

## 原 LIRe 源代码分析 2：基本接口（DocumentBuilder）

2013年10月31日 19:24:27 阅读数：6219

=====

LIRe源代码分析系列文章列表：

[LIRe 源代码分析 1：整体结构](#)

[LIRe 源代码分析 2：基本接口（DocumentBuilder）](#)

[LIRe 源代码分析 3：基本接口（ImageSearcher）](#)

[LIRe 源代码分析 4：建立索引（DocumentBuilder）\[以颜色布局为例\]](#)

[LIRe 源代码分析 5：提取特征向量\[以颜色布局为例\]](#)

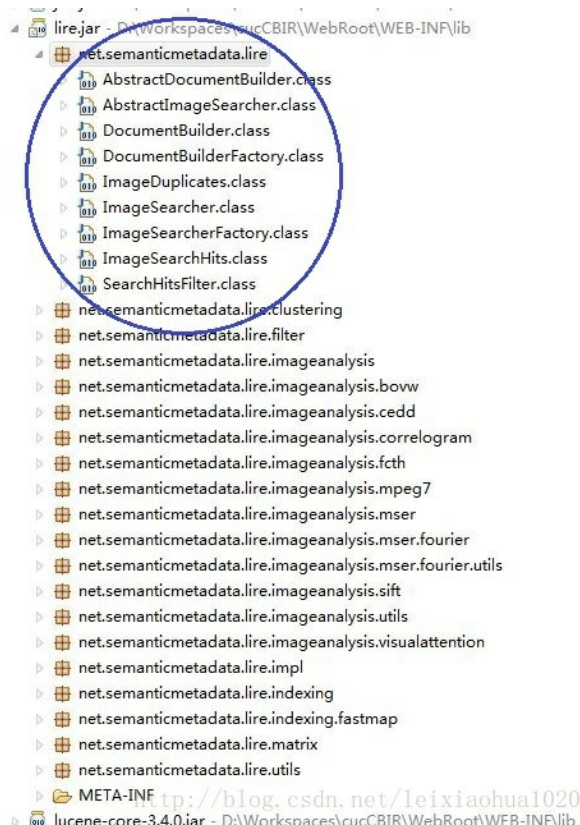
[LIRe 源代码分析 6：检索（ImageSearcher）\[以颜色布局为例\]](#)

[LIRe 源代码分析 7：算法类\[以颜色布局为例\]](#)

=====

本文分析LIRe的基本接口。LIRe的基本接口完成的工作不外乎两项：生成索引和检索。生成索引就是根据图片提取特征向量，然后存储特征向量到索引的过程。检索就是根据输入图片的特征向量到索引中查找相似图片的过程。

LIRe的基本接口位于net.semanticmetadata.lire的包中，如下图所示：



将这些接口分为2类：

DocumentBuilder：用于生成索引

ImageSearcher：用于检索

下面来看看与DocumentBuilder相关的类的定义：

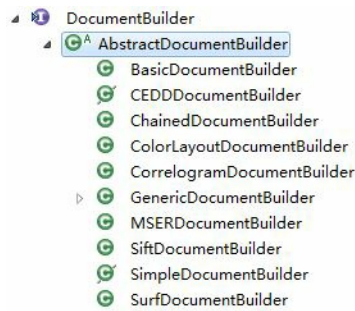
（LIRe在代码注释方面做得很好，每个函数的作用都写得很清楚）

DocumentBuilder：接口，定义了基本的方法。

AbstractDocumentBuilder：纯虚类，实现了DocumentBuilder接口。

DocumentBuilderFactory：用于创建DocumentBuilder。

DocumentBuilder相关的类的继承关系如下图所示。可见，各种算法类都继承了AbstractDocumentBuilder，而AbstractDocumentBuilder实现了DocumentBuilder。



