A Report

on

<u>**"OpenNLP"**</u>

<u>**"Text Mining Tool"**</u>

Submitted by

Ajinkya Kulkarni 33126

Yash Kulkarni 33129

Shubham Loya 33133

Sahil Naphade 33140

Under the guidance of

Mrs D.D.Londhe

Department Of Information Technology

Pune Institute of Computer Technology College of Engineering

Sr. No 27, Pune-Satara Road, Dhankawadi, Pune - 411 043.

2019-2020

Introduction

Apache OpenNLP is an open-source Java library which is used to process natural language text. We can build an efficient text processing service by making use of this library. OpenNLP comes with the services such as tokenization , part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution, etc. These tasks are usually required to build more advanced text processing services.

In 2010, OpenNLP entered the Apache incubation. In 2011, Apache OpenNLP 1.5.2 Incubating was released, and in the same year, it graduated as a top-level Apache project and In 2015, OpenNLP was 1.6.0 released.

Purpose

openNLP project is created in order to perform the above mentioned tasks .One of the main advantages of using openNLP is that it is provided with a large number of pre-built models inclusive of many languages , along with other text resources where these models are built from.

Overview

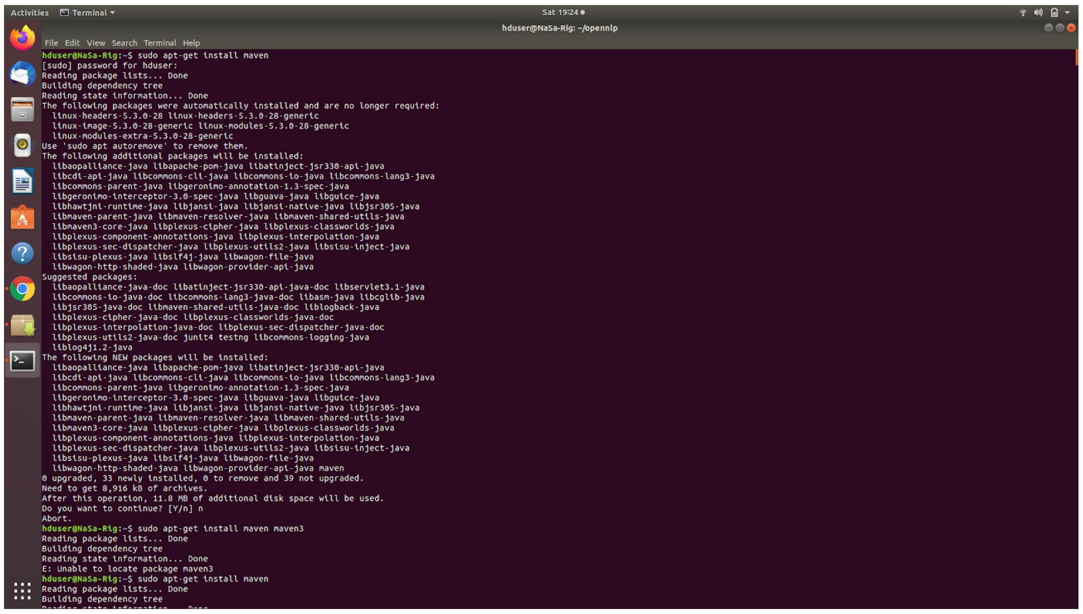
Apache OpenNLP is an open-source library that provides solutions to some of the Natural Language Processing tasks through its APIs and command line tools. Apache OpenNLP uses a machine learning approach for the tasks of processing natural language.

Named Entity Recognition:

Named Entity Recognition is to find named entities like person, place, organisation or a thing in a given sentence. OpenNLP has built models for NER which can be directly used and also helps in training a model for the custom data we have.

