Change request log

1 Team

Evolutie – Jason Stock (Document Specialist) and Kevin Bruhwiler (Developer)

2 Change Request (#je-2)

Currently, jEdit displays the horizontal and vertical scroll bars whenever the content of the opened document exceeds the size of the editor. Implement a **Toggle Scroll Bars** option in the **View** menu that allows to show/hide the scrollbars.

These changes can be seen on GitHub under the **stock-dev** branch:
https://github.com/stockeh/cs515-001-s20-evolutie-jedit

3 Concept Location

The following table describes the steps followed when performing concept location for this request. This includes various techniques, such as IDE features used, search queries, system executions, classes visited, etc.

Step #	Description	Rationale
1	Opened the project in Eclipse	
2	Went to the View class within the top level `org.gjt.sp.jedit` package. This class contains `ScrollHandler` class that could be a part of the scrolling functionality.	Told to implement "toggle scroll bars" in the view menu and thought this would be a good starting location.
3	Right clicked the method definition "`scrolledVertically()`" and selected Call Hierarchy within the Eclipse IDE. Followed the method class to the `TextArea` class.	It is possible that the text area class was responsible for organizing the content in the text area, e.g., the vertical/horizontal scroll bars.
4	Searched the constructor of the `TextArea` class and found the vertical/horizontal box `JScrollBar` from `JPanel`.	The constructor seemed to be the most logical entry point to the class.
5	`TextArea` is where the scroll options exist and where the components are added to the UI. We marked this class as located.	This class seems to be the location where the scrollbar can be added or removed from the text area.
6	Used the Eclipse IDE InstaSearch package is to search for the String "Toggle Line Numbers". There were a number of results displayed, including an XML file named `actions.xml` with further functionality controlling the state of the line numbers This file has been marked for as located.	Utilized the UI to view what other View options are available and saw that the feature functionality will be close to that of "Toggle Line Numbers".

7	Located the actions.xml file where some properties are updated based on the current action, including the `view.gutter.LineNumbers` property. We marked this xml file as located.	It is very possible that the view scroll options will need a similar action.

Time spent (in minutes): 61 minutes

4 Impact Analysis

The table below describes each of the steps performed during impact analysis for this request. This includes navigating the set of classes that are related to the located class and marking it as unchanged, changed, or propagated depending on the estimated impact.

Step #	Description	Rationale
1	Started with the `actions.xml` file and continued to identify the actions of the "Toggle Line Numbers" feature. Saw there were additional properties that these were setting named view.gutter.linenumbers and view.gutter.enabled. Used Eclipse IDE Search to File capabilities to see where they are referenced. And marked the `jedit.props` within the `org.gjt.sp.jedit` with interest to change as well.	
2	Need to update the localization properties within the package `org.jedit.localization` as well as the default `jedit.props` within the `org.gjt.sp.jedit` to have a configurable state	By updating the `TextArea` and the View Options Pane there needs to be a way to dynamically update the state of the `TextArea`
3	Need to update the `actions.xml` to update the state property for the action	
4	Need to update the `jedit_gui.props` with the name of the action to render the item in the View menu.	
5	Need to update the text area class to reload when the properties change	

Time spent (in minutes): 80

5 Prefactoring (optional)

The table below describes each of the steps performed during perfactoring for this request. This includes any refactoring operations and objects or methods that are modified renamed or moved. It is possible that the scope of the change is small enough that no prefactoring is needed.

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

Step #	Description	Rationale
1		

Time spent (in minutes): x

6 Actualization

The table below describes each of the steps performed during actualization for this request. Steps conducted during this process describe the changes to code, e.g., classes or methods added removed or modified.

