML class diagram of the classes visited while navigating through the code.

# Conclusions

*This change request even though it appeared quite simple at first but posed many challenges while implementing. Even though this is a small-scale software but due poor documentation it took some time to just build the software. As we were new to software maintenance, we had no idea about where to start, thus it took us almost a day to just get familiar with the software. For concept location, we used InstaSearch to search for keywords and we used JRipples for impact analysis and change propagation. We did all the testing manually.*

*Classes and methods changed:*

* *org/gjt/sp/jedit/menu/RecentFilesProvider.java/RecentFilesProvider*
  + *public void keyReleased(KeyEvent e)*