

Lius 1.0 Tutorial by examples

By : *Rida Benjelloun*

Rida.benjelloun@doculibre.com

Table of contents

1- Factory indexing.....	2
2- Mixed indexing.....	5
3- XML Node Indexing.....	7
4- JavaBeans indexing.....	8
5- Search.....	9

1- Factory indexing

The factory indexing allows to index XML, HTML, Word, Excel, PowerPoint, RTF, PDF, ZIP, Txt, OpenOffice and MP3 files. It uses the Lius configuration file and the Mime type to determine which indexer to use with the document.

Example 1 : Indexing with the factory:

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\test.pdf";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = IndexerFactory.getIndexer(toIndex, lc);

indexer.index(indexDir);
```

Note : version 1.0-RC2 automatically add the file's path in field « filePath ».

Example 2 : Adding custom fields

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\test.pdf";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = IndexerFactory.getIndexer(toIndex, lc);

//Add custom fields
List fields = new ArrayList();
fields.add(new Field( "fieldTest", "Test new field", Field.Store.YES,
Field.Index.TOKENIZED));

indexer.index(indexDir, fields);
```

Example 3 : Lius custom configuration file allowing to index a PDF file

For more examples, it is recommended to have a look at the file liusconfig.xml located in the « config » folder of the Lius distribution.

```
<?xml version="1.0" encoding="UTF-8" ?>
<uceneIndex>
<properties>
  <analyzer class="org.apache.lucene.analysis.lius.unicode.UTF8AccentRemoverAnalyzer" />
  <createIndex value="auto" />
  <indexWriterProperty mergeFactor="10" maxMergeDocs="100" optimize="true" />
</properties>
<index>
  <pdf setBoost="1.6">
    <indexer class="lius.index.PDF.PdfIndexer">
      <mime>application/pdf</mime>
    </indexer>
    <fields>
      <uceneField name="fullText" get="content" type="Text" />
      <uceneField name="title" get="title" type="Text" />
      <uceneField name="author" get="author" type="Text" />
      <uceneField name="creator" get="creator" type="Text" />
      <uceneField name="summary" get="summary" type="Text" />
      <uceneField name="keywords" get="keywords" type="Text" />
      <uceneField name="subject" get="subject" type="Text" />
    </fields>
  </pdf>
</index>
<search>
  <multiFieldQueryParser>
    <searchFields sep=",">title,author,creator,summary,fullText</searchFields>
  </multiFieldQueryParser>
</search>
<searchResult>
  <fieldsToDisplay setHighlighter="true">
    <uceneField name="title" label="title" />
    <uceneField name="subject" label="subject" />
    <uceneField name="creator" label="creator" />
    <uceneField name="filePath" label="File path" />
    <uceneField name="fullText" label="full text" setFragmenter="50" />
  </fieldsToDisplay>
</searchResult>
</uceneIndex>
```

Example 4: Indexing subdirectories

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
```

```
String toIndex = "c:\\files\\directory";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

LuceneActions.getSingletonInstance().indexSubDirectories(toIndex, indexDir, lc);
```

Example 5 : Extracting text from a document

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\test.pdf";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = IndexerFactory.getIndexer(toIndex, lc);

String text = Indexer.getContent();
```

Example 6 : Retrieving an object of type lucene Document from Lius

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\test.pdf";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = IndexerFactory.getIndexer(toIndex, lc);
Collection liusFields = indexer.getPopulatedLiusFields();

Document luceneDoc =
    LuceneActions.getSingletonInstance().populateLuceneDoc(liusFields);
```

2- Mixed indexing

Mixed indexing is a by directory indexing and consists in stocking in the same Lucene document (record) all the documents which are located in the given directory. It is pretty useful when the document or documentary unit is composed of many files (example: integral text in PDF and metadatas in an XML file)..

Example 7 : Example of mixed indexing

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\testDir";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = new MixedIndexer();
indexer.setUp(lc);
indexer.setMixedContentsObj(toIndex);
indexer.index(indexDir);
```

Example 8 : Example of a mixed configuration file supporting both PDF and XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<uceneIndex>
<properties>
  <analyzer class="org.apache.lucene.analysis.lius.unicode.UTF8AccentRemoverAnalyzer" />
  <createIndex value="auto" />
  <indexWriterProperty mergeFactor="10" maxMergeDocs="100" optimize="true" />
</properties>
<index>
<!-- Mixed indexing tag -->
<mixedIndexing>
  <fields>
    <uceneField mimeType="text/xml"/>
    <uceneField mimeType="application/pdf"/>
  </fields>
</mixedIndexing>

<!-- Xml indexing -->
<xml>
  <xmlFile ns="http://purl.org/dc/elements/1.1/" setBoost="2.0">
    <indexer class="lius.index.xml.XmlFileIndexer">
      <mime>text/xml</mime>
    </indexer>
```

```
<fields>
  <.luceneField name="title" xpathSelect="//dc:title" type="Text" >
  <.luceneField name="subject" xpathSelect="//dc:subject" type="Keyword"/>
  <.luceneField name="creator" xpathSelect="//dc:creator" type="Text"/>
</fields>
</xmlFile>
</xml>
<!-- Pdf indexing -->
<pdf setBoost="1.6">
  <indexer class="lius.index.PDF.PdfIndexer">
    <mime>application/pdf</mime>
  </indexer>
  <fields>
    <.luceneField name="fullText" get="content" type="Text" />
  </fields>
</pdf>
</index>
<search>
  <multiFieldQueryParser>
    <searchFields sep=",">title ,creator ,fullText</searchFields>
  </multiFieldQueryParser>
</search>
<searchResult>
  <fieldsToDisplay setHighlighter="true">
    <.luceneField name="title" label="title" />
    <.luceneField name="creator" label="creator" />
    <.luceneField name="fullText" label="full text" setFragmenter="50" />
    <.luceneField name="filePath" label="File path" />
  </fieldsToDisplay>
</searchResult>
</luceneIndex>
```

3- XML Node Indexing

XML node indexing allows to divide the XML document in many nodes et to stock them in distinct records in the index.

Example 9 : Node indexing example

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";
String toIndex = "c:\\files\\testNodes.xml";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

Indexer indexer = new XmlNodeIndexer();
indexer.setUp(lc);
indexer.setStreamToIndex(new FileInputStream(toIndex));
indexer.index(indexDir);
```

