Benjamin Knauth

CMSC 335

29 September 2014

Project #2 Description

*UMLs:*

**CMSC335\_Project\_3**

main(String args[]): void

**Library**

Authors : LinkedHashMap<Integer, Author>

+Library()

+toString() : String

+sortAuthorNames(LinkedHashMap<Integer, Author> authors) : LinkedHashMap<Integer, Author>

+sortAuthorIndexes(LinkedHashMap<Integer, Author> authors) : LinkedHashMap<Integer, Author>

**Author**

-index : int

-numberBooks : int

-name : String

-address : String

Books : LinkedHashMap<Integer, Book>

Journals : LinkedHashMap<Integer, Journal>

ExtraFields : ArrayList<String>

+Author()

+Author(int index, String name, String address, ArrayList<String> ExtraFields)

+toString() : String

+sortBooksTitle(LinkedHashMap<Integer, Author> books) : LinkedHashMap<Integer, Author>

+sortBooksPrice(LinkedHashMap<Integer, Author> books) : LinkedHashMap<Integer, Author>

+sortBooksIndex(LinkedHashMap<Integer, Author> books) : LinkedHashMap<Integer, Author>

+sortJournalDate(LinkedHashMap<Integer, Author> journals) : LinkedHashMap<Integer, Author>

+sortJournalIssue(LinkedHashMap<Integer, Author> journals) : LinkedHashMap<Integer, Author>

+getIndex() : int

+setIndex(int index) : void

+getNumberBooks() : int

+setNumberBooks() : void

+getName() : String

+setName(String name) : void

+getAddress() : String

+setAddress(String address) : void

**Book**

-index : int

-authorIndex : int

-price : double

-title : String

-genre : String

-author : String

ExtraFields : ArrayList<String>

+Book()

+Book(int index, String title, String genre, double price, int author\_index, ArrayList<String> ExtraFields)

+toString() : String

+getIndex() : int

+getAuthorIndex : int

+getPrice() : double

+getTitle() : String

+getGenre() : String

+getAuthor : String

+setIndex(int index) : void

+setAuthorIndex(int authorIndex) : void

+setPrice(double price) : void

+setTitle(String title) : void

+setGenre(String genre) : void

+setAuthor(String Author) : void

**Journal**

-issueNumber : int

-author\_index : int

-date : String

ExtraFields : ArrayList<String>

+Journal()

+Journal(String date, int issueNumber, int author\_index, ArrayList<String> ExtraFields)

+toString() : String

+getDate() : String

+getIssueNumber(): int

+getAuthorIndex() : int

+setDate(String newDate) : void

+setIssueNumber(int newIssueNumber) : void

+setAuthorIndex(int author\_index) : void

**JTreeGUI**

-tree : JTree

-coBar : JProgressBar

-reBar : JProgressBar

+JTreeGUI(Library)

**CoClient**

-tree : JTree

-coBar : ProgressBar

-button : JButton

+CoClient(JTree, ProgressBar, JButton)

+run()

**ReClient**

-tree : JTree

-reBar : ProgressBar

-button : JButton

+ReClient(JTree, ProgressBar, JButton)

+run()

*User Guide:*

Hello and welcome to the Library Information Management System. To begin, open the system and follow these instructions:

1. Select a .txt file with a list of books and authors to put in the system. The file must have the following format:

-Authors: a:index:name:address

-Books: b:index:title:genre:price:author index

-Journals: j:date:issue number:author index

(date needs to be in numerical form with a four digit year and two digit month, ie. 201405)

The LIMS reads the file line by line. If any index is unknown, please enter “99999”. If any name, address, title or genre is unknown, please use “Unknown”. Extra items can be added to each line as long as a colon(:) separates each item. Spaces next to colons are ignored, but spaces inside titles, addresses, genres, and names are not. Any book that has an author index that doesn’t match any listed author’s index will be placed under the fake author “Unknown Author” who has an index of “99999”.

1. Once your file is uploaded and the LIMS initialized, you are ready to search through the books and Journals. A window will pop up with a tree in the top panel, and buttons on the bottom panel. You can navigate the full inventory with this tree.
2. When you are ready to check out an item, simply highlight it, and click “Check Out”. If you wish to return a book, click “Return”. It takes a few seconds for the system to update. During this this time, if you are unsure of your decision, simply click “Pause”, and the program will pause. Click the button again to resume. You can also click “Cancel” to if you made your decision in error.

