MANUALE

PANORAMICA CONCISA E AD ALTO LIVELLO DEL PROGETTO:

Il progetto implementa un sistema software in grado di effettuare il download di articoli da testate giornalistiche online resi disponibili da diverse sorgenti e di estrarre e visualizzare i termini più "importanti" (numero di documenti in cui appaiono) nell'insieme degli articoli scaricati.

Il software supporta due tipologie di sorgenti diverse: file CSV e TheGuardianAPI. È possibile, se necessario, rendere il progetto in grado di supportare nuove sorgenti attraverso l'implementazione di una classe che permette di memorizzare la sorgente in oggetto Java ed il successivo aggiornamento del metodo GenericDeserializer della classe Tools. La modifica consiste nell'aggiunta di un "else if" specificando nella condizione il file path della nuova sorgente.

Il software è strutturato in 3 packages: com.apiguardian, com.apiguardian.bean e it.unipd.dei.eis_MFCD. Nei primi due sopracitati packages sono presenti le seguenti classi: GuardianContentApi, Article, Response e ResponseWrapper. L'insieme di queste classi permette l'implementazione di un client Java per il TheGuardian (Non implementate da noi ma reperito dal seguente link). La classe rimanente, ArticleCSV, è stata sviluppata al fine di permetterci di memorizzare il contenuto di file .csv .
Il terzo package implementa l'interazione utente-applicativo e utente-sistema, il download degli articoli (sia da API che da csv) ed estrazione e conteggio delle parole dagli articoli.
Le interazioni dell'utente vengono principalmente gestite dalle classi InputHandler e ShellManager, mentre il download degli articoli, l'estrazione ed il conteggio delle parole vengono gestite dalla classe Tools. Nello specifico:

- Download scarica gli articoli interrogando la TheGuardianApi oppure effettua la conversione da .csv a .json (questa operazione viene eseguita sulla base dell'opzione selezionata dall'utente);
- Serializer serializza successivamente gli oggetti java contenenti gli articoli in un file .json,
- GenericDeserializer si occupa poi della conversione dei file .json a oggetto java
- Extraction e WordsCounter sono infine utilizzati per l'estrazione ed il conteggio delle parole presenti negli articoli

(Per ulteriori informazioni riguardanti le singole classi e funzioni visionare la javadoc)

ISTRUZIONI SU COME INSTALLARE ED ESEGUIRE IL SOFTWARE:

(Progetto realizzato con Java 8)

Se nella cartella del progetto è presente la cartella target con al suo interno "EISProject-1.0-SNAPSHOT.jar":

- aprire il prompt dei comandi
- cambiare directory entrando nella cartella ProgettoEIS_MFCD con il comando "cd [percorso cartella]"
- eseguire il comando "start exe.bat"

Se nella cartella del progetto non è presente la cartella target o questa non contiene il file "EISProject-1.0-SNAPSHOT.jar":

- aprire il prompt dei comandi
- cambiare directory entrando nella cartella ProgettoEIS_MFCD con “cd [percorso cartella]”
- eseguire il comando “mvn clean install”
- eseguire il comando “start exe.bat”

Per visionare la javadoc, se è presente la cartella target\site\apidocs:

- aprire il file index.html all’interno della cartella target\site\apidocs

Per visionare la javadoc, se non è presente la cartella target\site\apidocs:

- aprire il prompt dei comandi
- cambiare directory entrando nella cartella ProgettoEIS_MFCD con “cd [percorso cartella]”
- eseguire il comando “mvn javadoc:javadoc”
- cambiare directory entrando nella cartella ProgettoEIS_MFCD con “cd [percorso cartella]”
- aprire il file index.html all’interno della cartella target\site\apidocs

---

Per visionare il report dei test, se è presente la cartella target\site e se contiene il file surefire-report.html:

- aprire il file surefire-report.html all’interno di target\site

Per visionare il report dei test, se non è presente la cartella target\site o se non contiene il file surefire-report.html al suo interno:

- aprire il prompt dei comandi
- cambiare directory entrando nella cartella ProgettoEIS_MFCD con “cd [percorso cartella]”
- eseguire il comando “mvn surefire-report:report”
- aprire il file surefire-report.html all’interno di target\site

---

I restanti documenti sono presenti all’interno della cartella “Documenti”

LIBRERIE UTILIZZATE E LE LORO FUNZIONI:

groupId:org.apache.commons
artifactId:commons-csv
version:1.10.0

org.apache.commons.csv.*

- CSVParser
- CSVFormat
- CSVRecord
- getHeaderMap()
- get()

groupId:org.json
artifactId:json
version:20210307

org.json.JSONArray

- JSONArray
- JSONArray()
- put()
- toString()

org.json.JSONObject

- JSONObject
- JSONObject()
- put()

groupId:com.mashape.unirest
artifactId:unirest-java
version:1.4.9

com.mashape.unirest.http.exceptions.UnirestException

- getContent()
- UnirestException

groupId:junit
artifactId:junit
version:4.12

org.junit.Assert.*assertNotEquals*

- assertNotEquals()

junit.framework.TestCase

- TestCase
- assertEquals()
- assertNotNull()
- assertTrue()

org.junit.Assert

- Assert

org.junit.Test

- @Test

groupId:com.fasterxml.jackson.core
artifactId:jackson-databind
version:2.8.8.1

(Utilizzate nelle classi scaricate da content-api-the-guardian)

groupId:io.rest-assured
artifactId:rest-assured
version:3.0.2

(Utilizzate nelle classi scaricate da content-api-the-guardian)

groupId:org.mockito
artifactId:mockito-all
version:1.10.19

(Utilizzate nelle classi scaricate da content-api-the-guardian)

groupId:com.google.code.gson
artifactId:gson
version:2.8.9

com.google.gson.Gson:

- Gson
- Gson()
- toJson()
- fromJson()

com.google.gson.GsonBuilder

- GsonBuilder()

com.google.gson.JsonSyntaxException

- printStackTrace()
- JsonSyntaxException

com.google.gson.JsonIOException

- printStackTrace()
- JsonIOException

groupId:commons-cli
artifactId:commons-cli
version:1.5.0

java.* :

java.awt.*

- Desktop.*getDesktop*()
- open()

java.io.*

- ByteArrayInputStream()
- ByteArrayOutputStream
- ByteArrayOutputStream()
- BufferedReader
- BufferedReader()
- BufferedWriter

- BufferedWriter()
- File
- File()
- FileReader
- FileReader()
- FileWriter
- FileWriter()
- InputStream
- OutputStream()
- PrintStream
- PrintStream()
- delete()
- exists()
- getMessage()
- getName()
- isFile()
- listFiles()
- newLine()
- printStackTrace()
- renameTo()
- toString()
- write()
- IOException

java.nio.file.*

- Files
- StandardCopyOption
- Paths
- Paths()

java.util.*

- Arrays
- ArrayList<>()
- Comparator<>
- Scanner
- Scanner()
- HashMap<>
- Map<>
- SortedMap<>
- TreeMap<>()
- containsKey()
- clear()
- close()
- get()
- hasNext()
- keySet()
- next()

- nextLine()
- put()
- remove()
- replace()
- useDelimiter()

java.lang.reflect.*

java.lang.reflect.Method

- Method
- getDeclaredMethod()
- invoke()

java.lang.reflect.InvocationTargetException

- invoke()
- InvocationTargetException

com.apiguardian.GuardianContentApi:

- GuardianContentApi
- GuardianContentApi()
- getContent()