CS102 Course Project, Spring 2018 (Part □)

The goal of this project is to generate a "Word Cloud" for Chinese text.

What is a word cloud? Basically, it is an image displaying the most important components from a text. In English, a word cloud displays words that occur the most often, once you have excluded frequent words that aren't very meaningful (such as "the", "and", "is", these words are called "stop words"), and usually after "stemming" words, which means counting "walk", "walked", and "walking" as the same word, as well as "shoe" and "shoes" or "foot" and "feet". In other words, the "stem" is the common component of different grammatical versions of words (note that for processing Chinese text, stemming is not necessary). You can find many examples of word clouds by searching the web. We provide one below.

In Chinese, we'll be interested by sets of consecutive characters (we often call them "tokens" or "words" or in Chinese"词"), and we'll analyze how often they appear.

The first part of this project consists of the following two tasks:

Task 1 (Preprocessing and Tokenization): The first task is to read a text file of Chinese characters and segment the Chinese characters into words. The program may repeatedly get one part or a whole sentence separated by a punctuation mark, call a word-splitting API (see appendix for references) to split the sentence into words, output the words with a specified delimiter (such as an English character 'a', a punctuation mark ';', a symbol '|', or white space characters like '\n') except all the stop

words. Before reading a Chinese text file, your program may need a parameter to specify the charset used in the text, such as UTF-8, GB2312, GB18030, Big5 etc. In fact, auto-detecting the charset of a plain text file is still an open problem. We provide a Java program called `FileCharsetDetector` (in `fileCharsetDetect.zip`), which can help detect the encoding charset of a text file. Please read the `ReadMe.txt` to learn how to run the program. The source code of `FileCharsetDetector.java` is also given. It might be a working sample to help you deal with charset encoding problem for Chinese text.

Task 2 (Word Frequency Count with sorting): The second task is to read the output words from Task 1, count the frequency of each word (i.e., how many times it appears in the text), sort the words by their frequency (from highest to lowest), and output the sorted words with frequencies. The encoding charset of the output from Task 1 can be fixed in one charset (e.g., you can choose to use UTF-8).

Please note that the two tasks are not sequential. They are expected to be done independently by two different team members as long as there is an agreement on the output format of Task 1 and input format of Task 2. In this way, when you need to test Task 2 before Task 1 is done, you can use mock data, possibly prepared by a third person in your team.

