Change request log

___Team
Max Roselius

___Change Request
#2. This change request seeks to add an option to the View menu that allows the user to hide the scroll bars for large texts.

___Concept Location

Step #	Description	Rationale
1	*Examined the current View menu.*	*To get familiar with the existing options, and find a logical place to add a new menu item to allow hiding scroll bars.*
2	*Search workspace for "Scroll"*	*Begin to narrow search to relevant areas*
3	*"Scroll" is too broad. Search for "scrolling".*	*"Scrolling" is the exact button label in the view menu, so hopefully this can further narrow the search space.*
4	*Scrolling was not an effective search term. Re-examined View menu buttons, searched for Toggle Line*	*Should find some sort of definition for the button, or at least something related to that button*
5	*In jedit_en.props, found `toggle-line-separator`. Do a new search for "toggle-line"*	*Further refining the search space*
6	*Found several relevant files, namely jedit_gui.props and actions.xml. I am unfamiliar with the way the authors map buttons with text files, so try some experimental changes here to see how they compile. Add new label in jedit_en.props, and a corresponding button tag in jedit_gui.props.*	*Need to see how changes in these files affect the text editor.*
7	*Recompile with new changes, examine text editor. Find that a new button is added to the menu, but with no functionality. Search through actions.xml to see how other buttons are mapped to actions.*	*This file should have some hints at which classes are used to change views and update variables.*
8	*See that many other buttons rely on jEdit.getBooleanProperty() (or getXXXProperty()). Examine how this class reads properties.*	*Need to see how to reflect global property changes.*
9	*See that I need to add a new property to determine the toggled state of scroll bars. Note that the view class will likely have some sort of state that must be modified. Examine View class and search for JScrollBar instances.*	*Need to locate where JScrollBar is defined and determine how to modify its visibility.*
10	*Find no scrollbars here, search entire workspace for JScrollBar definitions*	*Find a class that maintains a scroll bar, and which can be somehow linked to a View instance*
11	*Several locations have JScrollBar instances, noticed that textArea creates a vertical and horizontal scroll bar. Check View class for instances of textArea*	*If View maintains textAreas, this would likely be the location for changes.*
12	*Search View class for "textArea". Find that it*	*Not familiar with many GUI components,*

	doesn't maintain direct textAreas, but a set of EditPanes, which contain instances of textAreas. Check to see if JScrollBar instances can be made invisible.	*need to see how to alter them.*
13	*Find that JScrollBar instance can be set to invisible, concept location has been found.*	*I just need to update a given View's set of EditPanes whenever the button is selected.*
14	*Mark View, TextArea, actions.xml as located.*	

Time spent (in minutes): 140

Impact Analysis

Step #	Description	Rationale
1	*Inspect the JScrollBar instances in TextArea to see their usage. Search online for the effect of setting their visibility to false.*	*As long as visibility does not affect other functionality (mouse scroll wheel, arrow up/down, etc.), this change should not further impact these instances.*
2	*According to JScrollBar documentation, changing only the visibility will not impact other functionality, so TextArea class will not require changes besides altering scroll bar visibility on request.*	
3	*Examine the View class, search for methods of obtaining its set of instances of EditPanes.*	*Need to be able to access every EditPane managed by a given View instance to alter their scroll bars. Found a method that does just that. Will need to add methods for iterating through these instances and changing visibilities.*
4	*Examine actions.xml to see how other buttons are implemented.*	*This file dictates the action of a button click, so I need to see examples to determine if changes need be made elsewhere.*
5	*Find that I will need to add a variable to jedit.props and change this value upon button clicks.*	
6	*Determine that jedit.props is last location to be affected, can begin to implement changes.*	

