Java Based Text Scraper

- Shreyansh Daga

Given Assignment

Design and build a robust text scraper that will connect to a page on **www.shopping.com** and return results about a given keyword. There are two queries that will be performed:

- **Query 1**: Total number of results
  Given a keyword, such as "digital camera", return the total number of results found.
- **Query 2**: Result Object
  Given a keyword (e.g. "digital cameras") and page number (e.g. "1"), return the results in a result object and then print results on screen. For each result, return the following information:
  - Title/Product Name (e.g. "Samsung TL100 Digital Camera")
  - Price of the product
  - Shipping Price (e.g. "Free Shipping", "$3.50")
  - Vendor (e.g. "Amazon", "5 stores")

Solution

I have used the **Jsoup external library** for parsing the html.

The solution has the following files

1) JavaSC.java
   - Contains the main method of the program
   - Checks performed over user input argument
   - Appropriate methods invoked depending on query type
2) MyCrawler.java
   - Contains the class definition of the Crawler
   - Contains methods to receive input, format given query string and page number.
   - Returns the results for the respective query type
3) PageItem.java
   - Contains the model class definition for the PageItem object which represents a product on the search results

Running the solution

The solution can be executed in the following ways

1) Java -jar Assignment.jar <Key Word>
2) Java -jar Assignment.jar <Key Word> <Page Number>

Execution and Conclusions

1) Total number of results for a given keyword
   - For a given keyword, the crawler fetches the page, finds out if there are more pages, finds the last page, confirms whether it is indeed the last page and then returns the total number of results found based on what is shown on the last page of the results.

2) List of details of products on a given page with a given keyword and a page number
   - For a given keyword and a page number, the crawler hits the appropriate page number with the key word, checks if the page exists, and then traverses through all the results obtained on the page, parses each result item into a model object and returns an array of these page items.