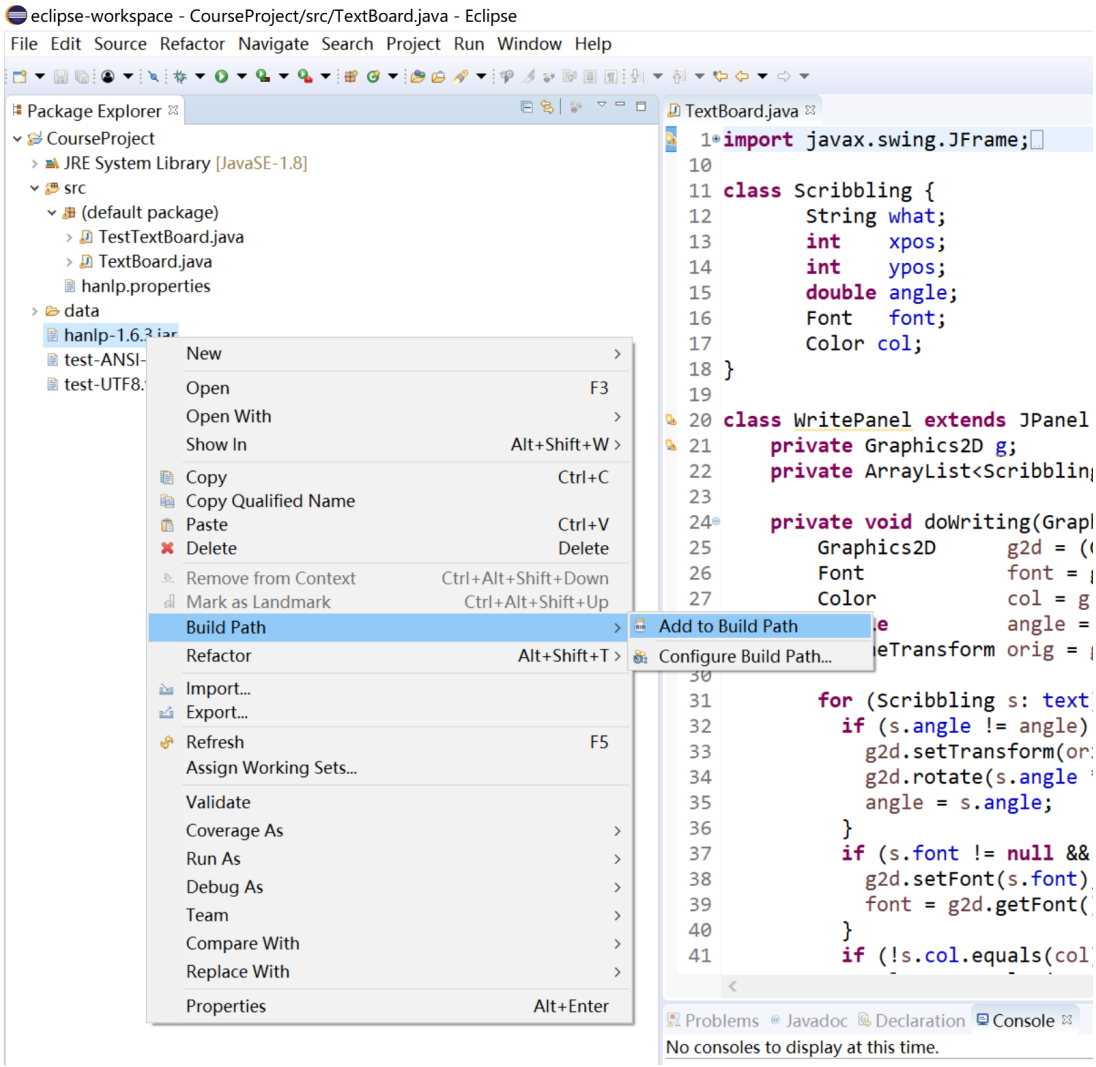
Appendix 1: Chinese Text Processing

HanLP (https://github.com/hankcs/HanLP) is a powerful Java library for Chinese Language Processing. To use HanLP, you need to perform the following setup tasks:

1. Download data.zip (containing various dictionaries and language models) from http://nlp.hankcs.com/download.php?file=data. Unzip the file and put the extracted content in any folder.
2. Download hanlp-release.zip from http://nlp.hankcs.com/download.php?file=jar. After unzipping, you will see a configuration file `hanlp.properties`. Modify the first line of the file to configure the path of the data folder obtained in the first step. For example, if the data folder is at `D:/JavaProjects/HanLP/data`, then the first line of `hanlp.properties` should be:

   ```
   root=D:/JavaProjects/HanLP/
   ```
3. After modification, put the configuration file in the `src` folder of your project (assuming that you are using Eclipse). Eclipse will copy it to the `bin` folder during compilation (i.e., the configuration file will be in `classpath`).

4. The hanlp-release.zip file also contains a jar file (e.g., `hanlp-1.6.3.jar`, your downloaded version may be slightly different, but this would not affect your program's correctness). Put the jar file in your project folder and add it to the project build path in Eclipse (if you cannot see the jar file in Eclipse Package Explorer after you put it in the project folder, refresh the project). See the following screenshot for how to add the jar file to build path in Eclipse.



Note that if you compile and run your project from command line, you should also make sure that the jar file is in `classpath` when invoking the command `javac` and `java`. Read the following Wikipedia page to learn how to do so:

https://en.wikipedia.org/wiki/Classpath_(Java)

After the above setup, you should be able to use HanLP for language processing. For tokenization, please refer to the following demo code:

https://github.com/hankcs/HanLP/blob/master/src/test/java/com/hankcs/demo/DemoSegment.java

For removing stop words, please use the following list of words for string matching:

https://github.com/hankcs/HanLP/blob/master/data/dictionary/stopwords.txt

For more information of HanLP, please read its `README` file at:

https://github.com/hankcs/HanLP

Appendix 2: Team Formation and Project Grading Policy

1. You must work in a team with 2-3 members to finish the project. The ideal team size is 3. For international students, you are encouraged to team up with Chinese students.
2. In principle, ALL team members should be from the same lab. If you want to team up with friends from other labs, please send us an email for confirmation and make sure all of you can attend the project presentation, which will be held in a lab session at the end of this semester.
3. Each team will be asked to give a project presentation. Except for medical reasons (a certificate must be shown in such cases), ALL team members must show up during the presentation and introduce what she/he has done. Anybody not attending the project presentation will only receive 50% of the grade.
4. We will consider the following factors when grading your project: (1) the overall outcome of your project, (2) your contribution to the project, (3) your project presentation quality, and (4) your team size.