**User Manual - Language Detector**

Utility to detect language of unknown file referring language sample files provided.

**Language Problem**

You are given a number of files from a number of different human languages. All languages use the Latin alphabet A to Z; case can be ignored and the only punctuation characters are . , ; : and single space characters separating words. For example, you might have the files FRENCH.1, GAELIC.2 to GAELIC.9, ENGLISH.1 to ENGLISH.6 and so on.

Given a file TEXT.txt in the same format but of unknown origin, Develop a program to identify its language.

**Solution**

* Utility LanguageDetector is a command line utility that can detect language for a unknown file provided language sample files. There is jUnit test also that can run this utility.
* More sample language files for a language result in better matching percentage.
* Punctuation chars are ignored as same Latin character languages. But they are effectively removed to ensure words around them are picked.
* Best matching language is picked as the winner. It is possible that more than one language has some matching percentage.
* Maximum test coverage for this utility is ensured.
* Sample language files are created in ‘samples’ folder.
* Unknown language files are also created for test purpose – UNKNOWN.txt, text.txt. Available at app root.

**Assumptions:**

* Unknown language file and directory of language sample files is provided as command line arguments
* No validation is performed on language sample files.

**Build**

* Run “mvn clean package” in the app root folder on command prompt.
* To run this utility independently, ‘jar-with-dependencies’ maven build plugin is used. Though maven also create independent jar.
* Utility jars will be created in target folder.
* Java 7 is used in development. Also latest utility libraries used.

**Usage**

* Download the langdetect.zip.util and rename it to langdetect.zip. Named due to security reasons while transferring over gmail.
* Unzip the utility to local folder.
* Either run command from target folder on command line.
* usage (image reference 1 below) :-

java –classpath <<path to single langdetect jar>> **gov.nbn.langdetect.LanguageDetector**

-f <<filename with path of unknown file>>

-s <<path for sample language files>>

E.g. from root folder: -

java -classpath D:\Development\workspace\langdetect\target\langdetect-1.0.0-jar-with-dependencies.jar gov.nbn.langdetect.LanguageDetector -f text.txt -s samples

* Or configure the project in Eclipse and Run test LanguageDetector.testMain() (Image reference 2 below)
  + Set filename with path for UNKNOWN text file
  + Set directory path for sample language files.
  + Run test and get results in console window.
* Check results on console or in log file ‘langdetect.log’ in execution folder.

**Result**

* Utility will pick the language having maximum matching percentage.
* Result are logged and displayed on console like:

*D:\Development\workspace\langdetect>java -classpath D:\Development\workspace\langdetect\target\langdetect-1.0.0-jar-with-dependencies.jar gov.nbn.langdetect.LanguageDetector -f UNKNOWN.txt -s samples*

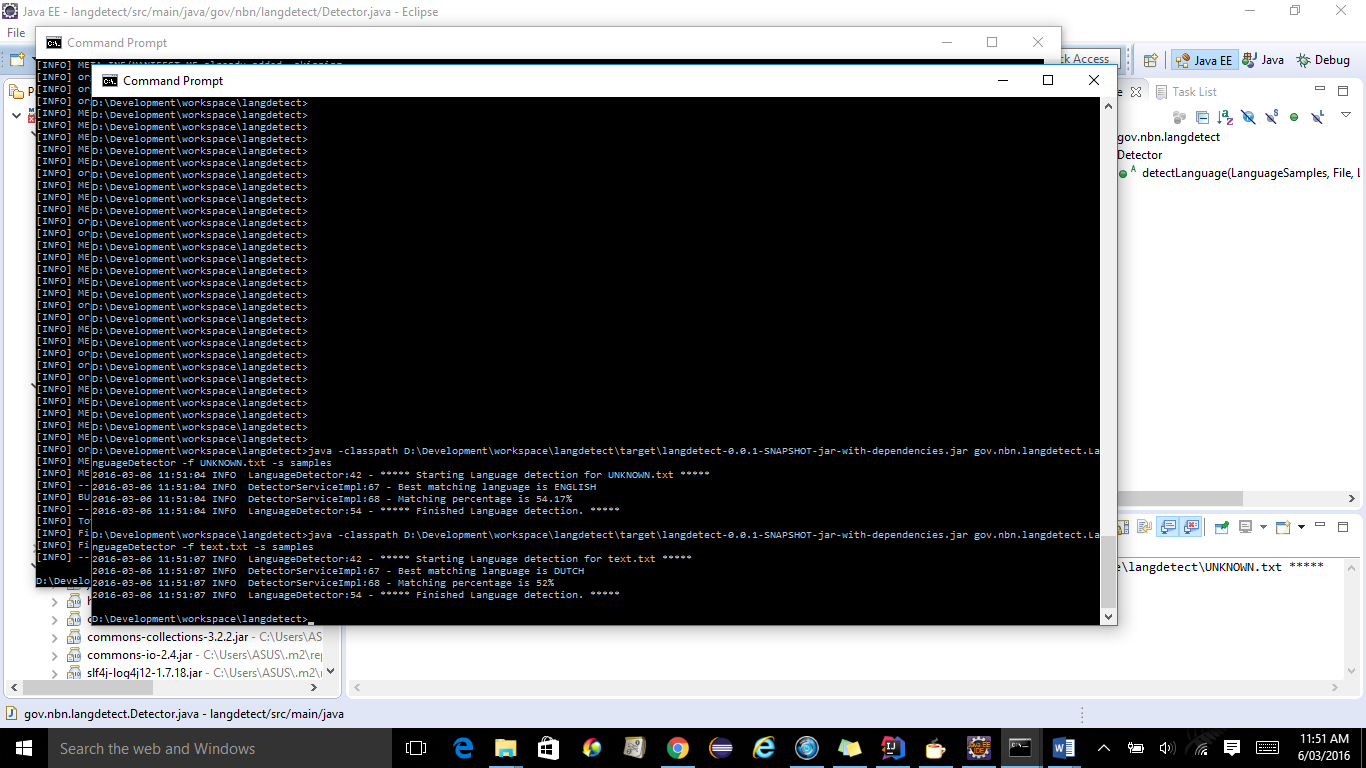
*2016-03-06 12:30:39 INFO LanguageDetector:42 - \*\*\*\*\* Starting Language detection for UNKNOWN.txt \*\*\*\*\**

*2016-03-06 12:30:39 INFO DetectorServiceImpl:68 - Best matching language is ENGLISH*

*2016-03-06 12:30:39 INFO DetectorServiceImpl:69 - Matching percentage is 57.14%*

*2016-03-06 12:30:39 INFO LanguageDetector:54 - \*\*\*\*\* Finished Language detection. \*\*\*\*\**

**Image reference 1**



**Image reference 2**