Step #	Description	Rationale
1	Added methods for adding and removing vertical and horizontal scroll bars in the TextArea class	We need a way to remove the JPanel box from the scroll layout view
2	Added a default property to jedit.props named view.textarea.scrollbars	Need to have an initial value that can be saved and loaded from in the TextArea class
3	Added property named “toggle-scroll-bars” to localization properties jedit_gui.props	Needed a checkbox tied to an action within the view menu
4	Added component to actions.xml called “toggle-scroll-bars” to fire the action when the view menu is de/selected	Need a way to update state when the checkbox is selected from the menu bar
5	Added a name for the checkbox in the file `jedit_en.props` within the localization package `org.jedit.localization`	A name has to be assigned to the view menu action

Time spent (in minutes): 90

7 Postfactoring (optional)

The table below describes each of the steps performed during postfactoring for this request. This includes any refactoring operations and objects or methods that are modified, renamed or moved after actualization. It is possible that the scope of the change is small enough that no postfactoring is needed.

Step #	Description	Rationale
1		

Time spent (in minutes): 0

8 Validation

The table below describes each of the steps performed during validation for this request. This includes steps after actualization such as testing or code inspections.

Step #	Description	Rationale
1	Manual functional test 1. Build and run the project with Ant 2. Open the View menu 3. Deselect the “Show Scroll Bars” 4. Add text to the text area to trigger scroll bars to show 5. Verify that the scroll bars do not appear It is expected that the scroll bars do not appear when the text begins to exceed the page limit.	The scroll bars are deselected and should not appear regardless of the text
2	Manual functional test 1. Build and run the project with Ant 2. Open two files in the UI that are large enough to trigger both vertical and horizontal scrolling 3. Verify that the scroll bars are present for each file opened 4. Split the pane vertically using the “Split Vertically” button 5. Open the View menu 6. Deselect the “Show Scroll Bars” 7. Add text to the text area to trigger scroll bars to show 8. Verify that the scroll bars do not appear for either of the opened files It is expected that the scroll bars do not appear when the text begins to exceed the page limit regardless of the number of open files	Multiple views could exist, and the scroll bars should still be able to be de/selected for each view.
3	Manual functional test 1. Build and run the project with Ant 2. Open the View menu 3. Deselect the “Show Scroll Bars” 4. Close jEdit ensuring it is completely shutdown 5. Reopen jEdit and verify that the scroll bars still do not exist and the option with the view menu is still deselected. It is expected that the previous option set within the view menu remain over multiple sessions.	The session should be saved and reloaded over multiple uses of the application. i.e., if the option to show scroll bars is selected, then it should remain selected and vise-versa