详细的源代码如下所示：

## DocumentBuilder

```
[java]
1.  /*
2.   * This file is part of the LIRE project: http://www.semanticmetadata.net/lire
3.   * LIRE is free software; you can redistribute it and/or modify
4.   * it under the terms of the GNU General Public License as published by
5.   * the Free Software Foundation; either version 2 of the License, or
6.   * (at your option) any later version.
7.   *
8.   * LIRE is distributed in the hope that it will be useful,
9.   * but WITHOUT ANY WARRANTY; without even the implied warranty of
10.  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
11.  * GNU General Public License for more details.
12.  *
13.  * You should have received a copy of the GNU General Public License
14.  * along with LIRE; if not, write to the Free Software
15.  * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
16.  *
17.  * We kindly ask you to refer the following paper in any publication mentioning Lire:
18.  *
19.  * Lux Mathias, Savvas A. Chatzichristofis. Lire: Lucene Image Retrieval 欽
20.  * An Extensible Java CBIR Library. In proceedings of the 16th ACM International
21.  * Conference on Multimedia, pp. 1085-1088, Vancouver, Canada, 2008
22.  *
23.  * http://doi.acm.org/10.1145/1459359.1459577
24.  *
25.  * Copyright statement:
26.  *
27.  * (c) 2002-2011 by Mathias Lux (mathias@juggle.at)
28.  * http://www.semanticmetadata.net/lire
29.  */
30.
31. package net.semanticmetadata.lire;
32.
33. import org.apache.lucene.document.Document;
34.
35. import java.awt.image.BufferedImage;
36. import java.io.IOException;
37. import java.io.InputStream;
38.
39. /**
40.  * <h2>Creating an Index</h2>
41.  * <p>
42.  * Use DocumentBuilderFactory to create a DocumentBuilder, which
43.  * will create Lucene Documents from images. Add this documents to
44.  * an index like this:
45.  * <p>
46.  * <pre>
47.  * System.out.println(">> Indexing " + images.size() + " files.");
48.  * DocumentBuilder builder = DocumentBuilderFactory.getExtensiveDocumentBuilder();
49.  * IndexWriter iw = new IndexWriter(indexPath, new SimpleAnalyzer(LuceneUtils.LUCENE_VERSION), true);
50.  * int count = 0;
51.  * long time = System.currentTimeMillis();
52.  * for (String identifier : images) {
53.  * Document doc = builder.createDocument(new FileInputStream(identifier), identifier);
54.  * iw.addDocument(doc);
55.  * count ++;
56.  * if (count % 25 == 0) System.out.println(count + " files indexed.");
57.  * }
58.  * long timeTaken = (System.currentTimeMillis() - time);
59.  * float sec = ((float) timeTaken) / 1000f;
60.  *
61.  * System.out.println(sec + " seconds taken, " + (timeTaken / count) + " ms per image.");
```

```

62.  * iw.optimize();
63.  * iw.close();
64.  * </pre>
65.  * <p/>
66.  * <p/>
67.  * This file is part of the Caliph and Emir project: http://www.SemanticMetadata.net
68.  * <br>Date: 31.01.2006
69.  * <br>Time: 23:02:00
70.  *
71.  * @author Mathias Lux, mathias@juggle.at
72.  */
73.  public interface DocumentBuilder {
74.      public static final int MAX_IMAGE_SIDE_LENGTH = 800;
75.
76.      public static final String FIELD_NAME_SCALABLECOLOR = "descriptorScalableColor";