Installation steps

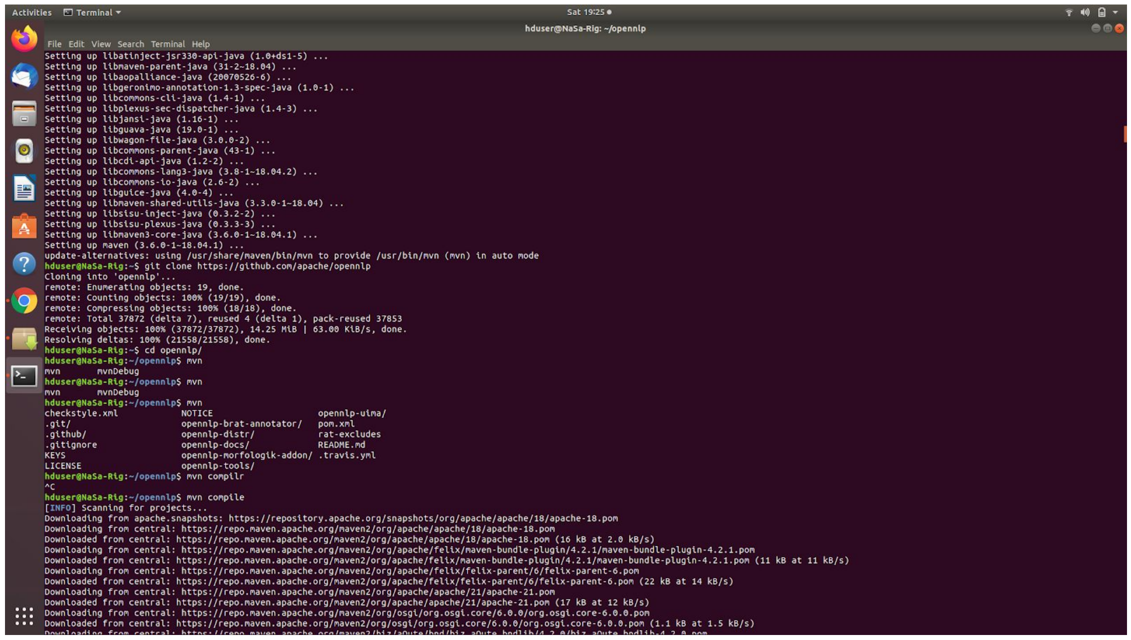
1. Install git.
   a. sudo apt-get install git
2. Install maven
   a. Sudo apt-get install maven



```
hduser@NaSa-Rig:~$ sudo apt-get install maven
[sudo] password for hduser:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-5.3.0-28 linux-headers-5.3.0-28-generic
  linux-image-5.3.0-28-generic linux-modules-5.3.0-28-generic
  linux-modules-extra-5.3.0-28-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
  libcommons-parent-java libgeronimo-annotation-1.3-spec-java
  libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java
  libhawtjni-runtime-java libjansi-java libjansi-native-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java
  libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java
  libplexus-component-annotations-java libplexus-interpolation-java
  libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java
  libsisu-plexus-java libslf4j-java libwagon-file-java
  libwagon-http-shaded-java libwagon-provider-api-java
Suggested packages:
  libaopalliance-java-doc libatinject-jsr330-api-java-doc libservlet3.1-java
  libcommons-io-java-doc libcommons-lang3-java-doc libasm-java libcglib-java
  libjsr305-java-doc libmaven-shared-utils-java-doc liblogback-java
  libplexus-cipher-java-doc libplexus-classworlds-java-doc
  libplexus-interpolation-java-doc libplexus-sec-dispatcher-java-doc
  libplexus-utils2-java-doc junit4 testng libcommons-logging-java
  liblog4j1.2-java
The following NEW packages will be installed:
  libaopalliance-java libapache-pom-java libatinject-jsr330-api-java
  libcdi-api-java libcommons-cli-java libcommons-io-java libcommons-lang3-java
  libcommons-parent-java libgeronimo-annotation-1.3-spec-java
  libgeronimo-interceptor-3.0-spec-java libguava-java libguice-java
  libhawtjni-runtime-java libjansi-java libjansi-native-java libjsr305-java
  libmaven-parent-java libmaven-resolver-java libmaven-shared-utils-java
  libmaven3-core-java libplexus-cipher-java libplexus-classworlds-java
  libplexus-component-annotations-java libplexus-interpolation-java
  libplexus-sec-dispatcher-java libplexus-utils2-java libsisu-inject-java
  libsisu-plexus-java libslf4j-java libwagon-file-java
  libwagon-http-shaded-java libwagon-provider-api-java maven
0 upgraded, 33 newly installed, 0 to remove and 39 not upgraded.
Need to get 8,916 kB of archives.
After this operation, 11.8 MB of additional disk space will be used.
Do you want to continue? [Y/n] n
Abort.
hduser@NaSa-Rig:~$ sudo apt-get install maven maven3
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package maven3
hduser@NaSa-Rig:~$ sudo apt-get install maven
Reading package lists... Done
Building dependency tree
```

Activities
Terminal
Sat 19:24
hduser@NaSa-Rig: ~/opennlp
File Edit View Search Terminal Help

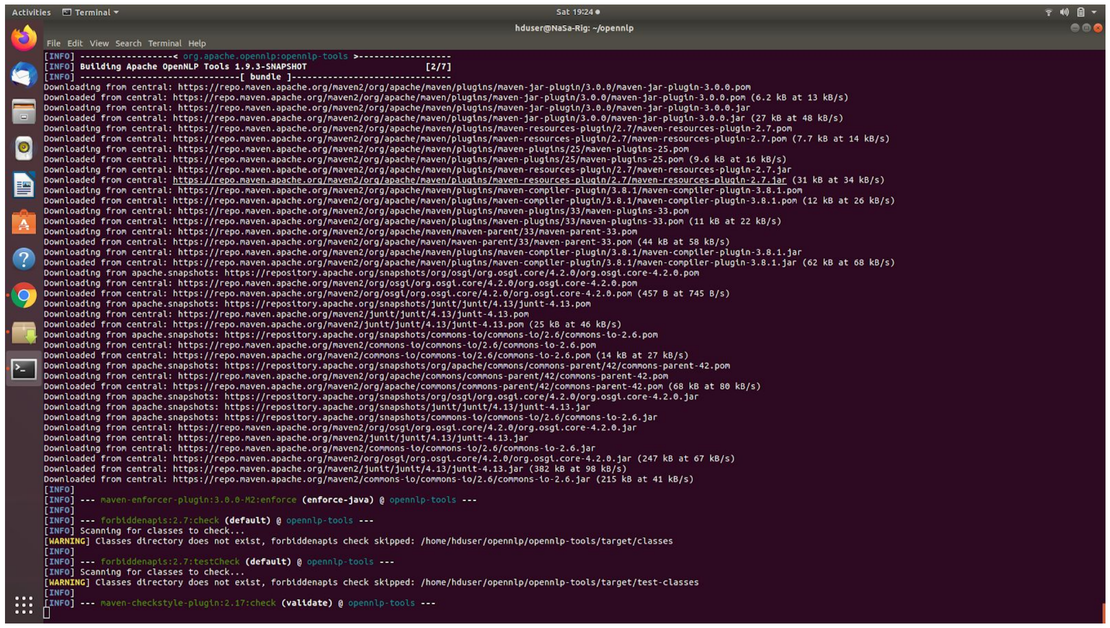
3. Clone repository to opennlp
   git clone https://github.com/apache/opennlp
4. Go to the cloned directory
   cd opennlp/7



```
Setting up libatinject-jsr330-api-java (1.0+ds1-5) ...
Setting up libmaven-parent-java (31-2~18.04) ...
Setting up libaopalliance-java (20070526-6) ...
Setting up libgeronimo-annotation-1.3-spec-java (1.0-1) ...
Setting up libcommons-cli-java (1.4-1) ...
Setting up libplexus-sec-dispatcher-java (1.4-3) ...
Setting up libjansi-java (1.16-1) ...
Setting up libguava-java (19.0-1) ...
Setting up libwagon-file-java (3.0.0-2) ...
Setting up libcommons-parent-java (43-1) ...
Setting up libcdi-api-java (1.2-2) ...
Setting up libcommons-lang3-java (3.8-1~18.04.2) ...
Setting up libcommons-io-java (2.6-2) ...
Setting up libguice-java (4.0-4) ...
Setting up libmaven-shared-utils-java (3.3.0-1~18.04) ...
Setting up libsisu-inject-java (0.3.2-2) ...
Setting up libsisu-plexus-java (0.3.3-3) ...
Setting up libmaven3-core-java (3.6.0-1~18.04.1) ...
Setting up maven (3.6.0-1~18.04.1) ...
update-alternatives: using /usr/share/maven/bin/mvn to provide /usr/bin/mvn (mvn) in auto mode
hduser@NaSa-Rig:~$ git clone https://github.com/apache/opennlp
Cloning into 'opennlp'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 37872 (delta 7), reused 4 (delta 1), pack-reused 37853
Receiving objects: 100% (37872/37872), 14.25 MiB | 63.00 KiB/s, done.
Resolving deltas: 100% (21558/21558), done.
hduser@NaSa-Rig:~$ cd opennlp/
hduser@NaSa-Rig:~/opennlp$ mvn
mvn        mvnDebug
hduser@NaSa-Rig:~/opennlp$ mvn
mvn        mvnDebug
hduser@NaSa-Rig:~/opennlp$ mvn
checkstyle.xml            NOTICE                     opennlp-uima/
.git/                     opennlp-brat-annotator/    pom.xml
.github/                  opennlp-distr/             rat-excludes
.gitignore                opennlp-docs/              README.md
KEYS                      opennlp-morfologik-addon/  .travis.yml
LICENSE                   opennlp-tools/
hduser@NaSa-Rig:~/opennlp$ mvn compilr
^C
hduser@NaSa-Rig:~/opennlp$ mvn compile
[INFO] Scanning for projects...
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/apache/apache/18/apache-18.pom
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/18/apache-18.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/18/apache-18.pom (16 kB at 2.0 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/felix/maven-bundle-plugin/4.2.1/maven-bundle-plugin-4.2.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/felix/maven-bundle-plugin/4.2.1/maven-bundle-plugin-4.2.1.pom (11 kB at 11 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/felix/felix-parent/6/felix-parent-6.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/felix/felix-parent/6/felix-parent-6.pom (22 kB at 14 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/21/apache-21.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/21/apache-21.pom (17 kB at 12 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/6.0.0/org.osgi.core-6.0.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/6.0.0/org.osgi.core-6.0.0.pom (1.1 kB at 1.5 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/biz/aQute/bnd/biz.aQute.bndlib/4.2.0/biz.aQute.bndlib-4.2.0.pom
```

