Change request log

1 Team

Evolutionary Rams:
Nada Alalyani
Sanket Mehrotra

2 Change Request

ID	Description
#je-1: modify the status bar;	add word offset and total words in file.

3 Concept Location

In the table below we describe each step we followed when performing concept location for this change request. In our description, we include the following information when appropriate and indicate is as follows in the below table:

HyperSearchRequest

3.1 je#1: Adding word total count and offset change request

Step #	Description	Rationale/Discovery
1	We ran the system	
2	We interacted with the system: we opened a file to understand the working of the current status bar functionality	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 "Status*Bar" using the regular expression feature of the IDE's search tool.	Because the change was to be made in a part of the tool called the Status bar.
4	There were no results	
5	We searched for the main function the file that runs first. Went through jedit.java	We wanted to find where the app begun. To start debugging it. Discovered main thread and splash screen code, and execution with additional command line arguments handling code.
6	We inspected the package org/jgt/sp/jedit/gui.statusbar We went to this class using the dependency navigator of the IDE editor.	We noticed that the method package contained factory classes for various other plugins.

7	Checked these plugins on the UI	We wanted to explore the intended functionality and check the tooltips to understand how each worked
8	Went through BufferSetWidget class, ErrorSetWidget class and other classes in org.jgt.sp.jedit.gui.statusbar	Wanted to find code related to the caret position/column offset OR onkeychange events
9	Checked the org.gjt.sp.jedit.textarea to find code related to key changes	
10	Search for "Offset" using Search -> File Search window	Found several matches in project(2604)
11	Checked matches in LineManager.java	Turned out to be for the emacs macro utility, not what we need.
12	Checked matches in TextArea.java	
13	Checked out matches in org.jedt.keymap	
14	Searched for caret related functions/methods in TextArea.java. Focused search on org.gjt.sp.jedit.textarea	Found several caret related methods.
15	Breakpointed org.gjt.sp.jedit.textarea .getCaretPosition and org.gjt.sp.jedit.textarea .getCaretLine and EditPane.saveCaretInfo	We Suspected that this definitely had something to do with the caret position displayed on the UI.
18	Found out that the getCaretPosition in the TextUtils class returns the right line number. Now to find the code where its is displayed	While debugging came across the textarea.magic* functions
19	Found that getLineStartOffset() returns right the character offset that's displayed on the UI. Now to find where it is being displayed.	
20	Checked all calls of the function getCaretPosition()	
21	Found org.gjt.sp.jedit.gui.StatusBar.java and its function updateCaretStatus()	The description describes it as the right function.
22	Discovered that properties are read via a call to getBooleanProperty()	It is defined in jEdit.java(the main java class)
23	Trailing getProperty() leads to org.gjt.sp.jedit.Propertymanager.java	Leads to Properties.class which is a java class. So its nothing we can/should touch.

24	Back to the PropertyManager.java file, follow the call hierarchy of the function loadSystemProps	to find where to define our new setting.
25	Found initSystemProperties function	Located a path to load system properties. • `/org/gjt/sp/jedit/jedit.props` • `/org/gjt/sp/jedit/jedit_gui.props` • `/org/jedit/localization/jedit_en.props`
26	Added two new settings in jedit_gui.props: • view.status.show-word-offset • view.status.show-word-totalcount	To read the settings into the application via the system settings.
27	Read the two new settings and check their values in the updateCaretStatus() function	To check whether either one is false.
28	Concept Location complete: First method to modify: updateCaretStatus present in the file /org/gjt/sp/jedit/gui/StatusBar.java	

Time spent (in minutes): 130 mins(9:47pm)

4 Impact Analysis

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

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 detail 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	Any change to our finalized class and its method • Class: StatusBar ○ void updateCaretStatus() may affect the following methods:	Checked *call hierarchy* of the addInput() function: • Class: org/gjt/sp/jedit/View/CaretHandler/ ○ void caretUpdate(CaretEvent evt) • Class: org/gjt/sp/jedit/EditPane ○ void.handleBufferUpdate(BufferUpdate msg) • Class: org/gjt/sp/jedit/View/ ○ void handleEditPaneUpdate(EditPaneUpdate msg)

		• Class: org/gjt/sp/jedit/gui/StatusBar ○ void propertiesChanged()
2	Similarly checking the call hierarchy of the class StatusBar	• Class : org/gjt/sp/jedit/View as part of the constructor
3	Since this is a void function, there are no calls of this function where there needs to be an impact analysis of the modified output.	
4	Of the UI changes made to the class: StatusBarOptionPane.java	They are all made inline with the other UI changes, the only effect they have is on the Menu UI. Also they are configurable so it is possible to hide these changes via the properties file.

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.

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

Je#1: Word Offset field addition.

