Content Detection, Metadata and Content Extraction with Apache Tika

December 2nd, 2012 by micha kops

Encountering the situation that you want to extract meta-data or content from a file – might it be an office document, a spreadsheet or even a mp3 or an image – or you'd like to detect the content type for a given file then Apache Tika might be a helpful tool for you.

Apache Tika supports a variety of document formats and has a nice, extendable parser and detection API with a lot of built-in parsers available.

Dependencies

To build and run the following examples you'll be needing just two dependencies – *tika-core* and *tika-parsers*

Maven

If Maven is your build tool of choice here, you should add the following dependencies to your *pom.xml*

```
<properties>
 <tika.version>1.2</tika.version>
</properties>

<dependencies>
 <dependency>
 <groupId>org.apache.tika</groupId>
 <artifactId>tika-core</artifactId>
 <version>${tika.version}</version>
 </dependency>
 <dependency>
 <groupId>org.apache.tika</groupId>
 <artifactId>tika-parsers</artifactId>
 <version>${tika.version}</version>
 </dependency>
</dependencies>
```

SBT

If you prefer SBT, just add the following two lines to your *build.sbt*

```
libraryDependencies += "org.apache.tika" % "tika-core" % "1.2"
libraryDependencies += "org.apache.tika" % "tika-parsers" % "1.2"
```

Extracting Metadata from a PDF using a concrete Parser

In the first example, we're using a concrete parser implementation to extract metadata information from a PDF file. I've used the Guide: Writing Testable Code from Jonathan Wolter, Russ Ruffer, Miško Hevery here, Blaine R Southam kindly created a PDF file from the online version that you may download here from Mr. Hevery's blog.

```
package com.hascode.tutorial;

import java.io.IOException;
import java.io.InputStream;

import org.apache.tika.exception.TikaException;
import org.apache.tika.metadata.Metadata;
import org.apache.tika.parser.ParseContext;
import org.apache.tika.parser.Parser;
import org.apache.tika.parser.pdf.PDFParser;
import org.apache.tika.sax.BodyContentHandler;
import org.xml.sax.SAXException;

public class ConcretePDFExtractor {
      public static void main(final String[] args) throws IOException,
                  SAXException, TikaException {
            Parser parser = new PDFParser();
            BodyContentHandler handler = new BodyContentHandler(10000000);
            Metadata metadata = new Metadata();
            InputStream content = ConcretePDFExtractor.class
                        .getResourceAsStream("/demo.pdf");
            parser.parse(content, handler, metadata, new ParseContext());
            for (String name : metadata.names()) {
                  System.out.println(name + ":\t" + metadata.get(name));
            }
      }
}
```

Running the example should produce the following output:

```
dcterms:modified:    2009-02-23T18:04:40Z
meta:creation-date:    2009-02-23T18:04:40Z
meta:save-date:    2009-02-23T18:04:40Z
dc:creator:    Blaine R Southam
Last-Modified:    2009-02-23T18:04:40Z
Author:    Blaine R Southam
dcterms:created:    2009-02-23T18:04:40Z
date:    2009-02-23T18:04:40Z
modified:    2009-02-23T18:04:40Z
creator:    Blaine R Southam
xmpTPg:NPages:    38
Creation-Date:    2009-02-23T18:04:40Z
title:    Guide-Writing Testable Code
meta:author:    Blaine R Southam
created:    Mon Feb 23 19:04:40 CET 2009
producer:    Microsoft® Office Word 2007
Content-Type:    application/pdf
xmp:CreatorTool:    Microsoft® Office Word 2007
Last-Save-Date:    2009-02-23T18:04:40Z
dc:title:    Guide-Writing Testable Code
```

Extracting Metadata from different Document Formats

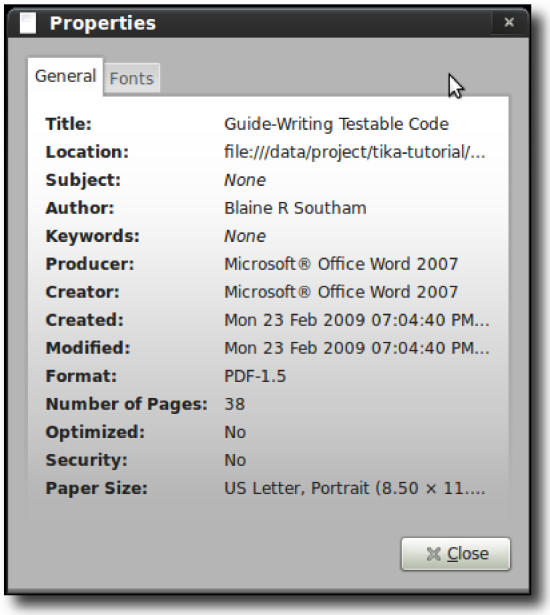
In the following example we're trying to extract information from a variety of different formats .. from PDF to MP3.