Time spent (in minutes): 45

Actualization

Step #	Description	Rationale
1	*Add two methods to TextArea to hide and unhide the horizontal and vertical scroll bars.*	*This is the class that manages scroll bars for the text area of jEdit, and the only functionality of the bars that need be changed is their visibility.*
2	*Add two new methods and a class variable to View. These methods allow to iterate through all instances of EditPanes managed by that View and hide or unhide their textArea scroll bars.*	*No need to add new classes, just need to modify each EditPane of the selected View.*
3	*Modify actions.xml to call the newly defined*	*This file defines the methods that get called*

	functions when the GUI button is selected. Also add simple logic to check current state and perform the correct action based on that.	*when the button is selected.*
4	*Add a new variable definition in jedit.props that tracks the current state of the scroll bars (hidden or unhidden) with a boolean value.*	*Actions.xml logic for this button requires knowledge of the current state to perform the correct action (hide or unhide).*
5	*Recompile and perform quick test for functionality.*	*Do a quick simple test to ensure the logic is correct.*
6	*Receive error from runtime, look at stack trace and try to locate source of problem.*	*Need to find origin of error that stemmed from new changes.*
7	*Error trace points to actions.xml, check it multiple times before figuring out a semicolon was left out on one line. Fix error and recompile, perform test that was attempted earlier.*	*Quick test succeeds, changes (hopefully) complete.*

Time spent (in minutes): 30

Validation

Step #	Description	Rationale
1	*Open a single pane in jEdit (one document). Toggle to hide scroll bars. Expect bars to disappear. Toggle back on, expect to reappear.*	*Simple test case for only one document (EditPane). Test passed.*
2	*Open several documents, toggle to hide scroll bars. Scroll bar should remain hidden when switching between documents. Toggle back on after checking each document, and check each document again.*	*Need to ensure that the toggle is maintained for all documents, not just the current one. Test passed.*
3	*Toggle off scroll bar and click along area where scroll bar existed. Document should not move.*	*Ensure that changing visibility also prevents clicking action. Test passed.*
4	*Toggle scroll bar off and then back on, attempt to click scroll bar to move it. (Bar itself as well as the up/down buttons).*	*Need to ensure functionality is correct once toggled back to visible. Test passed.*

