Install ElasticSearch and Kibana and set the system environment variables properly. Then cd into the directory /bin and start ES/Kibana with elasticsearch.bat and kibana.bat. When ES starts, open Kibana (localhost: 5601), and use Dev Tools on the side bar to test the result.

Create the maven project:

\*Update the pom.xml before starting, add plugin and install dependencies (groupId/artifactId)

org.elasticsearch.client/elasticsearch-rest-high-level-client -- 6.5.4

org.elasticsearch/elasticsearch – 6.5.4

org.apache.logging.log4j/log4j-api -- 2.11.1 ---- get the log information

org.apache.logging.log4j/log4j-core – 2.11.1 ---- get the log information

org.apache.storm/storm-elasticsearch – 1.2.2

org.json/json – 20180813 – used to get json obejct

\*Create log4j2.properties file in main/src/java/resources

Content (copy):

appender.console.type = Console  
appender.console.name = console  
appender.console.layout.type = PatternLayout  
rootLogger.level = info  
rootLogger.appenderRef.console.ref = console

1. Creating index in ES with custom mappings:

\*Create mapping in Kibana Dev Tools directly:

e.g.

https://www.elastic.co/guide/en/elasticsearch/reference/current/mapping.html

\*Use Java API

e.g.

**IndexResponse response = client.prepareIndex (“your\_index\_name”, “your\_index\_type”,”your\_index\_id”)**

**.setSource (XContentFactory.jsonBuilder ()**

**.startObject ()**

**.field (“field\_name\_1”, “field\_value\_1”)**

**.field (“field\_name\_2”,”field\_value\_2”)**

**.endObject ())**

**.get ();**

The ES client in the above method should be initiated first then you can pass it like an argument:

**Settings settings = Settings.builder ()**

**.put (“cluster.name”, “ealsticsearch”)**

**.build ();**

**TransportClient client = new PreBuiltTransportClient (settings)**

**.addTransportAddress (new TransportAddress (InetAddress.getByName (“localhost”), 9300));**

**client.close ();**

In Kibana, try GET /your\_index\_name/your\_index\_type/your\_index\_id to see whether the index has been created. It should return a json like string.

1. Loading files (JSON) into ES index:

If you have a lot of files and they all have the same mapping, use the ES Bulk API.

First, get files’ paths from the directory, and attach the path name into a list/ArrayList.

Then, loop through the list, go into each json files and make the json string become json object. After having the object, we can use something like object.getString to get the value of one field.

In the jsonToEs method, use Bulk API to push multiple request at the same time

e.g.

**try {**

**BulkRequestBuilder bulkRequestBuider = client.prepareBulk ();**

**bulkRequestBuilder.add (client.prepareIndex (“your\_index\_name”,”your\_index\_type”, Integer.toString (IdNumber))**

**.serSource (jsonBuilder ()**

**.startObject () //create custom mapping**

**.field (“ name ”, “ value ”)**

**.field (“ name ”, “ value ”)**

**.endObject ()**

**)**

**);**

**BulkResponse bulkResponse = bulkRequestBuilder.get ();**

**} catch (FileNotFoundException e) {**

**e.printStackTree ();**

**}**

1. Loading index data from ES into Java:

First, connect to ES like before, then set the destination json file path –- filePath

e.g.

**SearchRequestBuilder builder = client.preapreSearch (“your\_index\_name”);**

**builder.setType (“your\_index\_type”);**

**builder.setSize (“the\_size\_you\_want\_pass\_one\_time”);**

**builder.setScroll (new TimeValue (6000));**

**SearchResponse scrollResp = builder.execute ().actionGet ();**

**try {**

**BufferedWriter out = new BufferedWriter (new FileWriter (filePath, true));**

**while (true) {**

**for (SearchHit hit: scrollResp.getHits ().getHits ()) {**

**String json = hit.getSourceAsString ();**

**out.write (json);**

**out.write (“\r\n”);**

**}**

**ScrollResp = client.preapreSearchScroll (scrollResp.getScrollId ())**

**.getScroll (new TimeValue(6000).execute().actionGet());**

**}**

**out.close ();**

**}catch ……**

1. Loading script fields from ES into Java:

First, connect to ES client.

Then, use UpdateRequest to update the mapping, like:

**UpdateRequest updateRequest = new UpdateRequest (“the\_index\_name”, “the\_index\_type”,”the\_index\_id”).doc (jsonBuilder (). StartObject .field (“new\_field\_name”, “new\_field\_value”).endObject ());**

**client.update (updateRequest).get ();**

Next, update the original value using UpdateRequest.

e.g.

**UpdateRequest updateRequest = new UpdateRequest (**

**“your\_index\_name”,**

**“your\_index\_value”,**

**“index\_id”**

**) ;**

**updateRequest. script (new Script (“things\_you\_want\_to\_change\_here”));**

**client.update (updateRequest).get ();**