Step #	Description	Rationale
1	Function: updateCaretStatus() We had to add to the current functionality of (caret offset/ total number of characters) Experimented with adding dummy values in the status bar buffer object	To see if the changes would reflect on the UI
2	Calculated the total number of words in the text file by reading all the text in the buffer and splitting by space and new line character	To get total number of words
3	Calculated the current word offset by reading the text buffer until the position of the caret and splitting by new line characters and space.	To obtain word offset

4	Added settings for our change in the jedit-gui.props property file	To read in the code and toggle visiblilty
5	Read properties via the getBooleanProperty function and appended to the status bar buffer object accordingly	To make the change configurable.
6	Added two new checkboxes in the GUI class org/gjt/sp/jedit/options/StatusBarOptionPane.java	To create the GUI objects in the menu.
7	Added all required gui object code: such as initialization and reading default values from the properties file.	To make sure the GUI components work.

Time spent (in minutes): 120 mins

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.

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

Step #	Description	Rationale
1	*Test case defined: Open new text document.* *Inputs: Text document and in jedit-gui.props:* • `view.status.show-word-totalcount:true` • `view.status.show-word-offset:false.` *Expected output: Total number of words seen on the status bar*	*Total number of words visible in the status bar next to the caret offset field for normal text document.* *This is the regular expected behavior.* *The test passed.*
2	*Test case defined: Add new words into an open text document.* *Inputs: Text document and in jedit-gui.props:* • `view.status.show-word-totalcount:true` • `view.status.show-word-offset:false.` *Expected output: The total number of words increases to reflect the new entered text.*	*The total number of words increases in the status bar next to the caret offset field to reflect the new entered text.* *The test passed.*

3	*Test case defined: Open new text document.* *Inputs: Text document and in jedit-gui.props:* • `view.status.show-word-totalcount:false` • `view.status.show-word-offset:true.` *Expected output: Current caret position based word offset seen on the status bar.*	*The current word offset of the caret from the beginning of the file is shown in the status bar next to the caret offset field.* *Test Passed.*
4	*Test case defined: Add new words into an open text document.* *Inputs: Text document and in jedit-gui.props:* • `view.status.show-word-totalcount:false` • `view.status.show-word-offset:true` *Expected output: The word offset increases to reflect the new entered text*	*The word offset increases in the status bar next to the caret offset field.* *Test Passed.*
5	*Test case defined: Open a text document.* *Inputs: Text document and in jedit-gui.props:* • `view.status.show-word-totalcount:true` • `view.status.show-word-offset:true.` *Expected output: both word offset and total word count are visible in the status bar next to the caret offset field.*	*Both word offset and total word count are visible in the format(word offset/ total number of words)in the status bar next to the caret offset field.* *Test Passed.*

Time spent (in minutes): 50

7 Timing

Summarize the time spent on each phase.

Phase Name	Time (in minutes)
Concept location	100
Impact Analysis	50
Actualization	120
Verification	50
Total	320

