Moteur de recherche avec Hadoop

Généré par Doxygen 1.8.2

Vendredi Mai 10 2013 14 :14 :59

Table des matières

Chapitre 1

Index des classes

1.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

Chapitre 2

Documentation des classes

2.1 Référence de la classe window.Fenetre

Est dérivée de JApplet.

Fonctions membres publiques

– void init ()

2.1.1 Description détaillée

Applet du programme, récupère l'entrée utilisateur dans un premier JTextField et réécrit les résultats en dessous

Auteur

Corbel Elodie, M'Ghari Kevin, Renou Clarisse

Version

2.0

Voir également

Page web

2.1.2 Documentation des fonctions membres

2.1.2.1 void window.Fenetre.init ( )

Méthode d'initialisation de l'applet whereSearch représente le chemin où sont stockés les fichiers txt où l'on veut lancer les recherches

Voir également

Paths

Action sur le bouton Search Construit la hashmap par rapport au ficher d'Index Puis rend le résultat

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/window/Fenetre.java

2.2 Référence de la classe reader.FileRead

Fonctions membres publiques

- FileRead (String file, List< Long > lines, String word)
- StringBuilder getLinesText ()

Fonctions membres privées

- StringBuilder getContextLine (Long line)
- String getFilePart (Long line)
- String formatStringStrong (String line)

Attributs privés

- final String fileInputDir = Paths.inputFilesSplitDir
- final int nbLinesPerFile = 100
- String fileName
- List< Long > lines
- String wordToSearch

2.2.1 Description détaillée

Classe qui récupère les lignes du fichier pour un mot donné

Auteur

Corbel Elodie, Renou Clarisse

2.2.2 Documentation des constructeurs et destructeur

2.2.2.1 reader.FileRead.FileRead (String *file,* List< Long > *lines,* String *word*)

Récupère la liste des lignes et le nom du fichier pour le mot recherché

Paramètres

file	nom du fichier où sont faites les recherches
lines	liste des numéros de lignes où les mots sont trouvés
word	mot recherché

2.2.3 Documentation des fonctions membres

2.2.3.1 String reader.FileRead.formatStringStrong (String *line*) [private]

Met en forme le résultat avant l'affichage : Mise en avant (gras) du mot recherché

Paramètres

line	texte de la ligne où se trouve le mot

Renvoie

le texte de la ligne où se trouve le mot avec le mot en gras

2.2.3.2 StringBuilder reader.FileRead.getContextLine (Long *line*) [private]

On décide de prendre 3 lignes autours du Mot recherché

Paramètres

line	numéro de la ligne qui contient le mot

Renvoie

le contexte pour la ligne donnée = 3 lignes autours

Exceptions

Attrape	l'exception si les fichiers splites n'ont pas ete trouves ou s'il sont mal decoupes

2.2.3.3 String reader.FileRead.getFilePart (Long *line*) [private]

Récupère le chemin où sont stockés les fichiers découpés

Paramètres

line	ligne qui contient le mot

Renvoie

le chemin du fichier découpé

Voir également

fileInputDir

2.2.3.4 StringBuilder reader.FileRead.getLinesText ( )

Trouve les lignes qui entourent le mot une fois trouvé.

Renvoie

le contexte des lignes où se trouve le mot dans le fichier

Voir également

getContextLine(Long)

2.2.4 Documentation des données membres

2.2.4.1 final String reader.FileRead.fileInputDir = Paths.inputFilesSplitDir [private]

fileInputdir Répertoire où se trouvent les fichiers découpés

2.2.4.2 String reader.FileRead.fileName [private]

fileName Nom du fichier à lire

2.2.4.3 List<Long> reader.FileRead.lines [private]

lines lignes à afficher

2.2.4.4 final int reader.FileRead.nbLinesPerFile = 100 [private]

Pour parcourir plus vite les fichiers on a décidé de les couper toutes les 100 lignes avec un script bash nbLinesPerFile Nombre de lignes par fichier

2.2.4.5 String reader.FileRead.wordToSearch [private]

wordToSearch mot que l'on cherche

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/reader/FileRead.java

2.3 Référence de la classe search.FoundInfos

Fonctions membres publiques

– FoundInfos (String mot, List< String > fichiers)
– String getMot ()
– List< String > getFichiers ()

Attributs privés

– String mot
– List< String > fichiers

2.3.1 Description détaillée

Classe qui associe à un mot la liste de ses fichiers

Auteur

Olivier Mickaël

2.3.2 Documentation des constructeurs et destructeur

2.3.2.1 search.FoundInfos.FoundInfos (String *mot,* List< String > *fichiers*)