5. Run command

    mvn compile



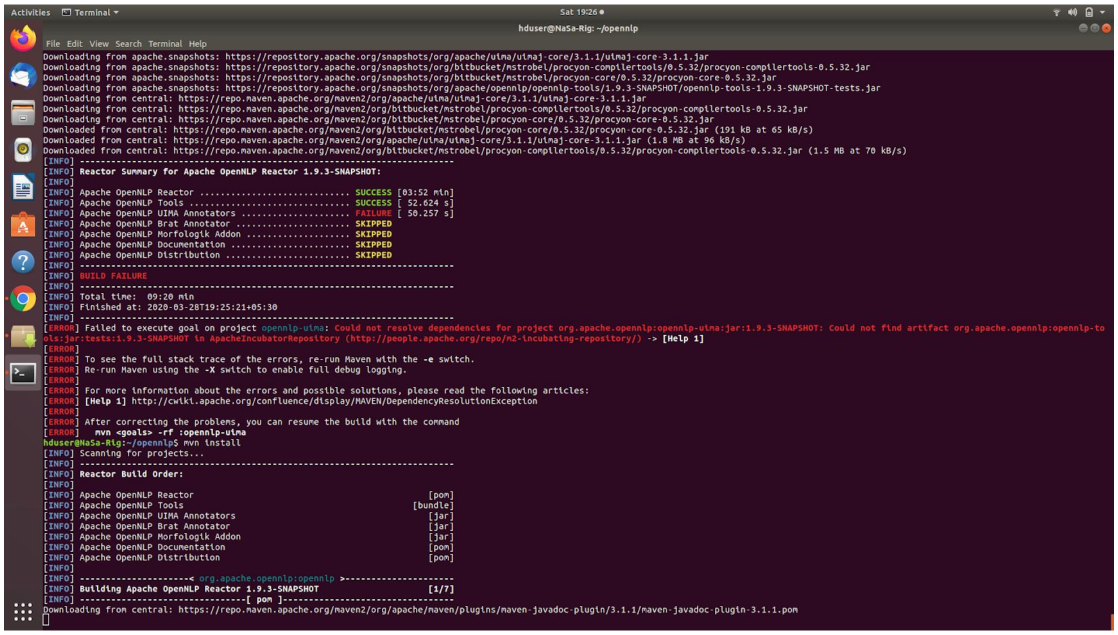
```
[INFO] ------------------< org.apache.opennlp:opennlp-tools >------------------
[INFO] Building Apache OpenNLP Tools 1.9.3-SNAPSHOT                        [2/7]
[INFO] ------------------------------[ bundle ]--------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-jar-plugin/3.0.0/maven-jar-plugin-3.0.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-jar-plugin/3.0.0/maven-jar-plugin-3.0.0.pom (6.2 kB at 13 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-jar-plugin/3.0.0/maven-jar-plugin-3.0.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-jar-plugin/3.0.0/maven-jar-plugin-3.0.0.jar (27 kB at 48 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.7/maven-resources-plugin-2.7.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.7/maven-resources-plugin-2.7.pom (7.7 kB at 14 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/25/maven-plugins-25.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/25/maven-plugins-25.pom (9.6 kB at 16 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.7/maven-resources-plugin-2.7.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.7/maven-resources-plugin-2.7.jar (31 kB at 34 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.8.1/maven-compiler-plugin-3.8.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.8.1/maven-compiler-plugin-3.8.1.pom (12 kB at 26 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/33/maven-plugins-33.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/33/maven-plugins-33.pom (11 kB at 22 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/33/maven-parent-33.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/33/maven-parent-33.pom (44 kB at 58 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.8.1/maven-compiler-plugin-3.8.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-compiler-plugin/3.8.1/maven-compiler-plugin-3.8.1.jar (62 kB at 68 kB/s)
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.pom
Downloading from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.pom (457 B at 745 B/s)
Downloading from apache.snapshots: https://repository.apache.org/snapshots/junit/junit/4.13/junit-4.13.pom
Downloading from central: https://repo.maven.apache.org/maven2/junit/junit/4.13/junit-4.13.pom
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/4.13/junit-4.13.pom (25 kB at 46 kB/s)
Downloading from apache.snapshots: https://repository.apache.org/snapshots/commons-io/commons-io/2.6/commons-io-2.6.pom
Downloading from central: https://repo.maven.apache.org/maven2/commons-io/commons-io/2.6/commons-io-2.6.pom
Downloaded from central: https://repo.maven.apache.org/maven2/commons-io/commons-io/2.6/commons-io-2.6.pom (14 kB at 27 kB/s)
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/apache/commons/commons-parent/42/commons-parent-42.pom
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-parent/42/commons-parent-42.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-parent/42/commons-parent-42.pom (68 kB at 80 kB/s)
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.jar
Downloading from apache.snapshots: https://repository.apache.org/snapshots/junit/junit/4.13/junit-4.13.jar
Downloading from apache.snapshots: https://repository.apache.org/snapshots/commons-io/commons-io/2.6/commons-io-2.6.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/junit/junit/4.13/junit-4.13.jar
Downloading from central: https://repo.maven.apache.org/maven2/commons-io/commons-io/2.6/commons-io-2.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/osgi/org.osgi.core/4.2.0/org.osgi.core-4.2.0.jar (247 kB at 67 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/4.13/junit-4.13.jar (382 kB at 98 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-io/commons-io/2.6/commons-io-2.6.jar (215 kB at 41 kB/s)
[INFO]
[INFO] --- maven-enforcer-plugin:3.0.0-M2:enforce (enforce-java) @ opennlp-tools ---
[INFO]
[INFO] --- forbiddenapis:2.7:check (default) @ opennlp-tools ---
[INFO] Scanning for classes to check...
[WARNING] Classes directory does not exist, forbiddenapis check skipped: /home/hduser/opennlp/opennlp-tools/target/classes
[INFO]
[INFO] --- forbiddenapis:2.7:testCheck (default) @ opennlp-tools ---
[INFO] Scanning for classes to check...
[WARNING] Classes directory does not exist, forbiddenapis check skipped: /home/hduser/opennlp/opennlp-tools/target/test-classes
[INFO]
[INFO] --- maven-checkstyle-plugin:2.17:check (validate) @ opennlp-tools ---
```