Example 10 : Node indexing example configuration file

```
<?xml version="1.0" encoding="UTF-8" ?>
<uceneIndex>
<properties>
  <analyzer class="org.apache.lucene.analysis.lius.unicode.UTF8AccentRemoverAnalyzer" />
  <createIndex value="auto" />
  <indexWriterProperty mergeFactor="10" maxMergeDocs="100" optimize="true" />
</properties>
<index>
  <xml>
    <xmlNodes>
      <node select="//book">
        <uceneField name="title" xpathSelect="title" type="Text" />
        <uceneField name="creator" xpathSelect="creator" type="Text" />
      </node>
    </xmlNodes>
  </xml>
</index>
<search>
  <multiFieldQueryParser>
    <searchFields sep=",">title, creator </searchFields>
  </multiFieldQueryParser>
</search>
<searchResult>
  <fieldsToDisplay setHighlighter="true">
    <uceneField name="title" label="title" />
  </fieldsToDisplay>
</searchResult>
```

```
<.luceneField name="creator" label="creator" />
</fieldsToDisplay>
</searchResult>
</luceneIndex>
```

4- JavaBeans indexing

The JavaBean indexer allows to index Java objects.

Example 11 : JavaBean indexing example

```
String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

// Bean personne
Personne personne = new Personne();
personne.setNom("Benjelloun");
personne.setPrenom("Rida");
personne.setAdresse("Quebec Canada");

Indexer indexer = new BeanIndexer();
indexer.setUp(lc);
indexer.setObjectToIndex(personne);
indexer.index(indexDir);
```

Example 12 : Example configuration file for indexing JavaBeans

```
<?xml version="1.0" encoding="UTF-8" ?>
<luceneIndex>
<properties>
  <analyzer class="org.apache.lucene.analysis.lius.unicode.UTF8AccentRemoverAnalyzer" />
  <createIndex value="auto" />
  <indexWriterProperty mergeFactor="10" maxMergeDocs="100" optimize="true" />
</properties>
<index>
<JavaBeans>
  <Bean class="lius.test.beans.Personne">
    <luceneField name="nom" getMethod="getNom" type="Text"/>
    <luceneField name="prenom" getMethod="getPrenom" type="Text"/>
    <luceneField name="adresse" getMethod="getAdresse" type="Text"/>
  </Bean>
</JavaBeans>
</index>
```

```

<search>
  <multiFieldQueryParser>
    <searchFields sep=",">nom, prenom, adresse </searchFields>
  </multiFieldQueryParser>
</search>
<searchResult>
  <fieldsToDisplay setHighlighter="true">
    <luceneField name="nom" label="Nom" />
    <luceneField name="prenom" label="Prenom" />
    <luceneField name="adresse" label="Adresse" />
  </fieldsToDisplay>
</searchResult>
</luceneIndex>

```

5- Search

Example 13 : Example of search on a single index

```

String indexDir = "c:\\index" ;
String log4j = "c:\\lius\\config\\log4j.properties";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

LiusHitList ls = SearchIndex.search("canada", indexDir, liusConfig);

System.out.println("Nb doc = " + ls.size());
for (int i = 0; i < ls.size(); i++) {
  LiusHit lh = (LiusHit) ls.get(i);
  System.out.println("====*****====");
  System.out.println("Score = " + lh.getScore());
  System.out.println("Doc id = " + lh.getDocId());
  List liusHitFields = lh.getLiusFields();
  for (int j = 0; j < liusHitFields.size(); j++) {
    LiusField lf = (LiusField) liusHitFields.get(j);
    String name = lf.getLabel();
    String[] values = lf.getValues();
    System.out.print(name + " : ");
    for (int k = 0; k < values.length; k++) {
      System.out.println("\t" + values[k]);
    }
  }
  System.out.println("====");
}
}

```

Example 14 : Example of search on many indexes

```
String indexDir = "c:\\index" ;
String indexDir2 = "c:\\index2" ;
String log4j = "c:\\lius\\config\\log4j.properties";

LiusLogger.setLoggerConfigFile(log4j);
LiusConfig lc = LiusConfigBuilder.getSingletonInstance().getLiusConfig(liusConfig);

String[] indexes = { indexDir, indexDir2 };
LiusHitList res = SearchIndex.multiIndexSearch("maroc", indexes, liusConfig);

System.out.println("Nb doc = " + ls.size());
for (int i = 0; i < ls.size(); i++) {
    LiusHit lh = (LiusHit) ls.get(i);
    System.out.println("====*****====");
    System.out.println("Score = " + lh.getScore());
    System.out.println("Doc id = " + lh.getDocId());
    List liusHitFields = lh.getLiusFields();
    for (int j = 0; j < liusHitFields.size(); j++) {
        LiusField lf = (LiusField) liusHitFields.get(j);
        String name = lf.getLabel();
        String[] values = lf.getValues();
        System.out.print(name + " : ");
        for (int k = 0; k < values.length; k++) {
            System.out.println("\t" + values[k]);
        }
    }
    System.out.println("====");
}
}
```