Proiect 1 RIW - Indexare și căutare

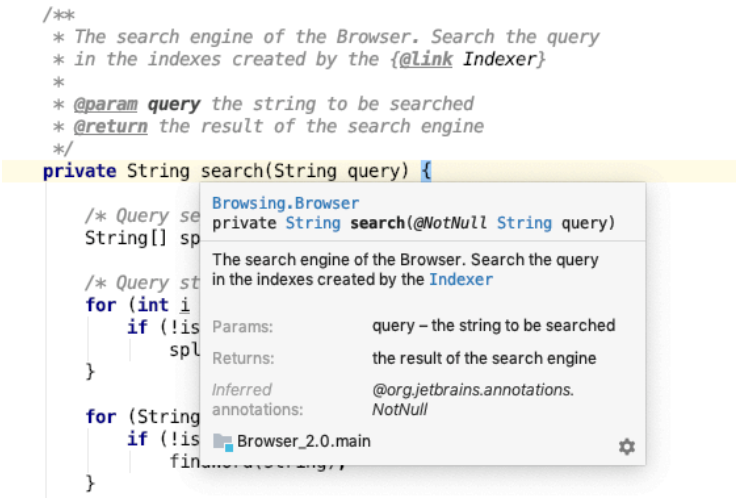
Student: Ciobanu Eduard-David

Manual de utilizare

În acest proiect am implementat un finder ce caută cuvinte în documente introduse de utilizator. Documentele trebuie să fie în format .txt și să fie introduse în folderul *"Browser_2.0/src/main/resources/Documents".*

În clasa Main este apelată la rulare funcția run() care oferă o interfață de tip consolă. În aceasta, utilizatorul este informat despre cum funcționează aplicația și cum se poate utiliza. Aplicația implementează două metode de stocare a indexului: în fișiere respectiv într-o bază de date mongoDB. În interfață utilizatorului i se cere să aleagă metoda pe care programul să o ruleze.

Codul sursă a fost scris în limbajul Java, iar pentru fiecare clasă și metode specifice au fost realizate și documentații javaDoc (Java Documentation). Acestea au scopul de a informa utilizatorul cu privire la scopu funcțiilor și cum funcționează în cadrul proiectului. Un exemplu de javaDoc este dat în următoarea imagine. Pentru o înțelegere amănunțită a programului, recomand parcurgerea codului sursă și a documentațiilor specifice fiecărei clase și funcții.



Etapele parcurse de program

1. Prelucrarea cuvintelor

În această etapă, în cadrul funcției *Indexer.createDirectIndex(…)*, fiecare document este parcurs și se extrag cuvintele. Folosind *algoritmul lui Porter*, implementat în clasa *Stemmer*, fiecare cuvânt este adus apoi la forma canonică din limba engleză, și apoi filtrat. Doar substantivele sau cuvintele ce au la baza de formare un substantiv vor trece de filtrare. Se realizează apoi o listă pentru fiecare document, în care se găsesc cuvintele

din documentul respectiv și numărul de apariții în document pentru fiecare cuvânt în parte. Aceste liste reprezintă indexul direct.

La inițializarea unui obiect *FileIndexer,* se utilizează constructorul specific clasei cu următorii parametri: *FileIndexer(**int** MAX_DOCS_PER_FILE, **int** MAX_WORDS_PER_FILE).*

- *MAX_DOCS_PER_FILE* reprezintă numărul de documente ce se dorește a fi stocat într-un singur fișier de indexare directă;
- *MAX_WORDS_PER_FILE* reprezintă numărul de cuvinte ce se dorește a fi stocat într-un singur fișier de indexare inversă;

În fișiere de indexare directă, stocate în folderul *"Browser_2.0/resources/Indexes/Direct",* listele create sunt stocate în format *JSON*. În următoare imagine este un exemplu de index direct stocat într-un fișier de indexare.


```
{"Volkswagen":{"german":3,"largest":1,"28":1,"deliv":1,"main":1,"compani":1,"wolfsburg":1,'
{"Audi":{"german":3,"ring":1,"luxuri":2,"four":2,"complex":1,"ingolstadt":1,"latin":2,"banc
{"BMW":{"tt":1,"german":2,"isl":1,"float":1,"bmw":7,"sub-brand":1,"twelfth":1,"english":1,'
```


2. Indexarea inversă

În etapa curentă, folosind indexul direct creat la pasul precedent, se realizează mai multe liste pentru fiecare cuvânt în parte. O listă de acest fel, pentru un anumit cuvânt, conține o listă de documente în care se află cuvântul respectiv, și numărul de apariții. Aceste liste se numesc Index Invers. Pentru *FileIndexer*, indexul este stocat în fișiere, în format *JSON*, în folderul *"Browser_2.0/resources/Indexes/Inverse".*

De aceste lucruri se ocupă funcția *Indexer.createInverseIndexer(…)*. Un exemplu de Index Invers este în imaginea următoare:


```
{"flour":{"Pancake":1,"Soup":1}}
{"fluffi":{"Pancake":1}}
{"food":{"Pancake":2,"Soup":1,"Pizza":2}}
{"forc":{"Furious_7_2015":2}}
{"fork":{"Pizza":1}}
{"form":{"Soup":1,"The_Avengers_2012":1,"Audi":1}}
{"formal":{"Pizza":1}}
{"formula":{"BMW":1}}
{"found":{"Furious_7_2015":1,"Volkswagen":1,"BMW":1,"Audi":1}}
{"foundat":{"Audi":1}}
{"founder":{"Audi":1}}
{"four":{"Audi":2}}
{"french":{"Pancake":1,"Soup":2}}
```


3. Căutarea cuvintelor

După realizarea indexului direct și invers, se instanțiază clasa Browser care oferă funcționalitatea de căutare a unor cuvinte introduse, în fișierele găsite în folderul "*Browser_2.0/src/main/resources/Documents".* Funcția de căutare din clasa Browser folosește „*metoda booleană*" de căutare. Utilizatorul este informat în interfața din consolă, cum se poate folosi această funcție. În cadrul funcției, cuvintele introduse de utilizator pentru a fi căutate, vor fi procesate și aduse la forma canonică. În funcție de operatorii specifici metodei booleane, introduși în query-ul de căutare, listele cu documentele găsite de către funcția *search* vor fi scrise în consolă.