**Test Plan**

Input(.txt file):

b : 10001 : Java Basics : Science : 11.50 : 20003

b : 10002 : Advanced Java : Science : 10.99 : 20003

/b : 10003 : The Making of Apples : Science : 10.99 : 20001

b:10004:Mary Knows Best:Science:10.99:20002

a:20001:John Smith:Computers St. 50 Seattle

a : 20002 : Mary Jones : Literature Lane 25

a:33333:Amy Adams:Wisconsin Street

j:201401:5:20001

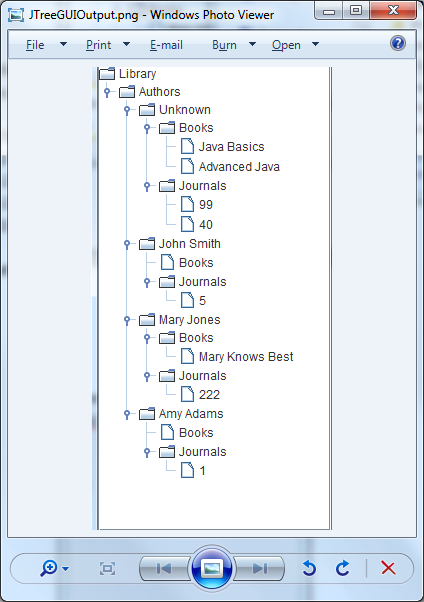
j:201405:222:20002

j:199404:99:20003

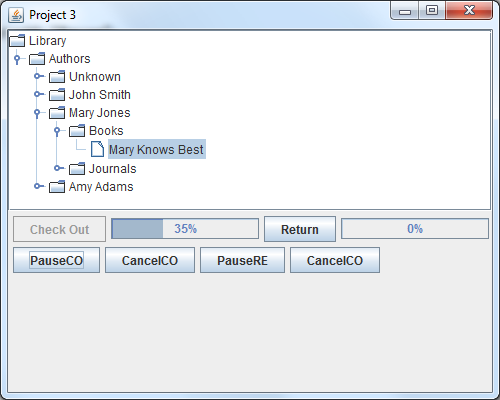
j:200012:40:20004

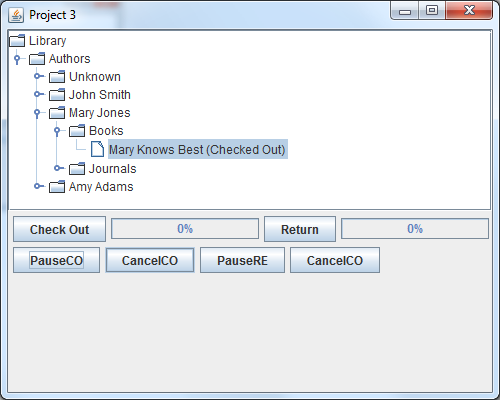
j:200110:1:33333

***JTree Output:***



**“Check Out” Output:**

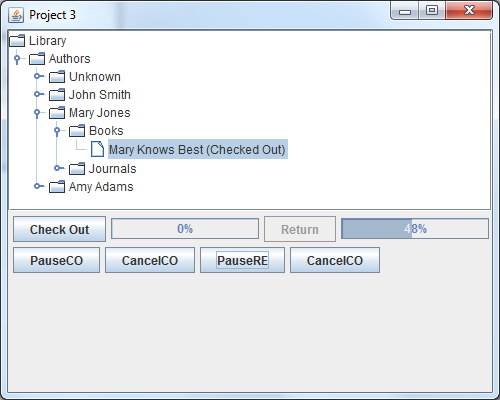


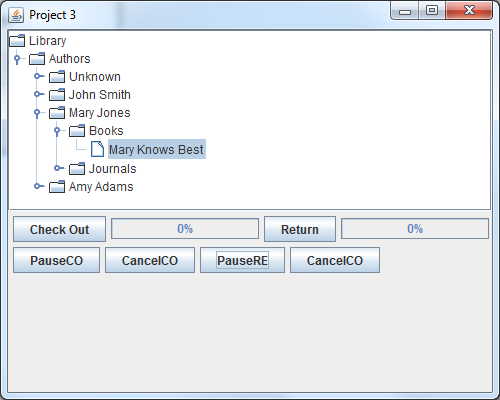


-“Pause” works

-“Cancel” works

**“Return” Output:**





-“Pause” works

-“Cancel” works

*Comments:*

Plans for improvement in the future:

-Make GUI look a bit nicer.

-Implement lock instead of disabling “Check Out” and “Return” buttons.

-Make entire program tighter (allow only leaves to be checked out, not nodes, etc…)

*Lessons Learned:*

-Multithreading is a whole new area of programming I need more practice with, since I’m just starting out as a programmer and multi core processors are the latest architectural innovation.

-Threads make more applicable sense to me.

*Areas to Improve:*

-Need more practice with parallel programming.