Paramètres

mot	
fichiers	liste des noms de fichiers dans lesquels se trouvent le mot

2.3.3 Documentation des fonctions membres

2.3.3.1 List<String> search.FoundInfos.getFichiers ( )

Renvoie

la liste des noms de fichiers

2.3.3.2 String search.FoundInfos.getMot ( )

Renvoie

le mot

2.3.4 Documentation des données membres

2.3.4.1 List<String> search.FoundInfos.fichiers [private]

fichiers liste des fichiers où se trouvent le mot

2.3.4.2 String search.FoundInfos.mot [private]

mot mot auquel on va associer des fichiers

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/search/FoundInfos.java

2.4 Référence de la classe grammar.Index2

Fonctions membres publiques statiques

– static void scanFile (BufferedReader input)
– static void builder (String line)
– static void build (String args, Search search)

2.4.1 Description détaillée

Cette classe va parcourir le fichier Index et appeler les méthodes adéquatse pour construire la hashmap Elle est statique afin d'avoir accès à la création d'index partout scanLine est un Scanner qui va s'occuper mot par mot de la ligne pour construire la hashmap builder est IndexBuilder chargé de construire la hashmap recherche est le Search

Auteur

Renou Clarisse

Voir également

IndexBuilder

2.4.2 Documentation des fonctions membres

2.4.2.1 static void grammar.Index2.build (String *args,* Search *search*) [static]

Principale méthode de la classe, elle initialise l'IndexBuilder et le BufferedReader puis lance l'analyse

Paramètres

args	String, nom du fichier Index
search	Search

Exceptions

catch	l'exception si le fichier n'est pas trouve

Voir également

Search : :toDo()

2.4.2.2 static void grammar.Index2.builder (String *line*) [static]

Ici on scanne la ligne envoyée mot par mot. On récupère ainsi grâce à l'index bien constitué, le titre, l'offset et le mot en question. L'index est constitué ainsi : word , title.txt , offset1 offset2 etc

Paramètres

line	la ligne envoyée

Voir également

IndexBuilder

2.4.2.3 static void grammar.Index2.scanFile (BufferedReader *input*) [static]

Cette méthode scanne ligne par ligne le fichier index pour la transmettre au builder(String)

Paramètres

input	BufferedReader on entre le fichier index

Voir également

builder(String)

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/grammar/Index2.java

2.5 Référence de la classe index.IndexBuilder

Fonctions membres publiques

– void addLine (long entierLu)
– void addFile (String ident)
– void addWord (String ident)
– void buildSet ()

Fonctions membres publiques statiques

– static Map< Integer, HashMap
< String, Informations > > getIndex ()

Attributs privés

– String currentWord
– String currentFile
– ArrayList< Long > lines
– Integer currentID

Attributs privés statiques

– static Map< Integer, HashMap
< String, Informations > > index

2.5.1 Description détaillée

Cette classe s'occupe de construire l'index avec la création d'une hashmap

Auteur

Olivier Mickaël

2.5.2 Documentation des fonctions membres

2.5.2.1 void index.IndexBuilder.addFile (String *ident*)

Mémorise le nom de fichier lu afin de pouvoir l'utiliser à la fin de la lecture des informations concernant le mot

Paramètres

ident	identifiant lu dans le fichier construit par Hadoop

2.5.2.2 void index.IndexBuilder.addLine (long *entierLu*)

Ajoute le numéro de ligne lu à la liste courante des numéros de ligne

Paramètres

entierLu	le numéro de ligne lu dans le fichier construit par Hadoop

2.5.2.3 void index.IndexBuilder.addWord (String *ident*)

Mémorise le mot lu afin de pouvoir l'utiliser à la fin de la lecture des informations concernant le mot Incrémente l'identifiant courant unique

Paramètres

ident	identifiant lu dans le fichier construit par Hadoop

2.5.2.4 void index.IndexBuilder.buildSet ( )

Ajoute à la hashmap les informations pour un mot donné et un fichier donné à effectuer à chaque fin de ligne du fichier construit par Hadoop

2.5.2.5 static Map<Integer, HashMap<String, Informations > > index.IndexBuilder.getIndex ( ) [static]

Renvoie

index construit à partir du fichier construit par Hadoop

2.5.3 Documentation des données membres

2.5.3.1 String index.IndexBuilder.currentFile [private]

currentFile fichier courant dans lequel se trouve le mot courant

2.5.3.2 Integer index.IndexBuilder.currentID [private]

currentID identifiant unique pour chaque combinaison (mot,nomFichier)