We're using the *AutodetectParser* here to handle the different formats.

```
package com.hascode.tutorial;

import java.io.IOException;
import java.io.InputStream;

import org.apache.tika.exception.TikaException;
import org.apache.tika.metadata.Metadata;
import org.apache.tika.parser.AutoDetectParser;
import org.apache.tika.parser.ParseContext;
import org.apache.tika.parser.Parser;
import org.apache.tika.sax.BodyContentHandler;
import org.xml.sax.SAXException;

public class AutoDetectionExample {
	public static void main(final String[] args) throws IOException,
			SAXException, TikaException {
		Parser parser = new AutoDetectParser();

		System.out.println("------------ Parsing a PDF:");
		extractFromFile(parser, "/demo.pdf");

		System.out.println("------------ Parsing an Office Document:");
		extractFromFile(parser, "/demo.docx");

		System.out.println("------------ Parsing a Spreadsheet:");
		extractFromFile(parser, "/demo.xlsx");

		System.out.println("------------ Parsing a Presentation:");
		extractFromFile(parser, "/demo.odp");

		System.out.println("------------ Parsing a PNG:");
		extractFromFile(parser, "/demo.png");

		System.out.println("------------ Parsing a Video/AVI:");
		extractFromFile(parser, "/demo.avi");

		System.out.println("------------ Parsing a MP3:");
		extractFromFile(parser, "/demo.mp3");

		System.out.println("------------ Parsing a Java Class:");
		extractFromFile(parser,
				"/com/hascode/tutorial/ConcretePDFExtractor.class");

		System.out.println("------------ Parsing a HTML File:");
		extractFromFile(parser, "/demo.html");
	}

	private static void extractFromFile(final Parser parser,
			final String fileName) throws IOException, SAXException,
			TikaException {
		long start = System.currentTimeMillis();
		BodyContentHandler handler = new BodyContentHandler(10000000);
		Metadata metadata = new Metadata();
		InputStream content = AutoDetectionExample.class
				.getResourceAsStream(fileName);
		parser.parse(content, handler, metadata, new ParseContext());
		for (String name : metadata.names()) {
			System.out.println(name + ":\t" + metadata.get(name));
		}
		System.out.println(String.format(
				"------------ Processing took %s millis\n\n",
				System.currentTimeMillis() - start));
	}
}
```

When running the code above the following output should be produced for each type:

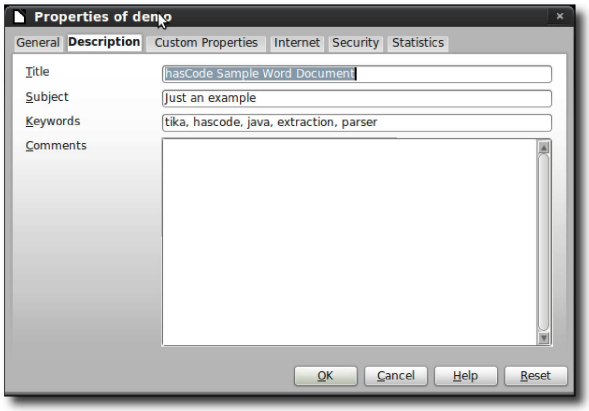
Portable Document Format / PDF



PDF Document Properties

```
------------ Parsing a PDF:
dcterms:modified: 2009-02-23T18:04:40Z
meta:creation-date:     2009-02-23T18:04:40Z
meta:save-date:   2009-02-23T18:04:40Z
dc:creator: Blaine R Southam
Last-Modified:    2009-02-23T18:04:40Z
Author:     Blaine R Southam
dcterms:created:  2009-02-23T18:04:40Z
date: 2009-02-23T18:04:40Z
modified:   2009-02-23T18:04:40Z
creator:    Blaine R Southam
xmpTPg:NPages:    38
Creation-Date:    2009-02-23T18:04:40Z
title:      Guide-Writing Testable Code
meta:author:      Blaine R Southam
created:    Mon Feb 23 19:04:40 CET 2009
producer:   Microsoft® Office Word 2007
Content-Type:     application/pdf
xmp:CreatorTool:  Microsoft® Office Word 2007
Last-Save-Date:   2009-02-23T18:04:40Z
dc:title:   Guide-Writing Testable Code
------------ Processing took 1372 millis
```