77.      public static final String FIELD_NAME_COLORLAYOUT = "descriptorColorLayout";
78.      public static final String FIELD_NAME_EDGEHISTOGRAM = "descriptorEdgeHistogram";
79.      public static final String FIELD_NAME_AUTOCOLORCORRELOGRAM = "featureAutoColorCorrelogram";
80.      public static final String FIELD_NAME_COLORHISTOGRAM = "featureColorHistogram";
81.      public static final String FIELD_NAME_CEDD = "featureCEDD";
82.      public static final String FIELD_NAME_FCTH = "featureFCTH";
83.      public static final String FIELD_NAME_JCD = "featureJCD";
84.      public static final String FIELD_NAME_TAMURA = "featureTAMURA";
85.      public static final String FIELD_NAME_GABOR = "featureGabor";
86.      public static final String FIELD_NAME_SIFT = "featureSift";
87.      public static final String FIELD_NAME_SIFT_LOCAL_FEATURE_HISTOGRAM = "featureSiftHistogram";
88.      public static final String FIELD_NAME_SIFT_LOCAL_FEATURE_HISTOGRAM_VISUAL_WORDS = "featureSiftHistogramVWords";
89.      public static final String FIELD_NAME_IDENTIFIER = "descriptorImageIdentifier";
90.      public static final String FIELD_NAME_CEDD_FAST = "featureCEDDfast";
91.      public static final String FIELD_NAME_COLORLAYOUT_FAST = "featureColorLayoutfast";
92.      public static final String FIELD_NAME_SURF = "featureSurf";
93.      public static final String FIELD_NAME_SURF_LOCAL_FEATURE_HISTOGRAM = "featureSURFHistogram";
94.      public static final String FIELD_NAME_SURF_LOCAL_FEATURE_HISTOGRAM_VISUAL_WORDS = "featureSurfHistogramVWords";
95.      public static final String FIELD_NAME_MSER_LOCAL_FEATURE_HISTOGRAM = "featureMSERHistogram";
96.      public static final String FIELD_NAME_MSER_LOCAL_FEATURE_HISTOGRAM_VISUAL_WORDS = "featureMSERHistogramVWords";
97.      public static final String FIELD_NAME_MSER = "featureMSER";
98.      public static final String FIELD_NAME_BASIC_FEATURES = "featureBasic";
99.      public static final String FIELD_NAME_JPEGCOEFFS = "featureJpegCoeffs";
100.
101.
102.      /**
103.       * Creates a new Lucene document from a BufferedImage. The identifier can be used like an id
104.       * (e.g. the file name or the url of the image)
105.       *
106.       * @param image the image to index. Cannot be NULL.
107.       * @param identifier an id for the image, for instance the filename or an URL. Can be NULL.
108.       * @return a Lucene Document containing the indexed image.
109.       */
110.      public Document createDocument(BufferedImage image, String identifier);
111.
112.      /**
113.       * Creates a new Lucene document from an InputStream. The identifier can be used like an id
114.       * (e.g. the file name or the url of the image)
115.       *
116.       * @param image the image to index. Cannot be NULL.
117.       * @param identifier an id for the image, for instance the filename or an URL. Can be NULL.
118.       * @return a Lucene Document containing the indexed image.
119.       * @throws IOException in case the image cannot be retrieved from the InputStream
120.       */
121.      public Document createDocument(InputStream image, String identifier) throws IOException;
122.
123.  }