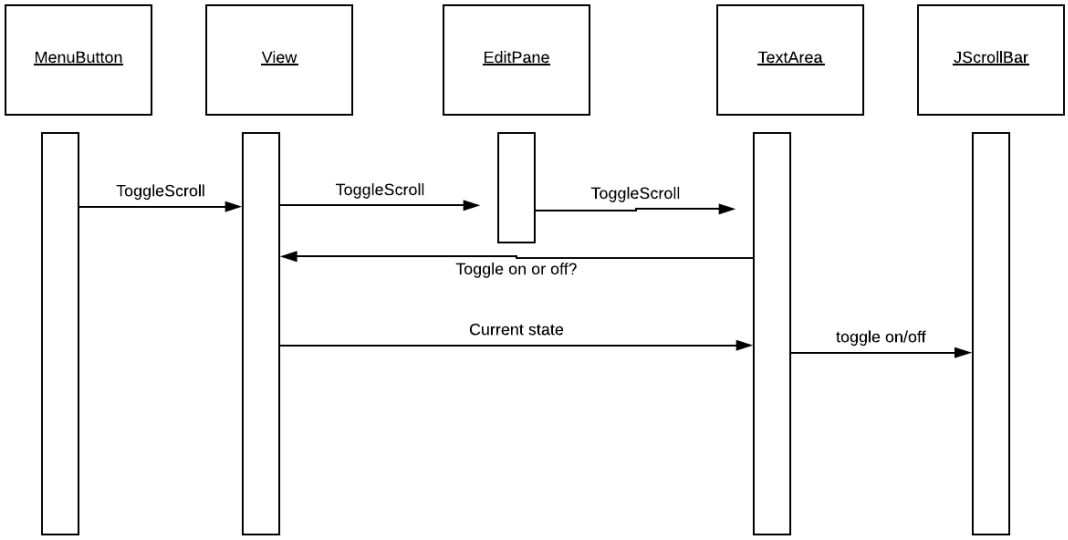
Time spent (in minutes): 10

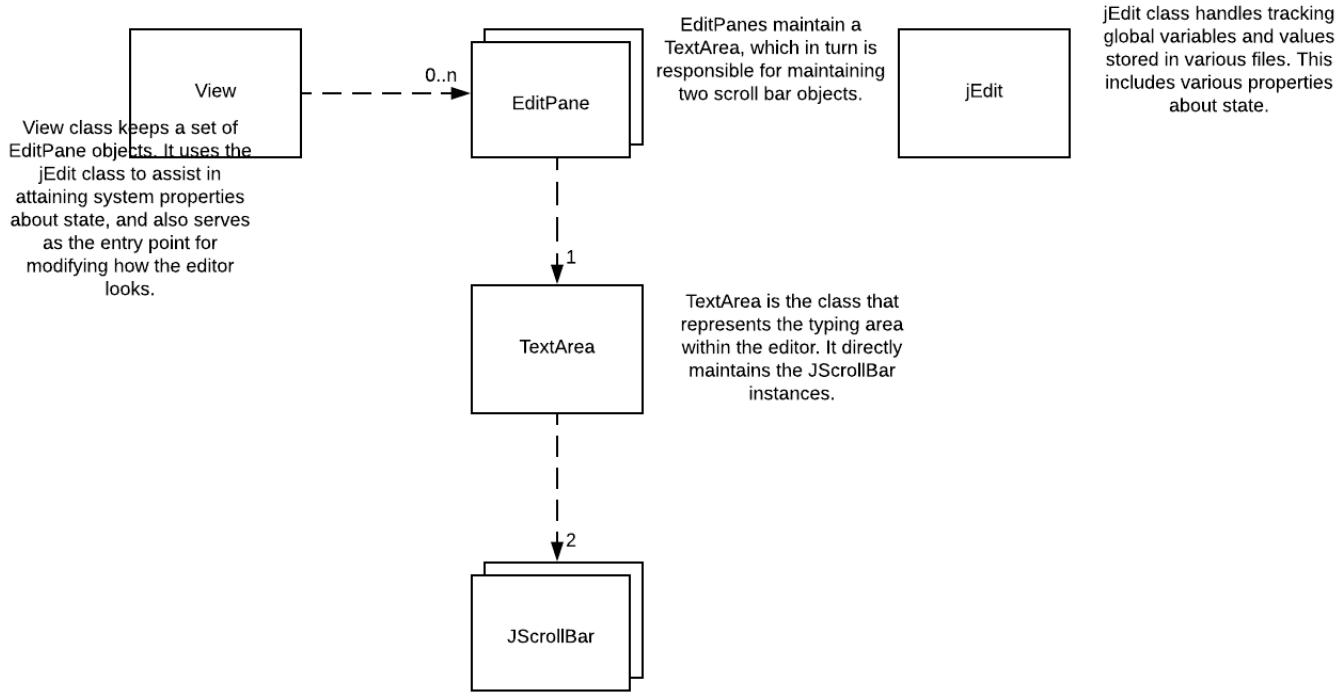
Timing

Summarize the time spent on each phase.

Phase Name	Time (in minutes)
Concept location	140
Impact Analysis	45
Prefactoring	NA
Actualization	30
Postfactoring	NA
Verification	10
Total	225

Reverse engineering





___Conclusions

This change was rather difficult for me. Mostly this is due to my unfamiliarity with GUIs. Concept location was more difficult than in the other changes for this assignment due to needing to locate multiple places for changes to be made and also to determine how buttons were mapped to the GUI and then actions mapped to those buttons through separate files. There are likely some steps I accidentally left out of the concept location description due to some frustration and hastily searching through files looking for something relevant to the concept. The time is accurate, but I failed to accurately record each step along the way and had to reconstruct many from memory. The actual change took a fair amount of time as well, thanks to a missed semicolon in the actions.xml that wasn't caught at compile time. I ended up investigating several other classes that I saw in the stack trace but ultimately were not relevant to the error. After completing the change, I have gained a much better understanding of GUI implementations.

Classes and methods changed:

- *org/gjt/sp/jedit/textarea/TextArea.java*
  - *hideScrollBars(), unhideScrollBars()*
- *org/gjt/sp/jedit/View.java*
  - *hideScrollBars(), unhideScrollBars(), class variable scrollShowing*
- *actions.xml*
- *jedit_gui.props*
- *jedit_en.props*