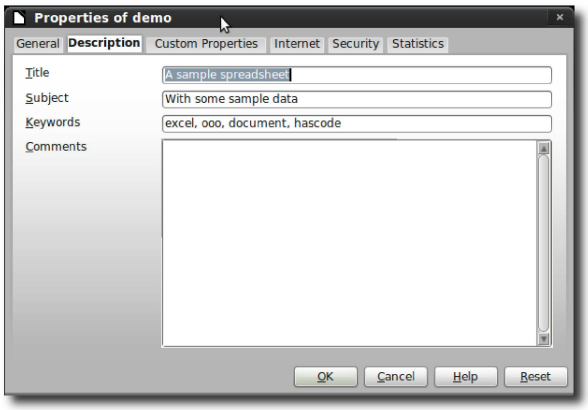
Office Document / DOCX



Office Document Properties

```
------------ Parsing an Office Document:
Revision-Number:  0
cp:revision:      0
title:      hasCode Sample Word Document
dc:subject: tika hascode java extraction parser
subject:    Just an example
cp:subject: Just an example
meta:keyword:     tika hascode java extraction parser
Content-Type:     application/vnd.openxmlformats-officedocument.wordprocessingml.document
Keywords:   tika hascode java extraction parser
dc:title:   hasCode Sample Word Document
------------ Processing took 420 millis
```

Spreadsheet / XLSX

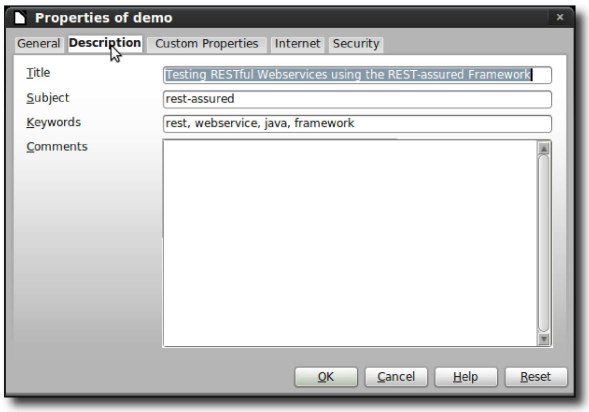


Spreadsheet Properties

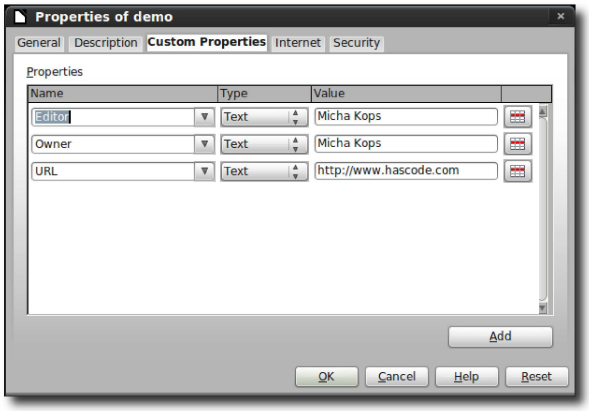
```
------------ Parsing a Spreadsheet:
cp:revision:      0
Revision-Number:  0
dc:subject: excel ooo document hascode
subject:    With some sample data
Application-Name: LibreOffice/3.5$Linux_X86_64 LibreOffice_project/350m1$Build-2
title:      A sample spreadsheet
protected:  false
meta:keyword:     excel ooo document hascode
cp:subject: With some sample data
extended-properties:Application:    LibreOffice/3.5$Linux_X86_64 LibreOffice_project/350m1$Build-2
Content-Type:     application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
Keywords:   excel ooo document hascode
dc:title:   A sample spreadsheet
------------ Processing took 115 millis
```

LibreOffice/OOO Presentation / ODP

I've used my presetation about *Testing RESTful Webservices using the REST-assured Framework* here – if you're interested in the presentation – it has been published to Slideshare.