It will start compiling the source code of OpenNLP.

6. Run command

    On the first checkout everything should be built. Type on terminal

    mvn install


Activities
Terminal
Sat 19:26
hduser@NaSa-Rig: ~/opennlp
File Edit View Search Terminal Help

```
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/apache/uima/uimaj-core/3.1.1/uimaj-core-3.1.1.jar
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/bitbucket/mstrobel/procyon-compilertools/0.5.32/procyon-compilertools-0.5.32.jar
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/bitbucket/mstrobel/procyon-core/0.5.32/procyon-core-0.5.32.jar
Downloading from apache.snapshots: https://repository.apache.org/snapshots/org/apache/opennlp/opennlp-tools/1.9.3-SNAPSHOT/opennlp-tools-1.9.3-SNAPSHOT-tests.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/uima/uimaj-core/3.1.1/uimaj-core-3.1.1.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/bitbucket/mstrobel/procyon-compilertools/0.5.32/procyon-compilertools-0.5.32.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/bitbucket/mstrobel/procyon-core/0.5.32/procyon-core-0.5.32.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/bitbucket/mstrobel/procyon-core/0.5.32/procyon-core-0.5.32.jar (191 kB at 65 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/uima/uimaj-core/3.1.1/uimaj-core-3.1.1.jar (1.8 MB at 96 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/bitbucket/mstrobel/procyon-compilertools/0.5.32/procyon-compilertools-0.5.32.jar (1.5 MB at 70 kB/s)
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Apache OpenNLP Reactor 1.9.3-SNAPSHOT:
[INFO]
[INFO] Apache OpenNLP Reactor ............................. SUCCESS [03:52 min]
[INFO] Apache OpenNLP Tools ............................... SUCCESS [ 52.624 s]
[INFO] Apache OpenNLP UIMA Annotators ..................... FAILURE [ 50.257 s]
[INFO] Apache OpenNLP Brat Annotator ...................... SKIPPED
[INFO] Apache OpenNLP Morfologik Addon .................... SKIPPED
[INFO] Apache OpenNLP Documentation ....................... SKIPPED
[INFO] Apache OpenNLP Distribution ........................ SKIPPED
[INFO] ------------------------------------------------------------------------
[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  09:20 min
[INFO] Finished at: 2020-03-28T19:25:21+05:30
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal on project opennlp-uima: Could not resolve dependencies for project org.apache.opennlp:opennlp-uima:jar:1.9.3-SNAPSHOT: Could not find artifact org.apache.opennlp:opennlp-to
ols:jar:tests:1.9.3-SNAPSHOT in ApacheIncubatorRepository (http://people.apache.org/repo/m2-incubating-repository/) -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/DependencyResolutionException
[ERROR]
[ERROR] After correcting the problems, you can resume the build with the command
[ERROR]   mvn <goals> -rf :opennlp-uima
hduser@NaSa-Rig:~/opennlp$ mvn install
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] Apache OpenNLP Reactor                                             [pom]
[INFO] Apache OpenNLP Tools                                            [bundle]
[INFO] Apache OpenNLP UIMA Annotators                                     [jar]
[INFO] Apache OpenNLP Brat Annotator                                      [jar]
[INFO] Apache OpenNLP Morfologik Addon                                    [jar]
[INFO] Apache OpenNLP Documentation                                       [pom]
[INFO] Apache OpenNLP Distribution                                        [pom]
[INFO]
[INFO] --------------------< org.apache.opennlp:opennlp >---------------------
[INFO] Building Apache OpenNLP Reactor 1.9.3-SNAPSHOT                     [1/7]
[INFO] --------------------------------[ pom ]---------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-javadoc-plugin/3.1.1/maven-javadoc-plugin-3.1.1.pom
```

to build all modules. The build installs the maven artifacts in the local repository and creates a binary and source distribution inside the opennlp-distr/target folder.

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text.