Time spent (in minutes): 20

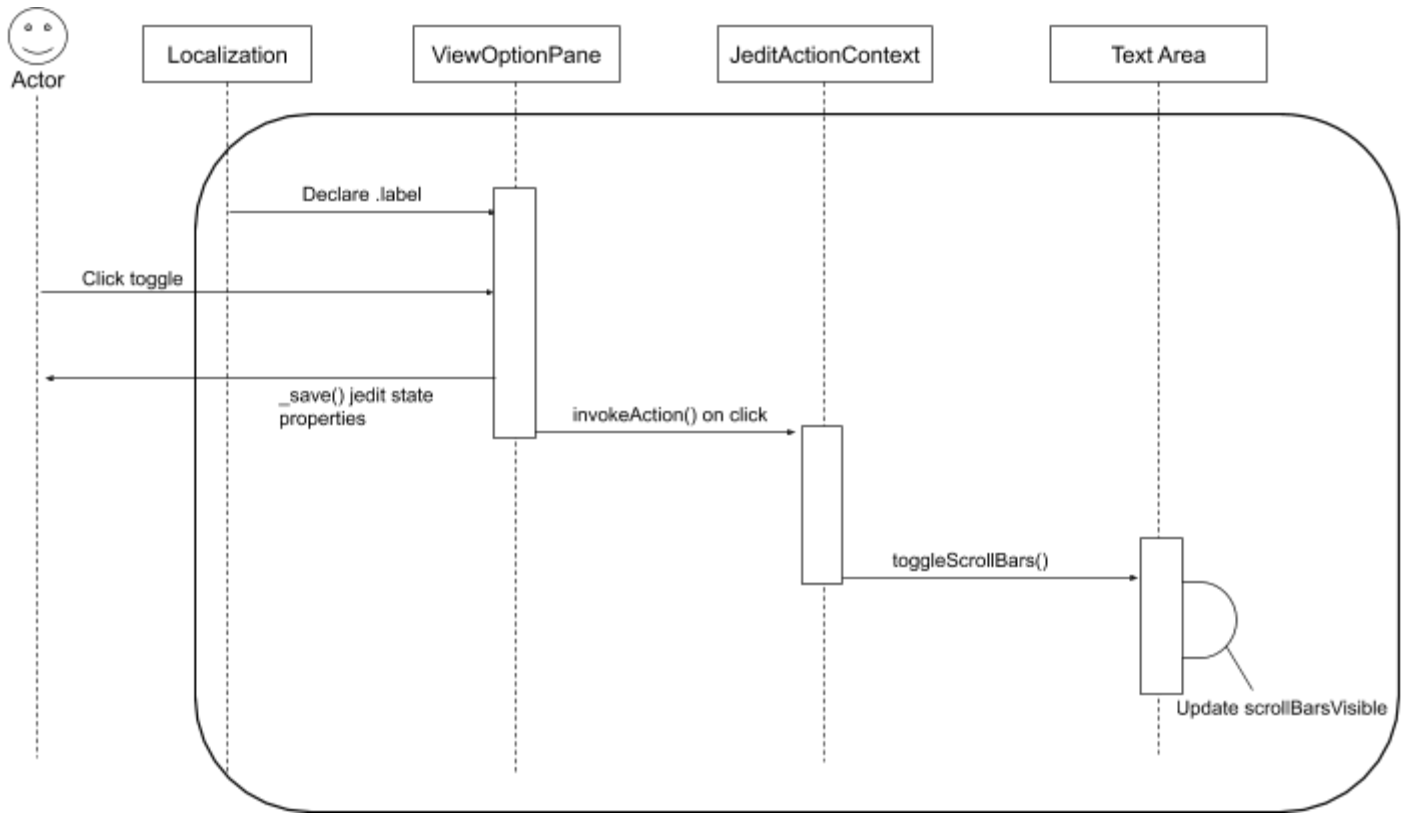
9 Timing

Summarize the time spent on each phase.