Presentation Document Properties

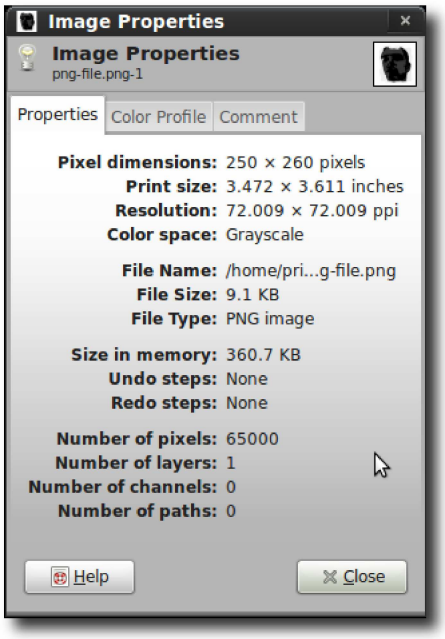


Presentation Document additional Properties

```
------------ Parsing a Presentation:
editing-cycles:   56
meta:save-date:   2012-01-24T21:04:57
dc:subject: rest, webservice, java, framework
subject:    rest-assured
dcterms:created:  2012-01-24T19:49:24
Author:     Micha Kops
date: 2012-01-24T19:49:24
dc:description:   rest-assured
creator:    Micha Kops
nbObject:   145
Edit-Time:  PT01H04M54S
Creation-Date:    2012-01-24T19:49:24
title:      Testing RESTful Webservices using the REST-assured Framework
Object-Count:     145
meta:author:      Micha Kops
description:      rest-assured
meta:object-count:      145
cp:subject: rest-assured
generator:  OpenOffice.org/3.2$Linux OpenOffice.org_project/320m12$Build-9483
custom:Editor:    Micha Kops
Keywords:   rest, webservice, java, framework
custom:URL: http://www.hascode.com
dc:title:   Testing RESTful Webservices using the REST-assured Framework
Last-Save-Date:   2012-01-24T21:04:57
dcterms:modified: 2012-01-24T21:04:57
meta:creation-date:     2012-01-24T19:49:24
dc:creator: Micha Kops
Last-Modified:    2012-01-24T21:04:57
modified:   2012-01-24T21:04:57
custom:Owner:     Micha Kops
initial-creator:  Micha Kops
meta:initial-author:    Micha Kops
language:   fi-FI
Content-Type:     application/vnd.oasis.opendocument.presentation
dc:language:      fi-FI
------------ Processing took 122 millis
```

Portable Network Graphics / PNG

I have used my ugly avatar image here 😃



PNG File Image Properties

```
------------ Parsing a PNG:
sRGB: Perceptual
Compression Lossless:   true
tIME: year=2010, month=8, day=15, hour=15, minute=28, second=58
Dimension PixelAspectRatio:   1.0
tiff:ImageLength: 260
height:     260
pHYs: pixelsPerUnitXAxis=2835, pixelsPerUnitYAxis=2835, unitSpecifier=meter
tiff:ImageWidth:  250
Chroma BlackIsZero:     true
Document ImageModificationTime:     year=2010, month=8, day=15, hour=15, minute=28, second=58
bKGD bKGD_Grayscale:    255
Chroma BackgroundColor: red=255, green=255, blue=255
Data BitsPerSample:     8
Dimension VerticalPixelSize:  0.35273367
tiff:BitsPerSample:     8
width:      250
Dimension ImageOrientation:   Normal
Compression CompressionTypeName:    deflate
Data SampleFormat:      UnsignedIntegral
Dimension HorizontalPixelSize:      0.35273367
Transparency Alpha:     none
Chroma NumChannels:     1
Compression NumProgressiveScans:    1
Chroma ColorSpaceType:  GRAY
IHDR: width=250, height=260, bitDepth=8, colorType=Grayscale, compressionMethod=deflate, filterMethod=adaptive, interlaceMethod=none
Data PlanarConfiguration:     PixelInterleaved
Content-Type:     image/png
------------ Processing took 18 millis
```