This toolkit is written completely in Java and provides support for common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, coreference resolution, language detection and more!

These tasks are usually required to build more advanced text processing services.

The goal of the OpenNLP project is to be a mature toolkit for the above mentioned tasks.

An additional goal is to provide a large number of pre-built models for a variety of languages, as well as the annotated text resources that those models are derived from.

Presently, OpenNLP includes common classifiers such as Maximum Entropy, Perceptron and Naive Bayes.

OpenNLP can be used both programmatically through its Java API or from a terminal through its CLI. OpenNLP API can be easily plugged into distributed streaming data pipelines like Apache Flink, Apache NiFi, Apache Spark.

Features:

Following are the notable features of OpenNLP –

- Translation: In NLP, Translation helps in translating one language into another.

- Summarize: Using the summarize feature, you can summarize Paragraphs, articles, documents or their collection in NLP.

- Searching: In OpenNLP, a given search string or its synonyms can be identified in a given text, even though the given word is altered or misspelled.

- Natural Language Generation: It is used for generating information from a database and automating the information reports such as weather analysis or medical reports.

- Speech recognition: Though it is difficult to analyse human speech, NLP has some built in features for this requirement.

- Named Entity Recognition (NER): OpenNLP supports NER, using which you can extract names of locations, people and things even while processing queries.

- Information grouping: This option in NLP groups the textual information in the content of the document, just like Parts of speech.

❑ Feedback Analysis: As the name implies, various types of feedback from people are collected, regarding the products, by NLP to analyse how well the product is successful in winning their hearts.

❑ Tagging (POS): Tagging in NLP is used to divide the text into various grammatical elements for further analysis.

Overview

Currently the library has different packages:

- opennlp-tools : The core toolkit.
- opennlp-uima : A set of Apache UIMA annotators.
- opennlp-brat-annotator : A set of annotators for BRAT
- opennlp-morfologik-addon : An addon for Morfologik
- opennlp-sandbox: Other projects in progress are found in the sandbox

Services in detail

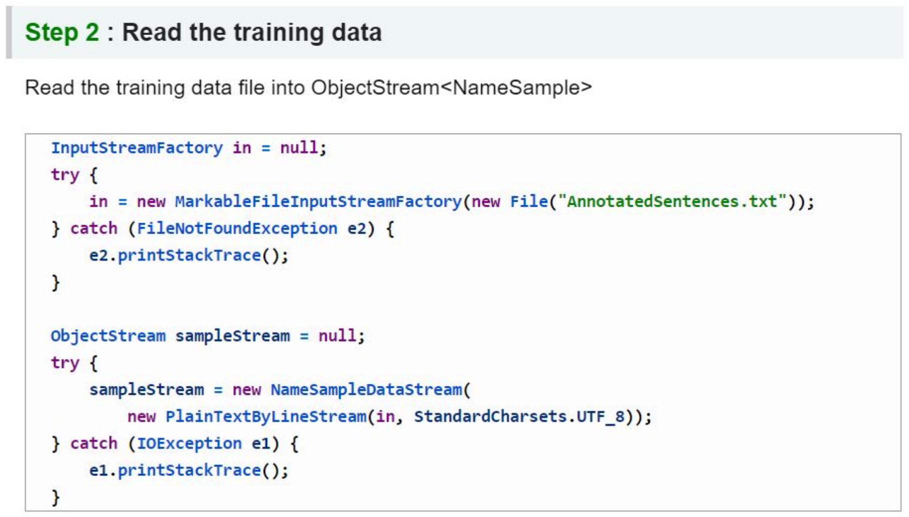
1) Training in OpenNLP:

Following steps are carried out in order to generate the models for training data-

❑ **Prepare training data**

As sugguested by OpenNLP manual[https://opennlp.apache.org/docs/1.8.0/manual/opennlp.html#tools.namefind.training], at least 15,000 sentences should be available in the training file, so that the trained model may perform well.

❑ Read training data



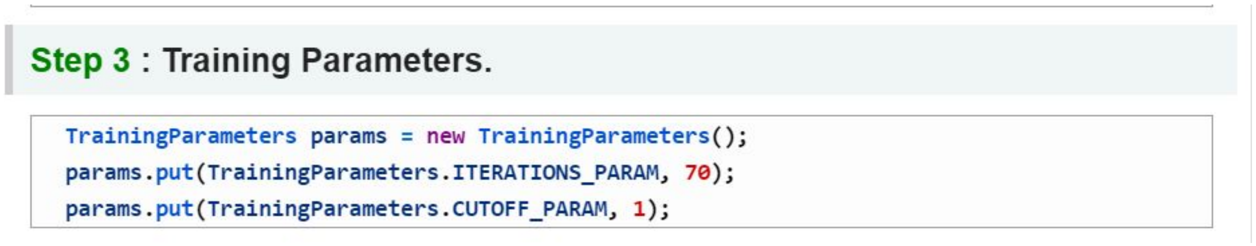
Step 2 : Read the training data

Read the training data file into ObjectStream<NameSample>

```
InputStreamFactory in = null;
try {
    in = new MarkableFileInputStreamFactory(new File("AnnotatedSentences.txt"));
} catch (FileNotFoundException e2) {
    e2.printStackTrace();
}

ObjectStream sampleStream = null;
try {
    sampleStream = new NameSampleDataStream(
        new PlainTextByLineStream(in, StandardCharsets.UTF_8));
} catch (IOException e1) {
    e1.printStackTrace();
}
```

