Report for w09-Practical
CS1003 W09 Practical 170025298 4th,April 2018

Overview:

This practical is mainly designed for use API to get rid of the .xml file by using Java program. There will be six argument that to be input as arguments which does contains search: which can only be input as author, venue, publication; query, which decide the word that is used to search in the DBLP API; cache, which is used for save the xml response that can make the program be faster for the next time.

However, there might be something wrong with the cache that have been provided because some of the cache are lack of .xml extension. It should be right if the extension was provided.

Design:

The program contains two classes with several methods, design of the method will be shown at the follow:

W09Practical class:

- static main(String[] args) – void:

This method is the main class which is used to start and call re-order() method to start the program.

```
public static void main(String[] args) {
    //running program
    //This method is used as re-order the input stream
    re_order(args);

}
```

-static re_order(String[] args) – void:

This method calls searchcheck() and missingArgumentCheck() method first in order to check whether the query after search and whether there are not enough argument. The re-order method will re-order the input stream as the order of search, query and cache and input it into search method as the parameter. Array is used as the important tool which leads to the result that action[i] reflect the vlue of query[i].

-static missingArgumentCheck(String[] args) – void & searchCheck(String[] args) - void

This method is used for check whether there are values missing. If args is less than 5, the program will shows the usage of the program, if the program is 5, the program will shows which value is missing or which value is invalid. If there is something wrong, the malformed message will be print.

The searchCheck method will check whether the query after search switch “venue”, “author” and “publication”, if bit, the failure message will be printed.

Special value:

```
private static final String argMissing = "Missing value for ";
private static final String malformedMsg = "Malformed command line arguments.";
private static final String argInvalid = "Invalid value for ";
```

Because the message was used quite often, I put the message at the top as final.

Search class:

This class is the function class which manage the cache directory, the search query and the result of reading webpage using API. Because of the structure of the DBLP url does not change, I set the final to the common part of the URL and get the whole API by the url() method:

url() - String

```
private final String dblpUrl = "http://dblp.org/search/";
private final String searchFor = "q=";
private final String format = "format=xml";
private final String maxHints = "h=40";
private final String maxConpletion = "c=0";

private String url() {
    String inner = "";
    switch (search) {
        case "publication" :
            inner = "publ";
            break;
        case "author" :
            inner = "author";
            break;
        case "venue" :
            inner = "venue";
            break;
        default :
            System.out.println("Invalid input!");
    }
    String dblpSearch = dblpUrl + inner;
    dblpSearch += "/api";
    dblpSearch += "?" + format;
    dblpSearch += "&" + maxConpletion;
    dblpSearch += "&" + maxHints;
    dblpSearch += "&" + searchFor + query;

    return dblpSearch;
}
```

url() method is used for design the url of the API, which uses a switch to decide the category of search among author, venue and publication. Because of the publication in the website is wrote like “publ”, so I can’t use search query directly into the url.

Search() - void

Search method is the most important method that is the central of this class. The method read the whole xml page and then decide next step to use by using switch(search). The method uses try and catch to avoid exception and if the file is available, the program will uses xml in the cache first.

If DBLP runs well, “switch” will decide the method that will be called next by switching search query.

```
if (doc.hasChildNodes()) {
    NodeList hits = doc.getElementsByTagName( s: "hit");
      System.out.println(hits.getLength());
    switch (search) {
        case "author" :
            author(hits);
            break;
        case "venue" :
            venue(hits);
            break;
        case "publication" :
            publication(hits);
            break;
        default :
            System.out.println("Wrong search category. Please try again!");
            break;
    }
}
else {
    System.out.print("No details found. Please try another query!");
}
```

Venue, publication & author with para: NodeList hits

These three method is quite similar and they will read the whole nodeList which got from search method and get the information of venue, author and publication.

“for” is used for the third time and aims if reach to the child node that include the information that I required, then uses if to reach to the exactly node that I want and get text content.

If all informations that I have got, I uses System.out.print to print the message as required.

```
private void publication(NodeList hits) {
    //body
      System.out.println(hits.getLength());
    for (int i = 0; i < hits.getLength(); i++) {
        NodeList hitItem = hits.item(i).getChildNodes();
        for (int j = 0; j < hitItem.getLength(); j++) {
            Node allChildNodeOfHit = hitItem.item(j);
            if (allChildNodeOfHit.getNodeName().equals("info")) {
                NodeList author = allChildNodeOfHit.getFirstChild().getChildNodes();
                Node numberOfAuthor = author.item( i: 0);
                int number_of_author;
                if (numberOfAuthor.getNodeName().equals("author")) {
                    number_of_author = author.getLength();
                }
                else {
                    number_of_author = 0;
                }
                String title = ""; // = allChildNodeOfHit.getChildNodes().item(1).getTextContent(); //<- will
                NodeList findTitle = allChildNodeOfHit.getChildNodes();
                for (int s = 0; s < findTitle.getLength(); s++) {
                    if (findTitle.item(s).getNodeName().equals("title")) {
                        title = findTitle.item(s).getTextContent();
                    }
                }
                System.out.println(title + " (number of authors: " + number_of_author + ")");
            }
        }
    }

}
```

calculateDetails(String secondUrl) – void

This method is designed for get the detail of the author which is get from “author” query of the search.