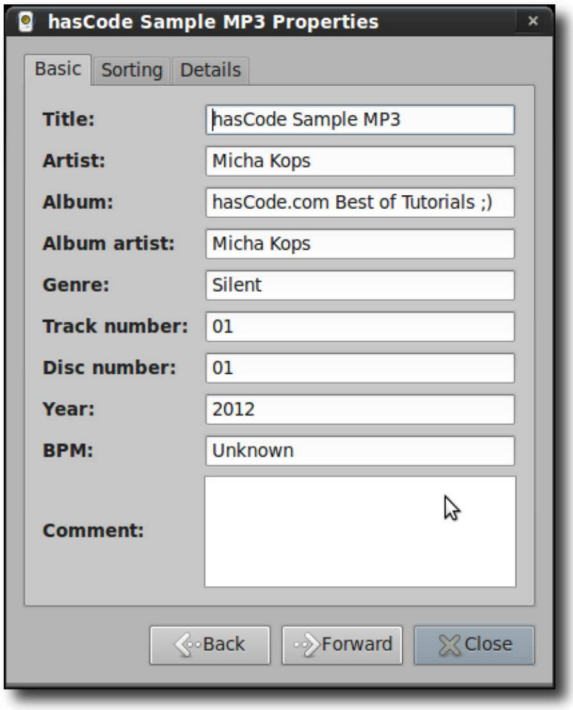
Video / AVI

I've simply used one of the videos from an Android tutorial of mine here ..

```
------------ Parsing a Video/AVI:
Content-Type:     video/x-msvideo
------------ Processing took 1 millis
```

MP3

I have used arecord and lame to create an empty mp3 file and added some metadata.



MP3 Properties

```
------------ Parsing a MP3:
xmpDM:releaseDate:      2012
xmpDM:audioChannelType: Stereo
dc:creator: Micha Kops
xmpDM:album:      hasCode.com Best of Tutorials ;)
Author:     Micha Kops
xmpDM:artist:     Micha Kops
channels:   2
xmpDM:audioSampleRate:  44100
xmpDM:logComment: XXX - Comment
An example mp3 for my tutorial on content extraction with Apache Tika.
xmpDM:trackNumber:      1
version:    MPEG 3 Layer III Version 1
creator:    Micha Kops
xmpDM:composer:   null
xmpDM:audioCompressor:  MP3
title:      hasCode Sample MP3
samplerate: 44100
meta:author:      Micha Kops
xmpDM:genre:      Silent
Content-Type:     audio/mpeg
dc:title:   hasCode Sample MP3
------------ Processing took 10 millis
```

Java Class

I've simply used the class file from the first example in this tutorial here …

```
------------ Parsing a Java Class:
title:      ConcretePDFExtractor
Content-Type:     application/java-vm
resourceName:     ConcretePDFExtractor.class
dc:title:   ConcretePDFExtractor
------------ Processing took 9 millis
```

Hypertext Markup Language / HTML

I've used a simple HTML file here and added a set of metatags and Dublin-Core-Metatdata here:

```
<html>
<head>
<title>hasCode.com Sample Page</title>
<meta http-equiv="content-type" content="text/html;charset=UTF-8" />
<meta name="author" content="Micha Kops" />
<meta name="description"
      content="A sample HTML file for my Tika Tutorial" />
<meta name="keywords"
      content="tika, java, programming, content extraction, tutorials">
<meta name="date" content="2012-11-30T06:30:00-01:00">
<meta name="DC.title" content="hasCode.com Sample Page" />
<meta name="DC.creator" content="Micha Kops" />
<meta name="DC.subject" content="Tika Tutorial" />
<meta name="DC.description"
      content="A sample HTML file for my Tika Tutorial" />
<meta name="DC.publisher" content="hasCode.com" />
<meta name="DC.contributor" content="Micha Kops" />
<meta name="DC.date" content="2012-11-30T06:30:00-01:00"
      scheme="DCTERMS.W3CDTF" />
<meta name="DC.type" content="Text" scheme="DCTERMS.DCMIType" />
<meta name="DC.format" content="text/html" scheme="DCTERMS.IMT" />
<meta name="DC.language" content="en" scheme="DCTERMS.RFC3066" />
<meta name="DC.relation" content="http://dublincore.org/"
      scheme="DCTERMS.URI" />
</head>
<body>
      <h1>hasCode.com</h1>
      <div>Now with an improved layout.</div>
</body>
</html>
```