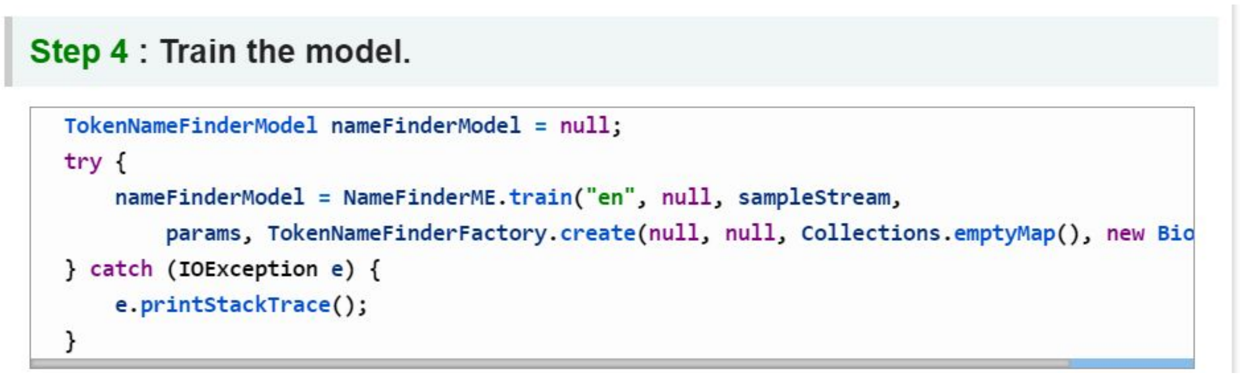
❑ Training parameters



Step 3 : Training Parameters.

```
TrainingParameters params = new TrainingParameters();
params.put(TrainingParameters.ITERATIONS_PARAM, 70);
params.put(TrainingParameters.CUTOFF_PARAM, 1);
```

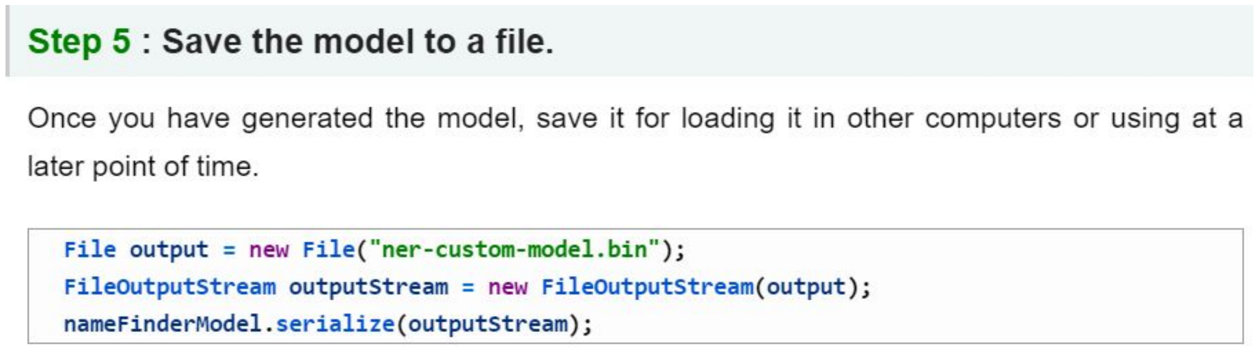
❑ Train model



Step 4 : Train the model.

```
TokenNameFinderModel nameFinderModel = null;
try {
    nameFinderModel = NameFinderME.train("en", null, sampleStream,
        params, TokenNameFinderFactory.create(null, null, Collections.emptyMap(), new Bio
} catch (IOException e) {
    e.printStackTrace();
}
```

❑ Save model



Step 5 : Save the model to a file.

Once you have generated the model, save it for loading it in other computers or using at a later point of time.

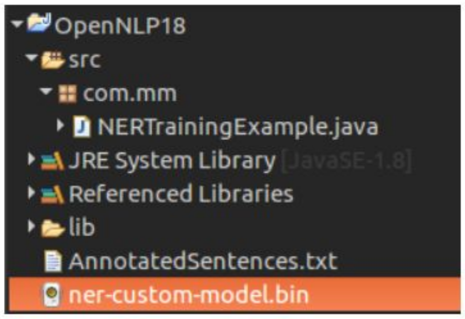
```
File output = new File("ner-custom-model.bin");
FileOutputStream outputStream = new FileOutputStream(output);
nameFinderModel.serialize(outputStream);
```

❑ Test program

To verify the program, use the model and predict the types from a sentence.

❑ output:

Once the program is run, the model is saved to "ner-custom-model.bin" as shown in the following screenshot.



Model saved to ner-custom-model.bin

2)Tokenization:

To tokenize the given sentences into simpler fragments, the OpenNLP library provides three different classes –

- SimpleTokenizer – This class tokenizes the given raw text using character classes.
- WhitespaceTokenizer – This class uses whitespaces to tokenize the given text.
- TokenizerME – This class converts raw text into separate tokens. It uses Maximum Entropy to make its decisions.

Example input is :

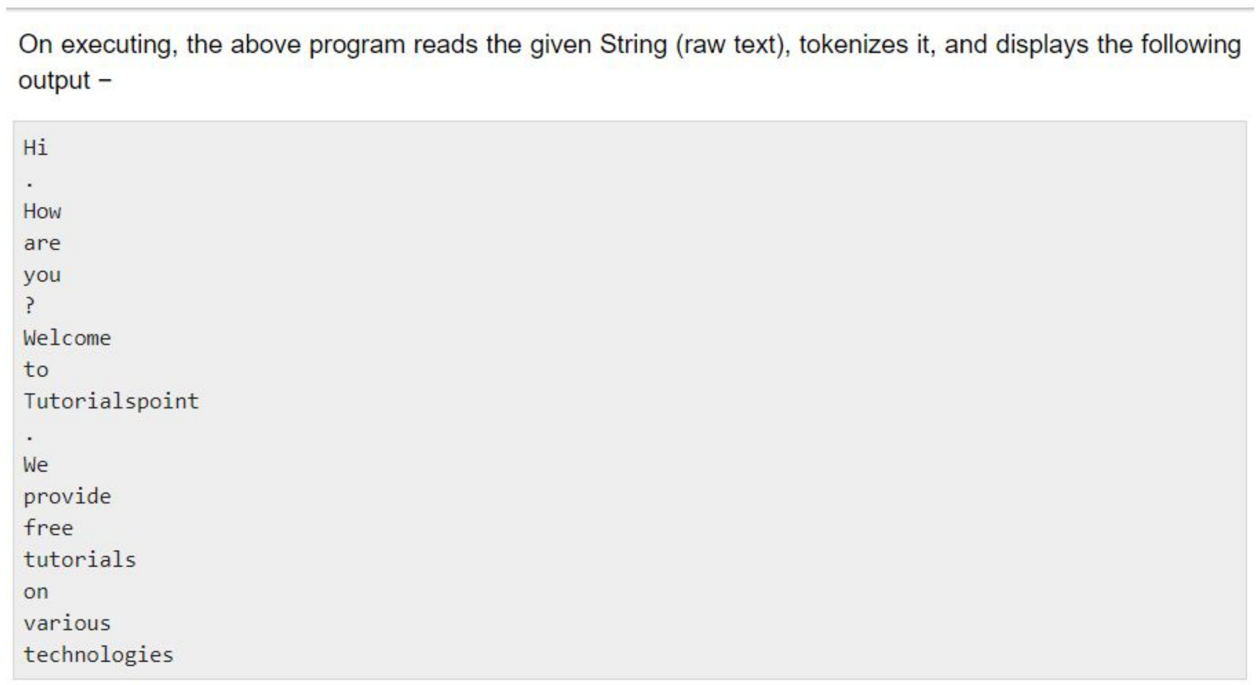
```
String sentence = "Hi. How are you? Welcome to Tutorialspoint. "

         + "We provide free tutorials on various technologies";
```

