Capstone Courseware, LLC
1 Washburn Place
Brookline, MA 02446
877-227-2477
capstonecourseware.com

Coding Exercise: Undo/Redo

In this exercise, you will build support for undo and redo functions in a text editor. We won't focus on the GUI parts of this – just the back-end support.

Assume that an application that uses your system holds editable text, very simply, as a **String**. Your system should present an API that lets the application do the following things:

- Insert new text at a specific point in the existing text.
- Delete a range of existing text.
- Replace a range of existing text with new text.
- Undo the most recent change to the text.
- Undo repeatedly.
- Redo the most recently "undone" change.
- Redo repeatedly

Any fresh insert/delete/change should erase any existing redo stack, and start a fresh one.

Assume that the edited text is too large to store in multiple versions in memory. So a solution that for example just keeps a lot of copies of the string as it's modified won't work. Your system must keep a record of the changes themselves, and be able to apply them "forward" and "backward" (when undoing).

Start by designing a model of Java classes to meet the above requirements, and send this to your instructor for review. You may be asked to revise the design one or more times.

When you have an approved design, build your classes, and build a test harness that proves that the requirements are met. Your test can be a simple Java **main** method that drives your API; or, if you're familiar with JUnit, a JUnit test class. See the following pages for test cases that you can script out. Do not build an interactive application.

Here are a set of actions the application might take, and the expected value of the edited text after each one:

0. Start with the string “Every good boy does fine.”

1. Insert "d" at offset 13: "Every good body does fine."

2. Delete 6 characters at offset 5: "Everybody does fine."

3. Change the 4 characters at offset 15 to "well": "Everybody does well."

4. Undo: "Everybody does fine."

5. Undo: "Every good body does fine."

6. Undo: "Every good boy does fine."

7. Redo: "Every good body does fine."

8. Redo: "Everybody does fine."

9. Change the 7 characters at offset 9 to "? F": "Everybody? Fine."

10. Try a re-do, and see that you can't redo action 3 anymore.

11. Undo: "Everybody does fine."

12. Undo: "Every good body does fine." (again undoing action 3)

13. Redo: "Everybody does fine."

Here's a second test case that focuses on the need to clear the redo stack:

0. Start with "She sells shells."

1. Insert "sea" to get "She sells seashells."

2. Change "sea to "she", to get "She sells sheshells."

3. Change "She" to "He", to get "He sells sheshells."

4. Undo once, back to "She sells sheshells."

5. Undo again, back to "She sells seashells."

6. Add the end of the sentence to get: "She sells seashells by the seashore."

7. It should not be possible to re-do changes 2 and 3; change 6 cleared the redo stack in favor of itself.

If you're done, and have more time, try adding additional features:

- Convert a range of existing text to uppercase.
- Convert a range of existing text to lowercase.
- Both of these new features to be undoable like the rest.