2.5.3.3 String index.IndexBuilder.currentWord [private]

currentWord mot courant mémorisé pour à la fin de la lecture de la ligne de l'index lui associer ses numéros de ligne

2.5.3.4 Map<Integer, HashMap<String, Informations > > index.IndexBuilder.index [static], [private]

index : Hashmap qui contiendra toutes les informations venant de l'indexation. Sa clé est un identifiant unique, les valeurs sont une HashMap ayant pour clé le mot et pour informations le nomdufichier et la liste des offsets

Voir également

Informations

2.5.3.5 ArrayList<Long> index.IndexBuilder.lines [private]

lines liste des numéros de ligne pour un mot dans un fichier

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/index/IndexBuilder.java

2.6 Référence de la classe index.Informations

Fonctions membres publiques

– Informations (ArrayList< Long > lines, String file)
– ArrayList< Long > getLines ()
– void setLines (ArrayList< Long > lines)
– String getFile ()
– void setFile (String file)

Attributs privés

– ArrayList< Long > lines
– String file

2.6.1 Description détaillée

Classe qui mémorise les Informations liées à un mot, à savoir le fichier dans lequel il se trouve et les lignes où il a été repéré dans ce fichier

Auteur

Olivier Mickael

2.6.2 Documentation des constructeurs et destructeur

2.6.2.1 index.Informations.Informations (ArrayList< Long > *lines*, String *file*)

Constructeur

Paramètres

lines	liste des numéros de ligne dans lequel se trouve un mot donné
file	nom du fichier dans lequel se trouve un mot donné

2.6.3 Documentation des fonctions membres

2.6.3.1 String index.Informations.getFile ( )

Renvoie

le nom de fichier

2.6.3.2 ArrayList<Long> index.Informations.getLines ( )

Renvoie

la liste des numéros de ligne

2.6.3.3 void index.Informations.setFile (String *file*)

Attribue le nom de fichier

Paramètres

file	nom du fichier

2.6.3.4 void index.Informations.setLines (ArrayList< Long > *lines*)

Paramètres

liste	des numéros de ligne

2.6.4 Documentation des données membres

2.6.4.1 String index.Informations.file [private]

Nom du fichier dans lequel se trouve un mot donné

2.6.4.2 ArrayList<Long> index.Informations.lines [private]

Liste des numéros de ligne dans lequel se trouve un mot donné

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/index/Informations.java

2.7 Référence de la classe window.Logger

Fonctions membres publiques statiques

– static void createLogger ()
– static void addInLog (String toLog)

Fonctions membres privées statiques

– static String formatDate (Date date)

2.7.1 Description détaillée

Classe qui met en place le log de ce que souhaite le programmeur

Auteur

Renou Clarisse

Version

1.0

2.7.2 Documentation des fonctions membres

2.7.2.1 static void window.Logger.addInLog (String *toLog*) [static]

Méthode principale qui permet d'ajouter à la suite du fichier la phrase de log que l'on veut

Paramètres

toLog	commentaire de log

Exceptions

en	cas de probleme d'ouverture ou d'écriture du fichier capture l'exception

2.7.2.2 static void window.Logger.createLogger ( ) [static]

CreateLogger permet d'initialiser le logger en indiquant le chemin de création du fichier pathLog et son nom logFileName.

Exceptions

IOException	en cas de probleme de création du fichier de log

2.7.2.3 static String window.Logger.formatDate (Date *date*) [static], [private]

Met la date au format dd/MM/yyyy kk :mm :ss

Paramètres

date	Date la date à formatter

Renvoie

la chaîne de caractères avec la date formattée

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/window/Logger.java

2.8 Référence de la classe path.Paths

Attributs publics statiques

– static final String outputIndexLocation = cheminElodieOutput
– static final String inputFilesSplitDir = cheminElodieSplit
– static final String logFilePath = pathElodieLog

2.8.1 Description détaillée

Classe générale dans laquelle on doit mettre les différents chemins utilisés par le moteur de recherche

Auteur

Corbel Elodie

2.8.2 Documentation des données membres

2.8.2.1 final String path.Paths.inputFilesSplitDir = cheminElodieSplit [static]

inputFilesSplitDir dossier où se trouvent les fichiers découpés par le script découpant les fichiers toutes les 100 lignes

2.8.2.2 final String path.Paths.logFilePath = pathElodieLog [static]

logFilePath dossier où on enregistre le log du programme

2.8.2.3 final String path.Paths.outputIndexLocation = cheminElodieOutput [static]

outputIndexLocation endroit où se trouve le fichier output.txt sortant de l'index