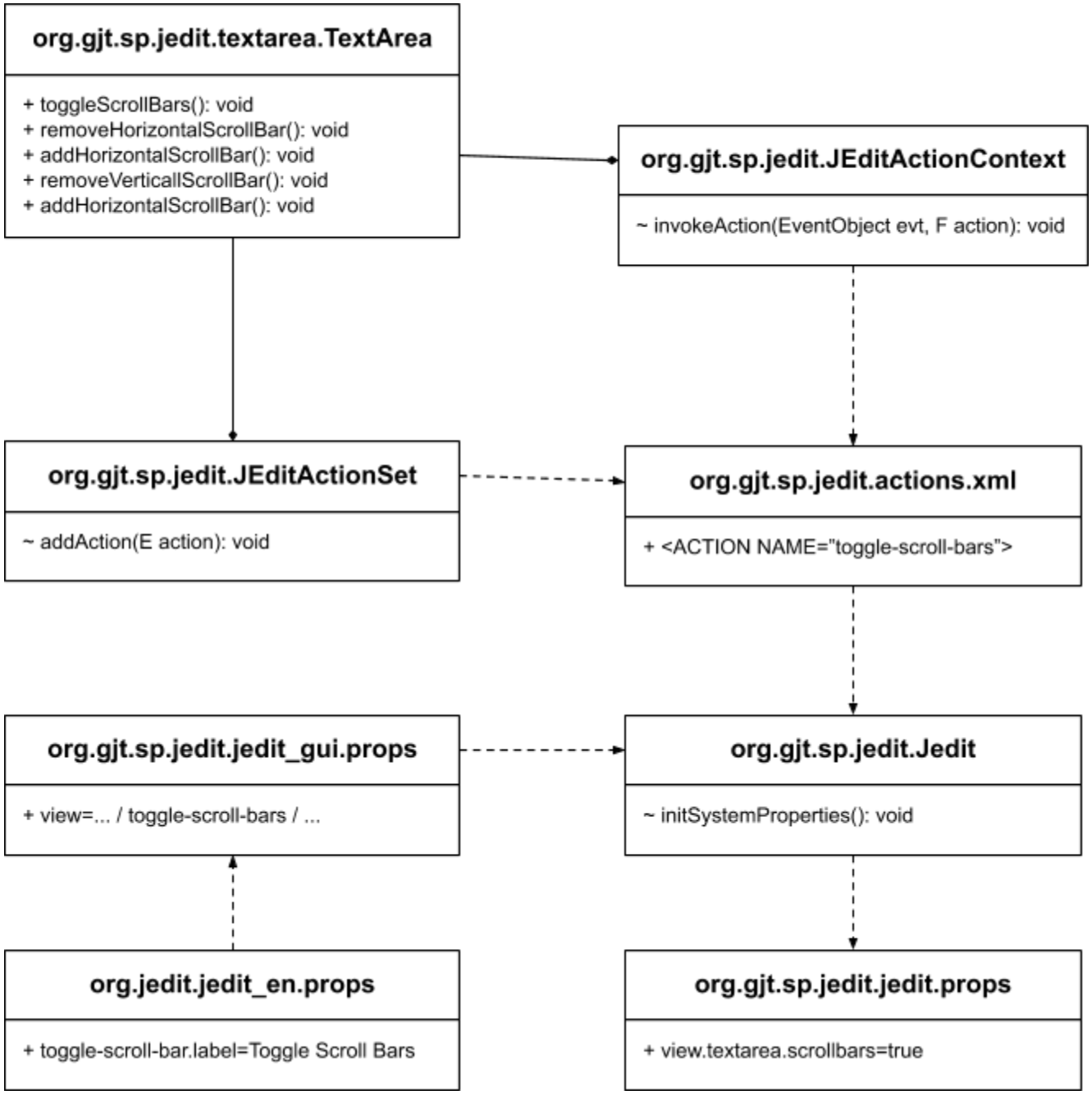
Phase Name	Time (in minutes)
Concept location	61
Impact Analysis	80
Prefactoring	-
Actualization	90
Postfactoring	-
Verification	20
Total	251 (4.18 hours)

10 Reverse engineering

Create a UML sequence diagram (or more if needed) corresponding to the main object interactions affected by your change.



A UML class diagram of the visited classes while navigating the code in concept location. This includes the associations between classes (e.g., inheritance, aggregations, compositions, etc.).



11 Conclusions

Identifying concept location was very difficult for this feature request due to the fact that jedit's "action flow" depends on a number of .xml files without clear class or function links. For example, finding where the files that define the gui were located was very difficult due to the fact that it's a field simply named "view" in an .xml file. Other complications resulted from the existence of a number of "action" files, only one of which ultimately needed to be modified, without any clear documentation as to which actions file was responsible for which parts of the GUI.

Once it was clear how to add buttons to part of the GUI and how actions are called automatically when buttons are clicked, determining how to toggle the scroll bars in the text area was very straightforward, and could be contained entirely within the TextArea class.

Classes and methods changed:

- org.gjt.sp.jedit.textarea.TextArea
    - void toggleScrollBars()
    - void removeHorizontalScrollBars()
    - void addHorizontalScrollBars()
    - void removeVerticalScrollBars()
    - void addVerticalScrollBars()