Assignments 3

Index:

I design 8 fields in my mapping:

Name	Type	Description
id	Text	Add f before the filename, such as f06_11.xml
name	Text	The name in the case report
URL	Text	The AustLII in the case report
person	Text	All the people found in the case report
location	Text	All the location found in the case report
organization	Text	All the organization found in the case report
catchphrase	Text	All the catchphrase in the case report
sentence	Text	All the sentence in the case report

Solution:

build.sbt:

Scalaversion 2.11.12

LibraryDependencies:

Spark-core 2.4.3 Scalaj-http 2.4.2 Scala-parser-combinators 1.1.2

caseindex.scala:

First, I create the index and the mapping use http request by scalaj library

Second, from the inupt directory I filter out the files which end with .xml and combine these files' name with comma.

```
val Files = path.listFiles().filter{ f => f.getName.endsWith(".xml")}.toList.mkString(",")
```

Then, load these files in a RDD with one block(due to nlp does not support parallel computing, so must set to 1 block),and set the format of it as xml.

```
val SCFile = sc.wholeTextFiles(Files,1)
val XMLFile = SCFile.map(line => (new File(line._1.trim()).getName(),XML.loadString(line._2)))
```

Then, I define a function called XmlProc(a:(String, scala.xml.Elem)):String, which can filter the name and url and combine the catchphrase and sentence, then sending the sentence to the NLP server and analyse the feedback result. Filter out the person, location and organization. Last post all results in JsonString then return it. The return format is as following:

```
val FinalJson =
    s"""{"id":"$FileName","name":"$name",
        "url":"$url",
        "person":"$person",
        "location":"$location",
        "organization":"$organization",
        "catchphrases":"$catchphrase",
        "sentence":"$sentence"}"""
```

With this function I can process my RDD data from input XML to I design output JSON format

```
val Result = XMLFile.map(line => (line._1,XmlProc(line)))
```

Finally, put the final result to the Elasticsearch:

```
Http("http://localhost:9200/legal_idx/cases/"+x._1+"?pretty").postData(x._2).method("PUT").header("Content-Type", "application/json").timeout(connTimeoutMs = 1000, readTimeoutMs = 60000).asString
```

** due to the default timeout is not suitable for the searching in NLP server, So in http command with scalaj, I set the read timeout to 60000 ms, and conntimeout as default 1000.

Running Command:

```
spark-submit –packages "org.scalaj:scalaj-http_2.11:2.4.2,org.scala-lang.modules:scala-parser-combinators_2.13:1.1.2" --class "caseindex" --master local[2] target/scala-2.11/caseindex_2.11-1.0.jar cases_test
```

Query example:

```
curl -X GET “http://localhost:9200/legal_idx/cases/_search?pretty&q=location:Melbourne”
```

```
curl -X GET
“http://localhost:9200/legal_idx/cases/_search?pretty&q=location:Victoria%20Melbourne”
```

```
curl -X GET
“http://localhost:9200/legal_idx/cases/_search?pretty&q=organization:Property%20Services%20Council%20of%20Australia”
```

```
curl -X GET
“http://localhost:9200/legal_idx/cases/_search?pretty&q=person:john%20peroustianis”
```