```

从接口的源代码可以看出，提供了两个方法，名字都叫createDocument()，只是参数不一样，一个是从BufferedImage，另一个是从InputStream。

此外，定义了很多的字符串。

### AbstractDocumentBuilder

```

1.  /*
2.   * This file is part of the LIRe project: http://www.semanticmetadata.net/lire
3.   * LIRe is free software; you can redistribute it and/or modify
4.   * it under the terms of the GNU General Public License as published by
5.   * the Free Software Foundation; either version 2 of the License, or
6.   * (at your option) any later version.
7.   *
8.   * LIRe is distributed in the hope that it will be useful,
9.   * but WITHOUT ANY WARRANTY; without even the implied warranty of
10.  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
11.  * GNU General Public License for more details.
12.  *
13.  * You should have received a copy of the GNU General Public License
14.  * along with LIRe; if not, write to the Free Software
15.  * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
16.  *
17.  * We kindly ask you to refer the following paper in any publication mentioning Lire:
18.  *
19.  * Lux Mathias, Savvas A. Chatzichristofis. Lire: Lucene Image Retrieval 鈥?
20.  * An Extensible Java CBIR Library. In proceedings of the 16th ACM International
21.  * Conference on Multimedia, pp. 1085-1088, Vancouver, Canada, 2008
22.  *
23.  * http://doi.acm.org/10.1145/1459359.1459577
24.  *
25.  * Copyright statement:
26.  * -----
27.  * (c) 2002-2011 by Mathias Lux (mathias@juggle.at)
28.  * http://www.semanticmetadata.net/lire
29.  */
30.
31. package net.semanticmetadata.lire;
32.
33. import org.apache.lucene.document.Document;
34.
35. import javax.imageio.ImageIO;
36. import java.awt.image.BufferedImage;
37. import java.io.IOException;
38. import java.io.InputStream;
39.
40. /**
41.  * Abstract DocumentBuilder, which uses javax.imageio.ImageIO to create a BufferedImage
42.  * from an InputStream.
43.  * <p/>
44.  * This file is part of the Caliph and Emir project: http://www.SemanticMetadata.net
45.  * <br>Date: 31.01.2006
46.  * <br>Time: 23:07:39
47.  *
48.  * @author Mathias Lux, mathias@juggle.at
49.  */
50. public abstract class AbstractDocumentBuilder implements DocumentBuilder {
51.     /**
52.      * Creates a new Lucene document from an InputStream. The identifier can be used like an id
53.      * (e.g. the file name or the url of the image). This is a simple implementation using
54.      * javax.imageio.ImageIO
55.      *
56.      * @param image the image to index. Please note that
57.      * @param identifier an id for the image, for instance the filename or an URL.
58.      * @return a Lucene Document containing the indexed image.
59.      * @see javax.imageio.ImageIO
60.      */
61.     public Document createDocument(InputStream image, String identifier) throws IOException {
62.         assert (image != null);
63.         BufferedImage bufferedImage = ImageIO.read(image);
64.         return createDocument(bufferedImage, identifier);
65.     }
66. }

```

从抽象类的定义可以看出, 只有一个createDocument(InputStream image, String identifier), 里面调用了createDocument(BufferedImage image, String identifier)。

其实说白了, 就是把接口的那两个函数合成了一个函数。

## DocumentBuilderFactory

```

1.  /*
2.   * This file is part of the LIRe project: http://www.semanticmetadata.net/lire
3.   * LIRe is free software; you can redistribute it and/or modify
4.   * it under the terms of the GNU General Public License as published by
5.   * the Free Software Foundation; either version 2 of the License, or
6.   * (at your option) any later version.
7.   *
8.   * LIRe is distributed in the hope that it will be useful,
9.   * but WITHOUT ANY WARRANTY; without even the implied warranty of
10.  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
11.  * GNU General Public License for more details.