8 Reverse engineering

- **Class Diagram**

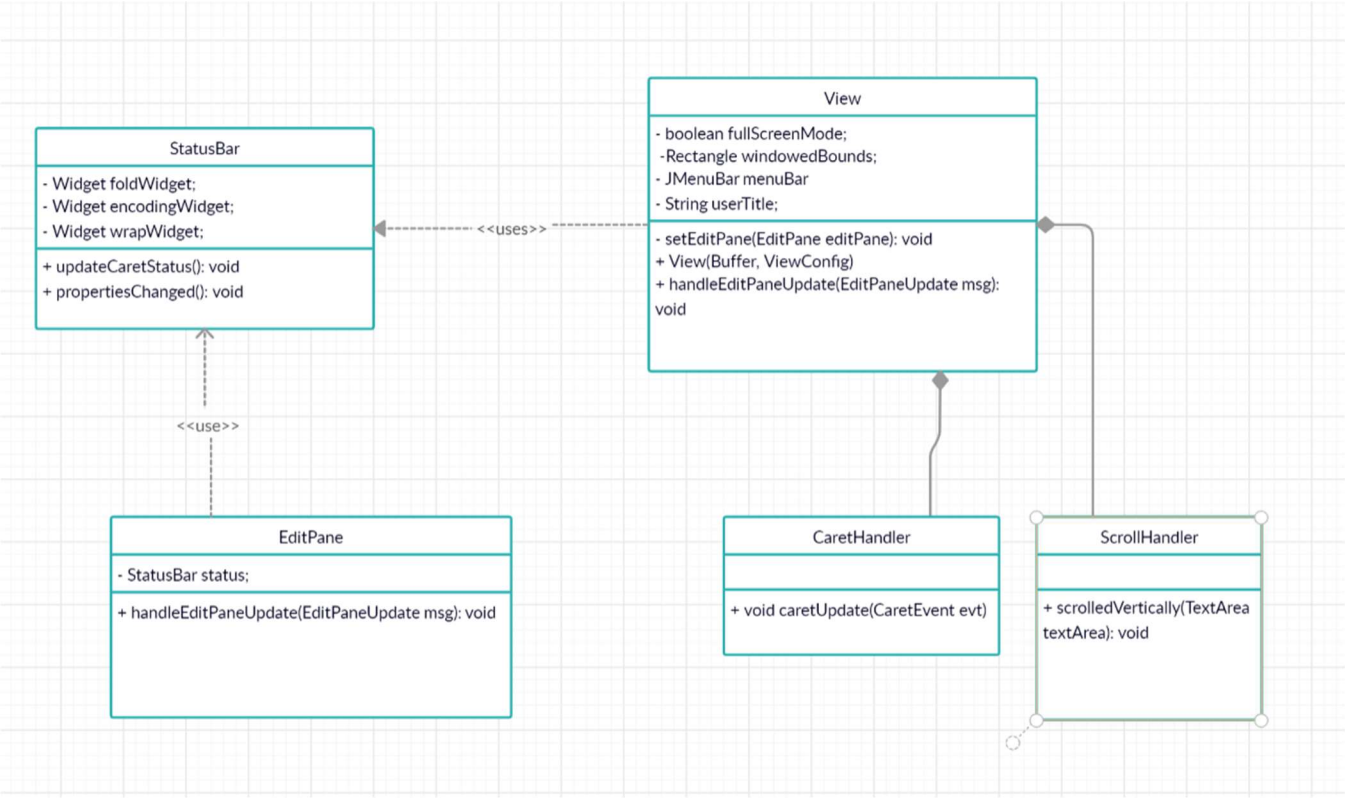

Figure 1: Class diagram for je#1. Relevant classes and Methods

- **Sequence Diagram**

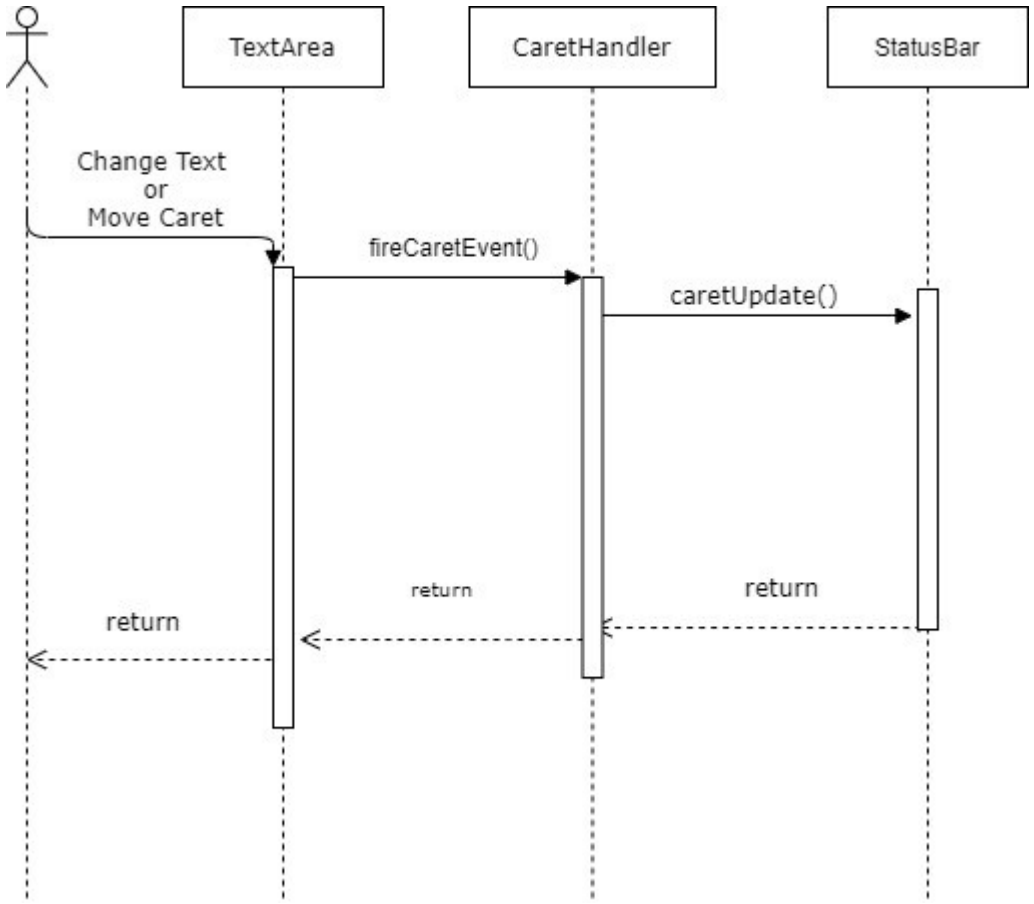

Figure 2: Sequence diagram for je#1.

9 Conclusions

For this change, concept location was a little challenging because there were a lot of caret related functions in the search results across the project. But after debugging the code and noting the flow of control, we managed to find the change. Designing validation test cases for this change was not too hard either. Actualization of the change required learning a little more about the functions used to return text buffers and manipulating the text. Making the change in the menu required its own concept location, reading from the properties file was a neat trick we learnt and will probably use for future projects. But no impact analysis was required since it was a single responsibility class and the addition of GUI elements wouldn't effect any other members.

Classes and methods changed:

- org/gjt/sp/jedit/gui /StatusBar.java/
    - void updateCaretStatus()