Cochrane Library Crawler

Nicholas Tylek

2/1/2022

Application Description:

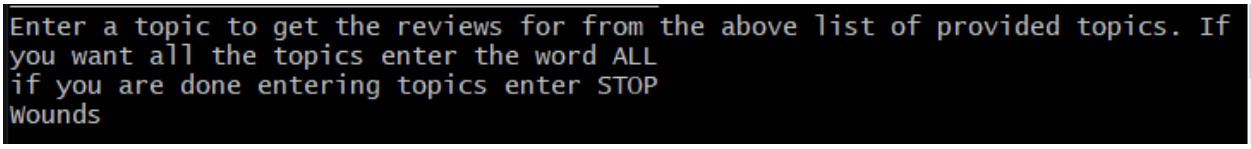
This is a command line application that scrapes data from the Cochrane Library website for the topics page: https://www.cochranelibrary.com/cdsr/reviews/topics. It will get all the topics. Then based on user input it will write the topic information for the topics specified by the user to a text file.

Getting Started:

To run this application in the command line type the command:

- java -jar cochrane-library-crawler.jar <OPTIONAL_FILE_PATH_AND_NAME>

While the application is running it print out a list of topics to the user. The user can keep entering topics from the list that they want written to the file.



Application Design:

This is written like a basic web scrapper. It makes a connection to url and retrieves the content. It parses the content to find the data. And then writes the data.

Detailed Flow:

1) Making an Http request to get the topics page content as html. Apache HttpClient is used to make a get request.
2) To make the get request first a CloseableHttpClient is created to maintain a session. The http client sets user agent to avoid a 419 error being returned from the Cochrane Library. A connection manager is created to maintain session to be able to make successful connections to pages like the next page that rely on cookies and other session variables. Circular redirects are also allowed to allow the user to enter the topic urls provided on the page.
3) When the topics page is returned alongside the HTTP client to use for later get requests the topics page content is stored as a Document object using Jsoup. Jsoup is then used to parse the data by class name. The data is then stored to topic objects. The topics are printed to the user so they can select what topics they want reviews for.
4) The user then inputs what topics they want. Based on the topics entered the topics will be looped through and a get request using the existing http client will be made to get the reviews content to be parsed by jsoup. If there is a next url it will recursively get the reviews for the next page. All the reviews will be stored in a list of review objects.
5) For each topic iteration the review objects will be written to the text file provided in the format of URL|Topic|Title|Authors|Date and then newline.

6) When each topic page and each review page for each topic is parsed and written to the file all connections will be closed and the application is finished.

Future Work:

Due to time constraints and requirements the following features were not implemented and would be nice to have.

It would be nice to scrape Reviews by other means such as Conclusion Changed, Language, Publication Date instead of just topic.

Unit testing for the classes using Junit would be a good addition to help maintain the code.

When a lot of topics or all the topics are being written to a file it can take a while. It could be a good idea to make this application multi-threaded if time constraints are a requirement.

Maven Dependencies:

Apache HttpClient 4.5.13

Jsoup 1.10.2

Junit 4.11(not leveraged)