```

```

12.  *
13.  * You should have received a copy of the GNU General Public License
14.  * along with LIRE; if not, write to the Free Software
15.  * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
16.  *
17.  * We kindly ask you to refer the following paper in any publication mentioning Lire:
18.  *
19.  * Lux Mathias, Savvas A. Chatzichristofis. Lire: Lucene Image Retrieval 欽
20.  * An Extensible Java CBIR Library. In proceedings of the 16th ACM International
21.  * Conference on Multimedia, pp. 1085-1088, Vancouver, Canada, 2008
22.  *
23.  * http://doi.acm.org/10.1145/1459359.1459577
24.  *
25.  * Copyright statement:
26.  * ~~~~~
27.  * (c) 2002-2011 by Mathias Lux (mathias@juggle.at)
28.  * http://www.semanticmetadata.net/lire
29.  */
30.
31. package net.semanticmetadata.lire;
32.
33. import net.semanticmetadata.lire.imageanalysis.*;
34. import net.semanticmetadata.lire.impl.ChainedDocumentBuilder;
35. import net.semanticmetadata.lire.impl.CorrelogramDocumentBuilder;
36. import net.semanticmetadata.lire.impl.GenericDocumentBuilder;
37. import net.semanticmetadata.lire.impl.GenericFastDocumentBuilder;
38.
39. /**
40.  * Use DocumentBuilderFactory to create a DocumentBuilder, which
41.  * will create Lucene Documents from images. <br/>
42.  * This file is part of the Caliph and Emir project: http://www.SemanticMetadata.net
43.  * <br>Date: 31.01.2006
44.  * <br>Time: 23:00:32
45.  *
46.  * @author Mathias Lux, mathias@juggle.at
47.  */
48. public class DocumentBuilderFactory {
49.     /**
50.      * Creates a simple version of a DocumentBuilder. In this case the
51.      * {@link net.semanticmetadata.lire.imageanalysis.CEDD} is used as a feature
52.      *
53.      * @return a simple and efficient DocumentBuilder.
54.      * @see net.semanticmetadata.lire.imageanalysis.CEDD
55.      */
56.     public static DocumentBuilder getDefaultDocumentBuilder() {
57.         return new GenericFastDocumentBuilder(CEDD.class, DocumentBuilder.FIELD_NAME_CEDD);
58.     }
59.
60.     /**
61.      * Creates a simple version of a DocumentBuilder using the MPEG/-7 visual features
62.      * all available descriptors are used.
63.      *
64.      * @return a fully featured DocumentBuilder.
65.      * @see net.semanticmetadata.lire.imageanalysis.ColorLayout
66.      * @see net.semanticmetadata.lire.imageanalysis.EdgeHistogram
67.      * @see net.semanticmetadata.lire.imageanalysis.ScalableColor
68.      * @deprecated Use ChainedDocumentBuilder instead
69.      */
70.     public static DocumentBuilder getExtensiveDocumentBuilder() {
71.         ChainedDocumentBuilder cb = new ChainedDocumentBuilder();
72.         cb.addBuilder(DocumentBuilderFactory.getColorLayoutBuilder());
73.         cb.addBuilder(DocumentBuilderFactory.getEdgeHistogramBuilder());
74.         cb.addBuilder(DocumentBuilderFactory.getScalableColorBuilder());
75.         return cb;
76.     }
77.
78.     /**
79.      * Creates a fast (byte[] based) version of the MPEG-7 ColorLayout document builder.
80.      *
81.      * @return the document builder.
82.      */
83.     public static DocumentBuilder getColorLayoutBuilder() {
84.         return new GenericFastDocumentBuilder(ColorLayout.class, DocumentBuilder.FIELD_NAME_COLORLAYOUT);
85.     }
86.
87.     /**
88.      * Creates a fast (byte[] based) version of the MPEG-7 EdgeHistogram document builder.
89.      *
90.      * @return the document builder.
91.      */
92.     public static DocumentBuilder getEdgeHistogramBuilder() {
93.         return new GenericFastDocumentBuilder(EdgeHistogram.class, DocumentBuilder.FIELD_NAME_EDGEHISTOGRAM);
94.     }
95.
96.     /**
97.      * Creates a fast (byte[] based) version of the MPEG-7 ColorLayout document builder.
98.      *
99.      * @return the document builder.
100.     */
101.     public static DocumentBuilder getScalableColorBuilder() {
102.         return new GenericFastDocumentBuilder(ScalableColor.class, DocumentBuilder.FIELD_NAME_SCALABLECOLOR);