Just is the same as the last method, this method will search the cache first, if not, the program will get the data from the internet. The method uses NodeList by searching tags to get number of publications and number of co-authors as there is only one hit inside of the API.

saveCache(String url, Document doc) – void

This method is used for save the xml response from the internet. If the cache does not exist, the program will create one. The file name uses URLEncoder to encode the name of url and use transform to save it.

Testing:

Stacscheck: 18/18 passed

```
* BUILD TEST - public/build : pass
* TEST - public/_malformed/1/test : pass
* TEST - public/_malformed/2/test : pass
* TEST - public/_malformed/3/test : pass
* TEST - public/_malformed/4/test : pass
* TEST - public/_malformed/5/test : pass
* INFO - public/_style/infoCheckStyle : pass
--- submission output ---
Starting audit...
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./Search.ja
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./Search.ja
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./Search.ja
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./Search.ja
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./Search.ja
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./W09Practi
[WARN] /cs/home/hl74/Documents/cs1003/W09-Practical/src/./noSuchDi
0.xml:0: File does not end with a newline. [NewlineAtEndOfFile]
Audit done.
---

* TEST - public/author/jackcole/test : pass
* TEST - public/author/johnrsmith/test : pass
* TEST - public/author/johnsmith/test : pass
* TEST - public/publication/database/test : pass
* TEST - public/publication/databaset/test : pass
* TEST - public/publication/mariadb/test : pass
* TEST - public/publication/semi-structured/test : pass
* TEST - public/venue/distributeddb/test : pass
* TEST - public/venue/logic/test : pass
* TEST - public/venue/math/test : pass
* TEST - public/venue/parallelmath/test : pass
18 out of 18 tests passed
```

No arguments:

```
/usr/lib/jvm/java-1.8.0-openjdk/bin/java ...
Usage: java W09Practical --search [search query] --query [query] --cache [path]
```

6 unrelated arguments(1 1 1 1 1 1):

```
Command may not true, please try again!
```

As a result, the program runs well and is well performed.

Evaluation:

This program fits all the requirement of the week 9 practical and is well performed. However there are a little bit too much in the main class and it should be improved.

The check of the argument should be set in the search class to brief the main class. I have also repaired a lot about the code style and now I think the program should be all fine. The code of get the missing value and argument is a little bit messive and it should be improved.

As a result, this is a good program that fit to the requirement of the practical, but this can be improved and be more tidy.

Conclusions:

In this practical, I have learned about the use of API and how to take the off-line search. However with experiment, I found that this method is not fit to all website because some of the website are not allowed to let me look at the .xml file of the webpage.

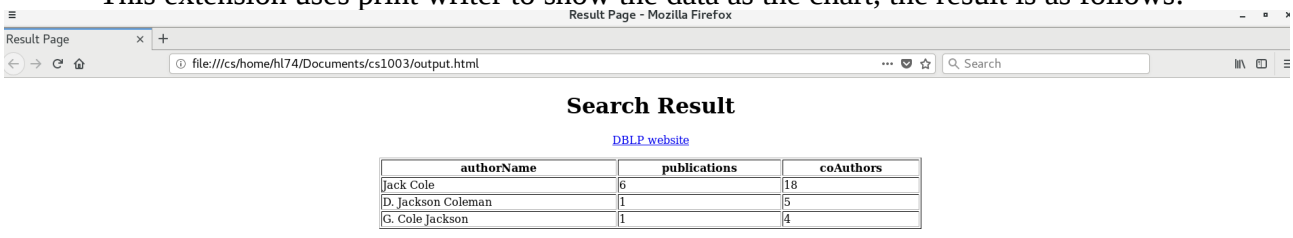
There are a lot of difficulty that I have faced such as how to get the value of xml, previously I use .getValue parameter, however I failed. Finally I uses .getTextContent and get the correct value.

I am not sure whether need I to create the document when there is no such directory, if I have created, please delete noSuchFile directory or the malformed will open and I will fail one check.

Extension will available at the follow:

Extension 1: Export to html:

This extension uses print writer to show the data as the chart, the result is as follows:



authorName	publications	coAuthors
Jack Cole	6	18
D. Jackson Coleman	1	5
G. Cole Jackson	1	4

I scan search first and establish the heading of the table first, then I use the data which was get from search class and print them as the table.

```
writer.println("<h1 style=\"text-align:center\">Search Result</h1>\n" +
        "<p  style=\"text-align:center\"><a href=\"http://dblp.org/\" target=\"blank\">DBLP website</a></p>");
writer.println("<table width=\"800\" border=\"1\" align=\"center\">");
//switch
String code = "";
switch (search) {
    case "venue":
        code = "<tr><th>" + "venue" + "</th></tr>";
        break;
    case "author" :
        code = "<tr><th>" + "authorName" + "</th><th>" + "publications" + "</th><th>" + "coAuthors" + "</th></tr>
        break;
    case "publication":
        code = "<tr><th>" + "title" + "</th><th>" + "number of author" + "</th></tr>";
        break;
}
writer.write(code);
for(String s : list){
      System.out.println(list.size());
    writer.println(s);
}
```

List is to collect data by using list.add() method and is in the Search class.