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/path/Paths.java

2.9 Référence de la classe search.Search

Fonctions membres publiques

– Search (String expression)
– List< String > getToSeek ()
– Seeker getSeeker ()
– void toDo ()

Fonctions membres privées

– void setToSeek (List< String > toSeek)
– List< String > supprNonIndexe (List< String > motsEntres)

Attributs privés

– List< String > toSeek
– String expression
– Seeker seeker

2.9.1 Description détaillée

Classe qui récupère l'expression entrée dans le moteur de recherche l'analyse, supprime les mots non référencés, lance des seekers qui vont récupérer les occurences dans les fichiers

Auteur

Olivier Mickaël

2.9.2 Documentation des constructeurs et destructeur

2.9.2.1 search.Search.Search (String *expression*)

Constructeur associé à l'objet Search qui opère la recherche

Paramètres

expression	entrée dans le moteur de recherche

2.9.3 Documentation des fonctions membres

2.9.3.1 Seeker search.Search.getSeeker ( )

Renvoie

Seeker

2.9.3.2 List<String> search.Search.getToSeek ( )

Permet d'obtenir la liste de mots sur laquelle on opère la recherche

Renvoie

la liste de mots sur laquelle opérer la recherche

2.9.3.3 void search.Search.setToSeek (List< String > *toSeek*) [private]

Permet de modifier la liste de mots à chercher

Paramètres

toSeek	liste des mots à chercher

2.9.3.4 List<String> search.Search.supprNonIndexe (List< String > *motsEntres*) [private]

Supprime les mots de la liste des mots entrés s'ils ne sont pas indexés

Paramètres

motsEntres	

Renvoie

la liste des mots entrés sans les mots non indexés

2.9.3.5 void search.Search.toDo ( )

Fonction qui permet d'analyser l'expression passée, de lancer le seeker mot par mot en tenant compte des prédicats, et d'écrire le resultat

Voir également

#setToSeek(List)
Seeker
FoundInfos

2.9.4 Documentation des données membres

2.9.4.1 String search.Search.expression [private]

Expression entrée dans le moteur de recherche

2.9.4.2 Seeker search.Search.seeker [private]

Seeker qui va récupérer les informations sur les mots

Voir également

Seeker

2.9.4.3 List<String> search.Search.toSeek [private]

Liste des mots à chercher

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/search/Search.java

2.10 Référence de la classe search.Seeker

Fonctions membres publiques

– void seek (String word)
– void seekAnd (String word1, String word2)
– void seekOr (String word1, String word2)
– void seekNot (String word)
– List< FoundInfos > getInfo ()
– void setInfo (List< FoundInfos > info)
– List< FoundInfos > getIntox ()
– void predicatInvalid (String pred)
– StringBuilder getMessage ()
– boolean isPresent (String word)
– List< String > getFichiers (String word)
– int getNbOccurences (String word)
– StringBuilder getLinesText (String word, List< String > sortedFileName)

Fonctions membres privées

– Map< String, List< Long > > getResult (String word)

Attributs privés

– StringBuilder message
– List< FoundInfos > info
– List< FoundInfos > intox

2.10.1 Description détaillée

Recherche dans la hashmap pour un mot donné

Auteur

Olivier Mickaël

2.10.2 Documentation des fonctions membres

2.10.2.1 List<String> search.Seeker.getFichiers (String *word*)

Rend une liste de fichiers qui est celle associee au mappage d'un mot

Paramètres

word	

Renvoie

les fichiers dans lequel le mot est présent

2.10.2.2 List<FoundInfos> search.Seeker.getInfo ( )

Renvoie

la liste des mots et des fichiers dans lequels ils se trouvent

2.10.2.3 List<FoundInfos> search.Seeker.getIntox ( )

Renvoie

la liste des mots et des fichiers à ignorer pour le résultat (prédicat NOT)

2.10.2.4 StringBuilder search.Seeker.getLinesText (String *word,* List< String > *sortedFileName*)

Paramètres

word	String le mot à chercher
sortedFileName	la liste des fichiers dans lequel se trouve le mot

Renvoie

StringBuilder texte des lignes pour un mot dans les fichiers dans lequel il se trouve

2.10.2.5 StringBuilder search.Seeker.getMessage ( )

Met fin au rendu du résultat final

Renvoie

le message

2.10.2.6 int search.Seeker.getNbOccurences (String *word*)