```

```

103.     }
104.
105.     /**
106.      * Creates a simple version of a DocumentBuilder using ScalableColor.
107.      *
108.      * @return a fully featured DocumentBuilder.
109.      * @see net.semanticmetadata.lire.imageanalysis.ScalableColor
110.      * @deprecated Use ColorHistogram and the respective factory methods to get it instead
111.      */
112.     public static DocumentBuilder getColorOnlyDocumentBuilder() {
113.         return DocumentBuilderFactory.getScalableColorBuilder();
114.     }
115.
116.     /**
117.      * Creates a simple version of a DocumentBuilder using the ColorLayout feature. Don't use this method any more but
118.      * use the respective feature bound method instead.
119.      *
120.      * @return a simple and fast DocumentBuilder.
121.      * @see net.semanticmetadata.lire.imageanalysis.ColorLayout
122.      * @deprecated use MPEG-7 feature ColorLayout or CEDD, which are both really fast.
123.      */
124.     public static DocumentBuilder getFastDocumentBuilder() {
125.         return DocumentBuilderFactory.getColorLayoutBuilder();
126.     }
127.
128.     /**
129.      * Creates a DocumentBuilder for the AutoColorCorrelogram feature. Note that the extraction of this feature
130.      * is especially slow! So use it only on small images! Images that do not fit in a 200x200 pixel box are
131.      * resized by the document builder to ensure shorter processing time. See
132.      * {@link net.semanticmetadata.lire.imageanalysis.AutoColorCorrelogram} for more information on the image feature.
133.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
134.      *
135.      * @return the created AutoCorrelation feature DocumentBuilder.
136.      */
137.     public static DocumentBuilder getAutoColorCorrelogramDocumentBuilder() {
138.         return new GenericDocumentBuilder(AutoColorCorrelogram.class, DocumentBuilder.FIELD_NAME_AUTOCOLORCORRELOGRAM, GenericDocumentBuilder.Mode.Fast);
139.     }
140.
141.     /**
142.      * Creates a DocumentBuilder for the AutoColorCorrelation feature. Note that the extraction of this feature
143.      * is especially slow, but this is a more fast, but less accurate settings version!
144.      * Images that do not fit in a defined bounding box they are
145.      * resized by the document builder to ensure shorter processing time. See
146.      * {@link net.semanticmetadata.lire.imageanalysis.AutoColorCorrelogram} for more information on the image feature.
147.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
148.      *
149.      * @return the created AutoCorrelation feature DocumentBuilder.
150.      * @deprecated Use #getAutoColorCorrelogramDocumentBuilder instead.
151.      */
152.     public static DocumentBuilder getFastAutoColorCorrelationDocumentBuilder() {
153.         return new CorrelogramDocumentBuilder(AutoColorCorrelogram.Mode.SuperFast);
154.     }
155.
156.     /**
157.      * Creates a DocumentBuilder for the CEDD feature. See
158.      * {@link net.semanticmetadata.lire.imageanalysis.CEDD} for more information on the image feature.
159.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
160.      *
161.      * @return the created CEDD feature DocumentBuilder.
162.      */
163.     public static DocumentBuilder getCEDDDocumentBuilder() {
164.         // return new CEDDDocumentBuilder();
165.         return new GenericFastDocumentBuilder(CEDD.class, DocumentBuilder.FIELD_NAME_CEDD);
166.     }
167.
168.
169.     /**
170.      * Creates a DocumentBuilder for the FCTH feature. See
171.      * {@link net.semanticmetadata.lire.imageanalysis.FCTH} for more information on the image feature.
172.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
173.      *
174.      * @return the created FCTH feature DocumentBuilder.
175.      */
176.     public static DocumentBuilder getFCTHDocumentBuilder() {
177.         return new GenericDocumentBuilder(FCTH.class, DocumentBuilder.FIELD_NAME_FCTH, GenericDocumentBuilder.Mode.Fast);
178.     }
179.
180.     /**
181.      * Creates a DocumentBuilder for the JCD feature. See
182.      * {@link net.semanticmetadata.lire.imageanalysis.JCD} for more information on the image feature.
183.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
184.      *
185.      * @return the created DocumentBuilder
186.      */
187.     public static DocumentBuilder getJCDDocumentBuilder() {
188.         return new GenericFastDocumentBuilder(JCD.class, DocumentBuilder.FIELD_NAME_JCD);
189.     }
190.
191.     /**
192.      * Creates a DocumentBuilder for the JpegCoefficientHistogram feature. See
193.      * {@link net.semanticmetadata.lire.imageanalysis.JpegCoefficientHistogram} for more