SimpleTokenizer:

To tokenize a sentence using the SimpleTokenizer class, you need to –

- Create an object of the respective class.
- Tokenize the sentence using the tokenize() method.
- Print the tokens.



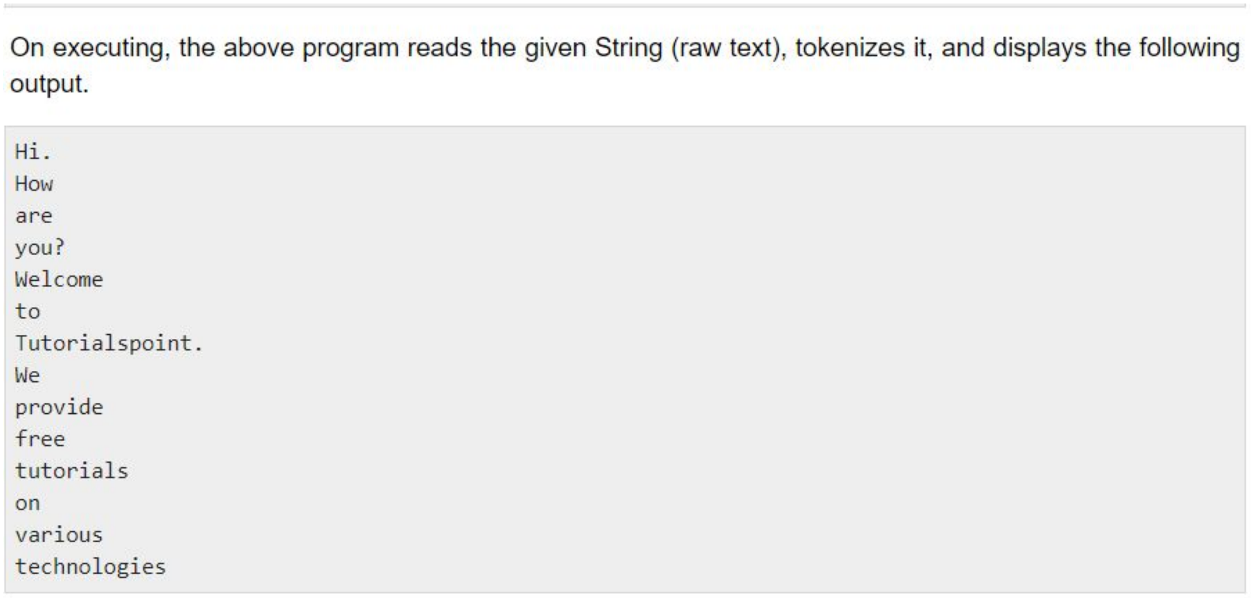
On executing, the above program reads the given String (raw text), tokenizes it, and displays the following output –

```
Hi
.
How
are
you
?
Welcome
to
Tutorialspoint
.
We
provide
free
tutorials
on
various
technologies
```

WhitespaceTokenizer:

To tokenize a sentence using the WhitespaceTokenizer class, you need to –

- Create an object of the respective class.
- Tokenize the sentence using the tokenize() method.
- Print the tokens.



On executing, the above program reads the given String (raw text), tokenizes it, and displays the following output.

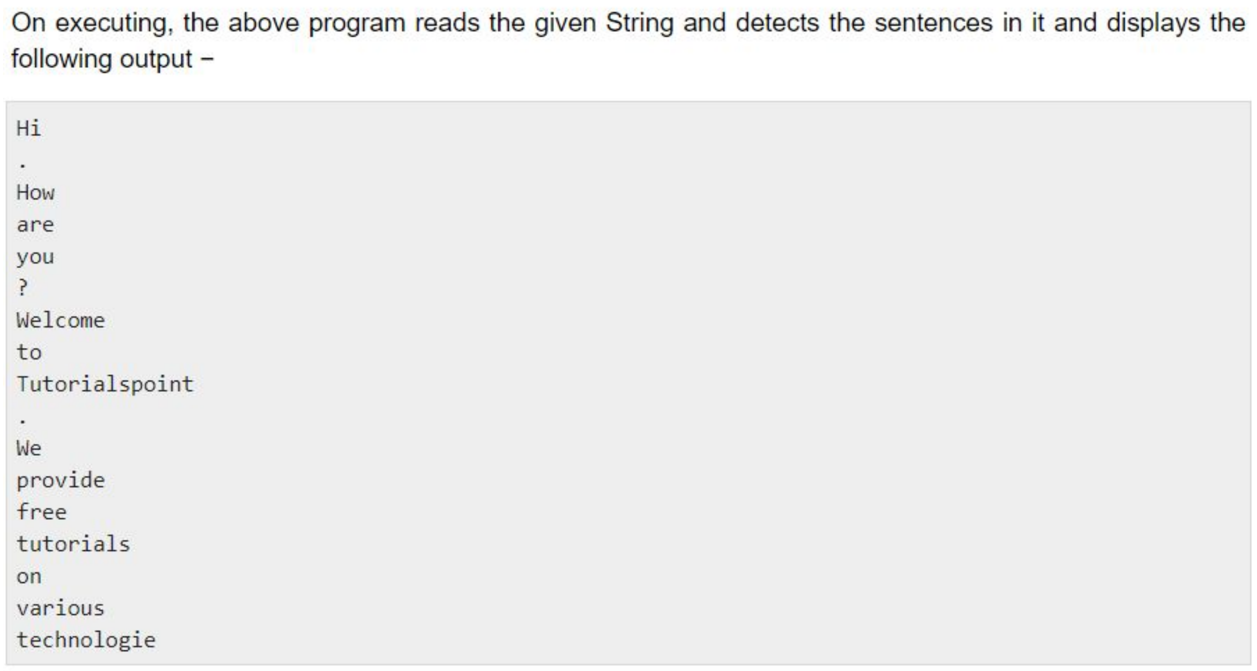
```
Hi.
How
are
you?
Welcome
to
Tutorialspoint.
We
provide
free
tutorials
on
various
technologies
```

