Cochrane Library Scraper

Harry A. Woodworth

24 March 2020

Purpose

The Cochrane Library Scraper was created to scrape the URLs and Metadata from reviews in the Cochrane Library website. The review data is formatted and output into a file.

Design Reasoning

The Cochrane Library is designed to be compiled using Maven and runs on the command line. It uses a Runner class to call the static function scrape() of the CochraneLibraryScraper class to scrape the Cochrane Library website and gather the URLs and metadata from the reviews under each topic. The function scrape() takes an integer that determines how many topics to scrape from the website, or if a 0 is entered then all of the topics are entered. Topics are scraped in alphabetical order.

The CochraneLibraryScraper class is designed for multithreading. Multithreading allows the long process of scraping thousands of reviews to be done concurrently with each topic. Currently the program runs on one thread because the Cochrane Library website becomes overloaded by more than one thread and does not provide any review results. This problem can be investigated to enable multithreading to lower the scrape completion time.

The CochraneLibraryScraper uses the Apache Http library to make an http GET request to the Cochrane Library topics page (https://www.cochranelibrary.com/cdsr/reviews/topics). Once it has a positive response it will use the JSoup library to parse the response String and scrape the topic names and URL's to each topic's search result page.

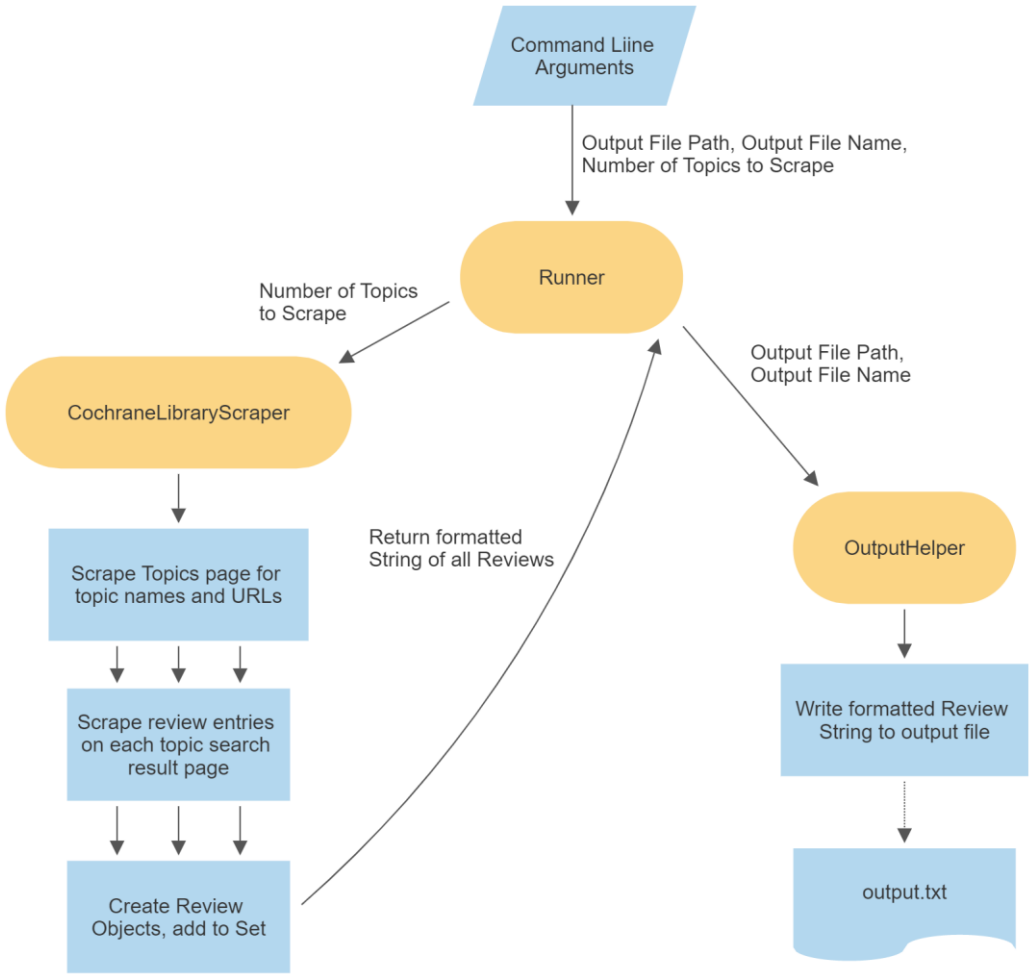
I chose to use the JSoup library because of its user friendliness, clear html element selection functions, and simple integration with the Apache Http library.

It loops through each topic (or in the case of enabling multithreading, a new thread will be created for each topic and the following logic will run on each thread), making a new http GET request to the topic search results page. Once it has a positive response it will scrape the URL and metadata from each review on the page (Each page by default has up to 25 reviews). The metadata includes the title, author names, publication date, and the current topic title. Once the URLs and metadata have been scraped from each page the Review objects are created and added to a synchronized Set. Once all the reviews from every topic have been scraped the scrape() method closes all connections and resources before returning a formatted String of all the Review objects. The Review object toString() function returns a String with the URL, topic, title, author, and date delimited by a '|' character. Each Review String is separated by a new line.

The formatted review data String is returned to the Runner class, where it is sent as an argument, along with the other two command line arguments denoting the output file path and name, to the OutputHelper class' static toFile() function. The toFile() function creates or overwrites the output file at the specified path with the formatted review data String.

The program uses a custom Logger class to log information, debug information, and error information to the command line. This is used in place of System prints in order to be replaced by a more robust Logger class in whatever program it is used with.

High Level Program Flow Chart



Additional Feature Ideas

Debugging the multithreading will allow the program to run much faster. Currently the time it takes to scrape every review on one thread is long.

The Cochrane Library offers additional information such as Protocols, Trials, Editorials, Special Collections, Clinical Answers, and other Reviews from an outside source (Epistemonikos systematic reviews) that can be scraped for URLs and metadata in a similar fashion to the topic reviews.