Rend le nombre d'occurences totales d'un mot dans l'index

Paramètres

word	

Renvoie

le nombre d'occurences d'un mot dans tous les fichiers

2.10.2.7 Map<String,List<Long> > search.Seeker.getResult (String *word*) [private]

Rend les numéros de lignes pour un mot c'est-à-dire une Map dans laquelle on a pour clé le fichier dans lequel le mot se trouve et pour valeur la liste du numéro des lignes

Paramètres

word	

Renvoie

table de hachage dans laquelle on a pour clé le fichier dans lequel le mot se trouve et pour valeur la liste du numéro des lignes

2.10.2.8 boolean search.Seeker.isPresent (String *word*)

Vérifie si le mot est présent dans la hashmap

Paramètres

word	

Renvoie

true si présent false sinon

2.10.2.9 void search.Seeker.predicatInvalid (String *pred*)

Permet de donner des informations sur le formalisme du prédicat utilisé en cas d'echec

Paramètres

pred	le prédicat qui est invalide

2.10.2.10 void search.Seeker.seek (String *word*)

Vérifie si le mot est présent dans un des fichiers txt si oui il récupère son contexte

Paramètres

word	seeked

Voir également

#getLinesText(String)
isPresent(String)

2.10.2.11 void search.Seeker.seekAnd (String *word1,* String *word2*)

Vérifie si les deux mots sont présents dans un même fichier txt si oui il récupère son contexte

Paramètres

word1	String premier mot du prédicat AND
word2	String deuxième mot du prédicat AND

Voir également

#getLinesText(String)
isPresent(String)

2.10.2.12 void search.Seeker.seekNot (String *word*)

Vérifie si le mot est présent dans un des fichiers txt si oui il élimine son contexte du message de retour si présent

Paramètres

word	mot dont les fichiers dans lequel il se trouve à éliminer

Voir également

#getLinesText(String)
isPresent(String)

2.10.2.13 void search.Seeker.seekOr (String *word1*, String *word2*)

Vérifie si les deux mots sont présents dans des fichiers différents si oui il récupère son contexte

Paramètres

word1	String premier mot du prédicat OR
word2	String deuxième mot du prédicat OR

Voir également

#getLinesText(String)
isPresent(String)

2.10.2.14 void search.Seeker.setInfo (List< **FoundInfos** > *info*)

Paramètres

info	liste des mots et des fichiers dans lequels ils se trouvent

2.10.3 Documentation des données membres

2.10.3.1 List<**FoundInfos**> search.Seeker.info [private]

Liste des informations sur la recherche

Voir également

FoundInfos

2.10.3.2 List<**FoundInfos**> search.Seeker.intox [private]

Liste des fichiers à ignorer (prédicat NOT)

Voir également

FoundInfos

2.10.3.3 StringBuilder search.Seeker.message [private]

Résultat rendu à la fin à afficher ensuite

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/search/Seeker.java

2.11 Référence de la classe reader.SortLineNumbers

Fonctions membres publiques

– SortLineNumbers (List< Long > listLignes)
– List< Long > getLinesSorted ()

Fonctions membres privées

– void quicksort (int low, int high)
– void exchange (int i, int j)

Attributs privés

– List< Long > listLignes

2.11.1 Description détaillée

Classe qui applique l'algorithme de tri rapide pour trier les numéros de ligne par ordre croissant

Auteur

Corbel Elodie

2.11.2 Documentation des constructeurs et destructeur

2.11.2.1 reader.SortLineNumbers.SortLineNumbers (List< Long > *listLignes*)

Constructeur

Paramètres

listLignes	liste des lignes a trier

2.11.3 Documentation des fonctions membres

2.11.3.1 void reader.SortLineNumbers.exchange (int *i*, int *j*) [private]

Permute 2 éléments de la liste des lignes

Paramètres

i	indice du premier element
j	indice du second element

2.11.3.2 List<Long> reader.SortLineNumbers.getLinesSorted ( )

Tri les lignes par ordre croissant du numéro de ligne

Renvoie

la liste des lignes triées par ordre croissant du numéro de ligne

2.11.3.3 void reader.SortLineNumbers.quicksort (int *low,* int *high*) [private]

Algorithme de tri rapide

Paramètres

low	indice inferieur non trié
high	indice superieur non trié

2.11.4 Documentation des données membres

2.11.4.1 List<Long> reader.SortLineNumbers.listLignes [private]

Liste des numéros de ligne

La documentation de cette classe a été générée à partir du fichier suivant :
– D :/Documents/workspace/mapreduce/SearchEngine/src/reader/SortLineNumbers.java

Index