TokenizerME class:

OpenNLP also uses a predefined model, a file named de-token.bin, to tokenize the sentences. It is trained to tokenize the sentences in a given raw text.

The TokenizerME class of the opennlp.tools.tokenizer package is used to load this model, and tokenize the given raw text using OpenNLP library. To do so, you need to –

- Load the en-token.bin model using the TokenizerModel class.
- Instantiate the TokenizerME class.
- Tokenize the sentences using the tokenize() method of this class.



On executing, the above program reads the given String and detects the sentences in it and displays the following output –

```
Hi
.
How
are
you
?
Welcome
to
Tutorialspoint
.
We
provide
free
tutorials
on
various
technologie
```

<u>3)Sentence detection:</u>

The OpenNLP Sentence Detector can detect that a punctuation character marks the end of a sentence or not. In this sense a sentence is defined as the longest white space trimmed character sequence between two punctuation marks. The first and last sentence make an exception to this rule. The first non whitespace character is assumed to be the begin of a sentence, and the last non whitespace character is assumed to be a sentence end.

Example sentence:

"Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29. Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group. Rudolph Agnew, 55 years old and former chairman of Consolidated Gold Fields PLC, was named a director of this British industrial conglomerate."

After detecting the sentence boundaries each sentence is written in its own line.

“Pierre Vinken, 61 years old, will join the board as a non executive director Nov. 29.Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group. Rudolph Agnew, 55 years old and former chairman of Consolidated Gold Fields PLC, was named a director of this British industrial conglomerate.”

Getting Started

You can import the core toolkit directly from Maven, SBT or Gradle:

Maven

```
<dependency>
  <groupId>org.apache.opennlp</groupId>
  <artifactId>opennlp-tools</artifactId>
  <version>${opennlp.version}</version>
</dependency>
```

SBT

```
libraryDependencies += "org.apache.opennlp" % "opennlp-tools" % "${opennlp.version}"
```

Gradle

```
compile group: "org.apache.opennlp", name: "opennlp-tools", version: "${opennlp.version}"
```

For more details please check our documentation

Building OpenNLP

At least JDK 8 and Maven 3.3.9 are required to build the library.

After cloning the repository go into the destination directory and run:

```
mvn install
```

steps to obtain the tags pragmatically in java using apache openNLP:

Step 1 : Tokenize the given input sentence into tokens.

Step 2 : Read the parts-of-speech maxent model, “en-pos-maxent.bin” into a stream.

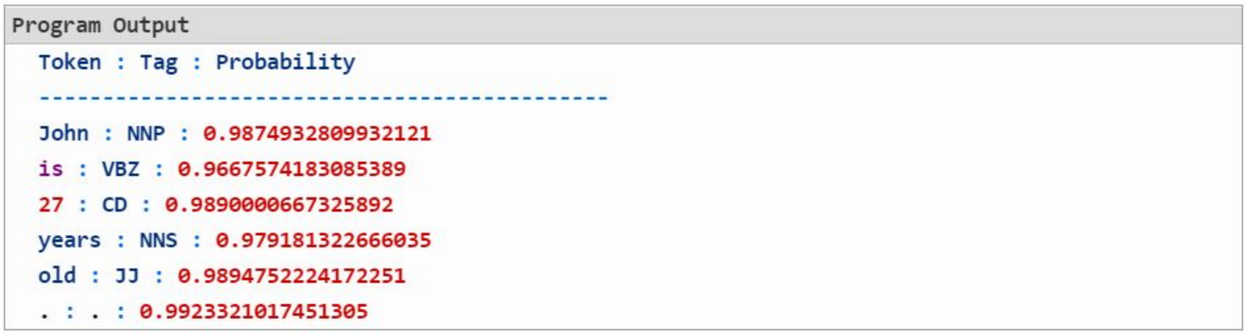
Step 3 : Read the stream into parts-of-speech model, POSModel.

Step 4 : Load the model into parts-of-speech tagger, POSTaggerME

Step 5 : Grab the tags using the method POSTaggerME.tag(), and probability for the tag to be given using the method PosTaggerME.probs();

Step 6 : Finally, print what we got, the token, their respective tags and probabilities of the tags.

Sample output:



```
Program Output
  Token : Tag : Probability
  ---------------------------------------------
  John : NNP : 0.9874932809932121
  is : VBZ : 0.9667574183085389
  27 : CD : 0.9890000667325892
  years : NNS : 0.979181322666035
  old : JJ : 0.9894752224172251
  . : . : 0.9923321017451305
```

Useful Links

For additional information, visit the OpenNLP Home Page

You can use OpenNLP with any language, demo models are provided here.

The models are fully compatible with the latest release, they can be used for testing or getting started.

Please train your own models for all other use cases.

Documentation, including JavaDocs, code usage and command-line interface examples are available here

You can also follow our mailing lists for news and updates.

References

1)https://www.tutorialkart.com/opennlp/ner-training-in-opennlp-with-name-finder-training-java-example/

2)https://www.tutorialspoint.com/opennlp/opennlp_tokenization.htm

3)https://opennlp.apache.org/docs/1.5.3/manual/opennlp.html

4)https://www.tutorialkart.com/opennlp/pos-tagger-example-in-apache-opennlp/

5)https://www.tutorialkart.com/opennlp/opennlp-overview/