```

```

193.      * {@link net.semanticmetadata.lire.imageanalysis.JpegCoefficientHistogram} for more
194.      * information on the image feature.
195.      * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
196.      *
197.      * @return the created DocumentBuilder
198.      */
199.      public static DocumentBuilder getJpegCoefficientHistogramDocumentBuilder() {
200.          return new GenericDocumentBuilder(JpegCoefficientHistogram.class, DocumentBuilder.FIELD_NAME_JPEGCOEFFS, GenericDocumentBuild
r.Mode.Fast);
201.      }
202.
203.      /**
204.       * Creates a DocumentBuilder for simple RGB color histograms. See
205.       * {@link net.semanticmetadata.lire.imageanalysis.SimpleColorHistogram} for more
206.       * information on the image feature.
207.       * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
208.       *
209.       * @return the created feature DocumentBuilder.
210.       */
211.      public static DocumentBuilder getColorHistogramDocumentBuilder() {
212.          return new GenericDocumentBuilder(SimpleColorHistogram.class, DocumentBuilder.FIELD_NAME_COLORHISTOGRAM, GenericDocumentBuild
r.Mode.Fast);
213.      }
214.
215.      /**
216.       * Creates a DocumentBuilder for three Tamura features. See
217.       * {@link net.semanticmetadata.lire.imageanalysis.Tamura} for more
218.       * information on the image feature.
219.       * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
220.       *
221.       * @return the created Tamura feature DocumentBuilder.
222.       */
223.      public static DocumentBuilder getTamuraDocumentBuilder() {
224.          return new GenericFastDocumentBuilder(Tamura.class, DocumentBuilder.FIELD_NAME_TAMURA);
225.      }
226.
227.      /**
228.       * Creates a DocumentBuilder for the Gabor feature. See
229.       * {@link net.semanticmetadata.lire.imageanalysis.Gabor} for more
230.       * information on the image feature.
231.       * Be sure to use the same options for the ImageSearcher as you used for the DocumentBuilder.
232.       *
233.       * @return the created Tamura feature DocumentBuilder.
234.       */
235.      public static DocumentBuilder getGaborDocumentBuilder() {
236.          return new GenericFastDocumentBuilder(Gabor.class, DocumentBuilder.FIELD_NAME_GABOR);
237.      }
238.
239.
240.      /**
241.       * Creates and returns a DocumentBuilder, which contains all available features. For
242.       * AutoColorCorrelogram the getAutoColorCorrelogramDocumentBuilder() is used. Therefore
243.       * it is compatible with the respective Searcher.
244.       *
245.       * @return a combination of all available features.
246.       */
247.      public static DocumentBuilder getFullDocumentBuilder() {
248.          ChainedDocumentBuilder cdb = new ChainedDocumentBuilder();
249.          cdb.addBuilder(DocumentBuilderFactory.getExtensiveDocumentBuilder());
250.          cdb.addBuilder(DocumentBuilderFactory.getAutoColorCorrelogramDocumentBuilder());
251.          cdb.addBuilder(DocumentBuilderFactory.getCEDDDocumentBuilder());
252.          cdb.addBuilder(DocumentBuilderFactory.getFCTHDocumentBuilder());
253.          cdb.addBuilder(DocumentBuilderFactory.getColorHistogramDocumentBuilder());
254.          cdb.addBuilder(DocumentBuilderFactory.getTamuraDocumentBuilder());
255.          cdb.addBuilder(DocumentBuilderFactory.getGaborDocumentBuilder());
256.          return cdb;
257.      }
258.  }

```

DocumentBuilderFactory是用于创建DocumentBuilder的。里面有各种get\*\*\*\*DocumentBuilder()。其中以下2种是几个DocumentBuilder的合集：

getExtensiveDocumentBuilder()：使用MPEG-7中的ColorLayout, EdgeHistogram, ScalableColor

getFullDocumentBuilder()：使用所有的DocumentBuilder

文章标签：[lire](#) [源代码](#) [索引](#) [检索](#) [lucene](#)

个人分类：[LIRe](#) [MPEG7/图像检索](#)

所属专栏：[开源多媒体项目源代码分析](#)

---

此PDF由[spygg](#)生成,请尊重原作者版权!!!

我的邮箱:[liushidc@163.com](mailto:liushidc@163.com)