Tutorial 7: Trees, Stacks, Parsing, and File I/O

Computer Science 214: Data Structures and Algorithms

27 March 2008 Due: 17 April 2009

Instructions

Your completed tutorial—which includes the work done for Tutorial 6—must be submitted in a `jar`, containing your source code, via WebCT by Friday, 17 April 2009 at 14:00. Because of the test week of the engineering faculty, there will not be a tutorial session next week.

Overview

In this tutorial, you will

- continue your valiant quest of last week to handle bookmarks by amending your existing tree structure so that it is editable; and
- enable your program to store and retrieve the structured bookmark data to and from disk.

Tutorial

1. Keep on working in the existing package structure.
2. Start with your test implementation for the bookmarks component from last week, and change this component so that it is editable. That is, it must provide the following functionality:
   - Adding a folder;
   - adding a bookmark;
   - removing a bookmark or folder;
   - cutting a bookmark or folder, keeping it available for pasting; and
   - pasting a bookmark or folder, if any is available.

   These operations must all be performed on the current node selection, if allowed by the context.
3. Remember that there are certain limitations to where new elements may be added. For example, a bookmark may not have "subbookmarks". Similarly, there are things to keep in mind when removing tree nodes: Removal of a folder must entail the recursive removal of all its subfolders as well.

4. Cutting and pasting may be accomplished by keeping a "clipboard" reference in your bookmarks component. If more than one cuts are performed in sequence, only the last node cut has to be stored. If the clipboard reference is non-null when the bookmarks component exits, remove this reference.

5. For extra credit, implement drag-and-drop functionality.

6. When the bookmarks component is closed, the bookmarks must be saved to disk. You may change the behaviour so that the component is actually kept around, but invisible—however, the bookmarks must still be saved to disk. Remember that if the browser window is closed while the bookmarks component is still open, the data must still be saved before exiting.

7. Use the following XML-like markup to structure your bookmarks file:

   (a) The file starts with "`<bookmarks>`" and ends with "`</bookmarks>`". In between them follow any succession of nested, balanced bookmark and folder tags.

   (b) A folder starts with "`<folder>`" and ends with "`</folder>`". It must contain a *name* tag, starting with "`<name>`" and ending with "`</name>`", as well as a *description* tag, starting with "`<desc>`" and ending with "`</desc>`". Any of the latter two may be empty, and as long as they are balanced correctly, they may appear any place between the starting and ending folder tags. A folder may then also contain further nested folder tags pairs, these storing the subfolders, as well as bookmark (note: singular) tag pairs.

   (c) A bookmark (note: singular) starts with "`<bookmark>`" and ends with "`</bookmark>`". Between these two tags, in no particular order, must appear the *title* ("`<title>`" and "`</title>`"), *description* ("`<desc>`" and "`</desc>`"), and URL ("`<url>`" and "`</url>`"). No other tag pairs may be nested in a bookmark.

   (d) The structure must mirror the structure of the tree in your bookmarks component, specifically in terms of order.

8. Save the bookmarks to a file "`bookmarks`" in the root directory of your project. Use the template method pattern in Section 7.3.7 of your text book for a generalised Euler tour to generate your XML-like structure of the bookmarks file. Note that the "visit below" action is empty in our case.

9. If the "`bookmarks`" file exists when the bookmarks component is opened, it should be read in and displayed correctly. Use the stack implementation available in Java to implement an idea, similar to that for parsing balanced parenthesis, for parsing your bookmarks file.

10. Use `JMenuBar` and its associated Swing objects to create a menu system for your application. For now you should have two menus attached to the menubar, "File" and "Manage". Add an "Exit" item to "File", a "Bookmarks" item to "Manage", and link them to their targets via the appropriate mechanism. Refer to the Java Tutorial and API docs for information.