WEBSCRAPPER FOR WWW.SHOPPING.COM

Saurabh Jaluka

Email: jaluka@usc.edu

Contact No.: 213-300 0922

Index

Table of Contents

Requirement

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")

  For "digital cameras", there should be either 40 or 80 results that return for page 1.

Architecture/Design

Software is designed using the OOP approach. It is properly divided into the needed classes. It is designed in Netbeans IDE using Java. This software is designed to be used as a command line tool, accepting the 2 types of queries as mentioned in the requirement.

Technical

Code

Used HTML parser API: JSOUP

This software has 4 classes, as mentioned below:

1. Product: This class represent the detail of the product as mentioned in the requirement.
   - 4 private field: Title, Price, ShippingInfo, Vendor.
   - No argument constructor
   - 4 argument constructor to initialize the data members.
   - Function displayDetail() to display the detail of the product.
     - Usage: object.displayDetail()
     - Return type void.
2. Products: This class represent list of product.
   - 1 private final List object, for storing all the products.
   - Function to add the product in the productList.
     - Usage: object.add(product)
     - Return type is void.
   - Function to display all the products that is in the List object
     - Usage: object.displayProductList()

        - Return type is void.
3. WebScrapper: This class is the heart of the software, which contains the function to scrap the data.
    - Function getTotalNumberOfResult(String keyword)
        - It takes a keyword and then submits a form with that keyword, gets the response html and scrap it for the result span.
        - It scraps the response html with
            - doc.getElementsByClass("numTotalResults")
        - Return type is void.
    - Function getProducts(String keyword, int pageno)
        - It takes 2 arguments keyword and pageno. Form is submitted based on the pageno value and the response html is then scrapped for the pageno mentioned. Another submit request is made on the basis of the pageno, then the response html is scrapped. It returns the List of products
        - It scraps the product detail as:
            - product.getElementById("nameQA" + index)
            - product.getElementsByClass("priceProductQA"+index)
            - product.getElementsByClass("newMerchantName")
            - product.getElementById("numStoresQA" + index)
            - product.getElementsByClass("freeShip")
            - product.getElementsByClass("calc")
    - Function parseNumberFromResultSpan(String number)
        - To parse the total number of result from the result span.
            - <span class="numTotalResults">
              Results 1 - 28 of 28
              </span>
        - Removed the '+ ' sign from the number.
            - Example: Result 1 – 40 of 1500+. As 1500.
4. Assignment: This is the main class
    - Function validArguments(String[] args) to check if valid arguments are provided or not.
    - Main function that calls the WebScrapper function based on the arguments given to the software.

Complexity

Time Complexity: O(n) whereas n is the no. of products in the page.

Space Complexity: O(n) because the products of the page are stored in the List data structure

End User

- Extract the zip folder.
- Open command prompt, move to the extracted location.
- Type: java –jar Assignment.jar <keyword>
- Type: java –jar Assignment.jar <keyword> <page no>