File di configurazione - bananaconf.json

bananaconf.json è il file di configurazione di bananaSPLIT e permette di "personalizzare" il funzionamento del software.

BananaSPLIT e' stato scritto per lavorare su file provenienti da LexisNexis, quasi sicuramente non funziona con altre fonti.
Non va spostato dalla cartella che contiene il file bananaSPLIT.exe

E' possibile avere piu' file di configurazione precompilati a seconda della lingua e/o del formato specifico del file che si desidera splittare, il nome dei file puo' essere scelto a piacere a patto che abbiano tutti estensione *.json*

Per editare i file di configurazione raccomando di usare Notepad++, il cui installer e' incluso nel file zip. Notepad++ e' un editor di file di testo molto potente che puo' venire comodo in molti casi.
È molto importante rispettare la formattazione del file, quindi evitare di cancellare due punti, virgole, e doppi apicini.
Inoltre ogni elemento della configurazione, se non è l'ultimo della sua sezione (che viene racchiusa gerarchicamente da [] o {} , deve avere una , che lo segue)
I campi racchiusi tra " " sono stringhe, gli altri sono: valori booleani **true,false** o numeri interi **int**. Nel caso si voglia modificare il contenuto di una stringa fare attenzione a mettere sempre le virgolette " "

Le voci di configurazione sono:

- Sorgenti e detinazioni dei file:
    - *"INworkPath": C:\\Esempio\\Ingresso"*
      Percorso in cui cercare I file da divider.
    - *"OUTworkPath": "C:\\Esempio\\Uscita"*
      Percorso in cui salvare I file divisi per articolo e il file con tutti gli articoli di una sorgente

      Nel caso di windows il persorso contiene \ anzi che / quindi bisogna raddoppiarli per evitare errori: **C:\\documenti\\testi\\newyorktimes**

- Nomenclatura dei file degli articoli:
  “OUTworkPath”: "{docnum}_{year}{month:02d}{day:02d}_{filename}_{title}.txt",
  permette di impostare il formato in cui verranno salvati i file contenenti un singolo articolo, per esempio se si sta operando sul file: **The_New_York_Times-2014-1.txt** usando la stringa di esempio i file verranno salvati come:
  3_20140102_the-new-york-times-2014-1_the-getting-was-good.txt
  La stringa si compone dei campi:
    - {docnum} numero progressivo dell’articolo estratto
    - {year} anno di pubblicazione
    - {month} mese di pubblicazione
    - {day} giorno di pubblicazione in numero
    - {filename} nome del file sorgente dell’articolo con ogni spazio e carattere strano sostituito da – per motivi di ordine
    - {title} titolo dell’articolo, ovvero l’ultima riga prima delle parole che identificano l’intestazione con ogni spazio e carattere strano sostituito da – per motivi di ordine
    - {papername} nome possibile della pubblicazione da cui e’ tratto l’articolo, viene cercato nella linea immediatamente superiore alla data di pubblicazione

  I campi possono essere ordinati come si preferisce e separati da qualsiasi carattere permesso nei nomi di file.
  Non è obbligatorio includere tutti i campi nel nome del file di uscita.
  Nel caso dei campi {*docnum},{year},{month:02d},{day:02d}* l’aggiunta della parte dopo i due punti fissa la lunghezza del campo (per esempio Gennaio che sarebbe 1, viene scritto come 01 se si mette :02d o come 0001 se si mette :04d) cosi’ da avere nomi di uguale lunghezza.
  Qualsiasi parte non racchiusa tra parentesi graffe viene riportata tale e quale nel nome del file di uscita.
  Quando si pensa al nome del file fare attenzione alla lunghezza che potrebbe avere, Windows tollera davvero male i nomi lunghi soprattutto quando i file vengono spostati sulle chiavette USB o mandati via mail.
  Facendo un altro esempio, se fosse:
  “OUTworkPath”: "TEST+{year}{month:02d}{day:02d}{docnum:04d}*{filename}.txt",*
  il file verra’ salvato come: TEST+**20140102*0003*the-new-york-times-2014-1.txt**

- Stuttura del file in ingresso:
    - “docSep” template che determina lo stacco tra gli articoli, contiene espressioni regolari, maneggiare con cura. Il default cerca la parola **Copyright** seguita da un anno a 4 cifre: **Copyright 2014**
    - “dateFormat” template per leggere la data dell’articolo, mima la data con dei segnaposto che vengono compilati quando se ne incontra una. Se si vuole modificare la logica è la stessa applicata per il nome dei file in uscita ma vanno per forza inseriti tutti i campi che sono presenti nella data degli articoli che si vuole estrarre, altrimenti non funziona
    - “dateWords” lista dei nomi dei mesi per individuare la data, se vengono usate abbreviazioni o nomi in altra lingua aggiungerli nella lista
    - “headWords” lista di parole che identificano l’inizio di un articolo, modificabile a piacere
    - “tailWords” lista di parole che identificano la fine di un articolo, modificabile a piacere
- Impostazioni Generali:
    - “encoding” codifica del file di ingresso, scelte possibili sono “ascii” o “utf-8” Con “utf-8”si dovrebbe aprire il maggior numero dei file inclusi quelli con caratteri diversi dal set latino standard. Per altri codifiche non sono mai stati effettuati test, quindi non e’ garantito il fuzionamento
    - “monthPosition” indica in quale posizione della data a partire d 0 e’indicato il nome del mese.
      Per esempio se nel file le date sono indicate come: ‘october 30, 2018’ il valore da inserire e’ 0. Se la data fosse ‘30 ottobre, 2018’ il valore sarebbe 1.
      Questo parametro e’ molto importante! Se nel file vengono utilizzati piu’modi per scrivere le date e’ possibile che non tutti gli articoli vengano separati corettamente: **int**
    - “getNewsPaperName” cerca di individuare il nome della pubblicazione cercandolo nella linea immediatamente superiore a quella della data.
      Nel caso non fosse presente usa il parametro “nameNotfoundStr” in sostituzione per completare il nome del file se e’stato specificato nel formato: **true,false**
    - “includeTitle” scrive il titolo dell’articolo nella prima linea del file che lo contiene. Il titolo e ’lo stesso che viene usato nel nome del file se specificato nel formato: **true,false**
    - “maxTitleLen”: lunghezza massima nel nome del fle che puo’ essere occupata dal nome dell’articolo: **int**

- “removeDuplicates” basandosi ESCLUSIVAMENTE sul titolo dell’articolo cerca se nel file ci sono dei duplicati e non li salva. Non funziona tra due file diversi. **true,false**
- “showRemovedDuplicates” stampa a schermo i nomi dei duplicati he pensa di aver trovato, da utilizzare principalmente a scopo di debug: **true,false**
- “loadTXT” per far caricare i file **.txt o .TXT**, scegliere tra i valori: **true,false**
- “loadDOCX” non funziona ancora, lasciare false
- “saveSeparateFiles” salva ogni articolo su un file separato formattato come da sopra: **true,false**
- “saveBodyFile” per ogni file in ingresso ne salva uno in uscita con solo gli articoli, eliminando righe vuote e intestazioni: **true,false**
- “delLF” cancella tutti gli “a capo” dei file, **ATTENZIONE** ogni articolo sara’ tutto su una riga, articoli diversi occupano righe diverse nel bodyFile: **true,false**
- “delWordBreak” riunisce le parole che nel testo sono andate a capo: **true,false**
- “delChars”: stringa dei caratteri da eliminare nel file sorgente prima di dividere gli articoli. Inserire i caratteri che si vogliono eliminare separati dal | (carattere pipe) esempio: **"'|@|#"**