This is the result:

```
------------ Parsing a HTML File:
DC.description:   A sample HTML file for my Tika Tutorial
keywords:   tika, java, programming, content extraction, tutorials
DC.publisher:     hasCode.com
DC.relation:      http://dublincore.org/
DC.language:      en
date: 2012-11-30T06:30:00-01:00
DC.type:    Text
author:     Micha Kops
title:      hasCode.com Sample Page
DC.date:    2012-11-30T06:30:00-01:00
description:      A sample HTML file for my Tika Tutorial
Content-Encoding: UTF-8
DC.title:   hasCode.com Sample Page
DC.format:  text/html
DC.contributor:   Micha Kops
Content-Type:     text/html; charset=UTF-8
DC.creator: Micha Kops
dc:title:   hasCode.com Sample Page
DC.subject: Tika Tutorial
------------ Processing took 64 millis
```

Content Extraction

Fetching metadata is one thing – another is to extract the actual content from a file. The following example extracts the content from different formats – if you want to take a look at the original files, please visit the files in my Bitbucket repository.

```
package com.hascode.tutorial;

import java.io.IOException;
import java.io.InputStream;

import org.apache.tika.exception.TikaException;
import org.apache.tika.metadata.Metadata;
import org.apache.tika.parser.AutoDetectParser;
import org.apache.tika.parser.ParseContext;
import org.apache.tika.parser.Parser;
import org.apache.tika.sax.BodyContentHandler;
import org.xml.sax.SAXException;

public class ContentExtraction {
    public static void main(final String[] args) throws IOException,
            SAXException, TikaException {
        Parser parser = new AutoDetectParser();

        System.out
                .println("------------ Extracting the content from an Office Document:");
        extractContentFromFile(parser, "/demo.docx");

        System.out
                .println("------------ Extracting the content from a Spreadsheet:");
        extractContentFromFile(parser, "/demo.xlsx");

        System.out
                .println("------------ Extracting the content from a Presentation:");
        extractContentFromFile(parser, "/demo.odp");

        System.out.println("------------ Extracting the content from a MP3:");
        extractContentFromFile(parser, "/demo.mp3");

        System.out
                .println("------------ Extracting the content from a Java Class:");
        extractContentFromFile(parser,
                "/com/hascode/tutorial/ConcretePDFExtractor.class");

        System.out
                .println("------------ Extracting the content from a HTML File:");
        extractContentFromFile(parser, "/demo.html");
    }

    private static void extractContentFromFile(final Parser parser,
            final String fileName) throws IOException, SAXException,
            TikaException {
        BodyContentHandler handler = new BodyContentHandler(10000000);
        Metadata metadata = new Metadata();
        InputStream content = AutoDetectionExample.class
                .getResourceAsStream(fileName);
        parser.parse(content, handler, metadata, new ParseContext());
        System.out.println(handler.toString());
    }
}
```

Running the code above gives the following output for each file:

Office Document

```
------------ Extracting the content from an Office Document:
This is a test document - nice isn't it?
```

Spreadsheet

```
------------ Extracting the content from a Spreadsheet:
Sheet1
        Data Summary
        2012
        Department      January February        March   April   May     June    July    August  September       Oktober November
        Sales   453     32      23      324     232     234     234     23      223     234     234     222
        IT      43      3       32      234     332     531     124     432     513     232     423     143
        Consulting      322     234     43      23      234     214     512     234     1432    12431   1411    1412
        2011
        Department      January February        March   April   May     June    July    August  September       Oktober November
        Sales   453     32      23      324     232     234     234     23      223     234     234     222
        IT      43      3       32      234     332     531     124     432     513     232     423     143
        Consulting      322     234     43      23      234     214     512     234     1432    12431   1411    1412
        2010
        Department      January February        March   April   May     June    July    August  September       Oktober November
        Sales   453     32      23      324     232     234     234     23      223     234     234     222
        IT      43      3       32      234     332     531     124     432     513     232     423     143
        Consulting      322     234     43      23      234     214     512     234     1432    12431   1411    1412

&"Times New Roman,Regular"&12&A

&"Times New Roman,Regular"&12Page &P

Sheet2

&"Times New Roman,Regular"&12&A

&"Times New Roman,Regular"&12Page &P

Sheet3

&"Times New Roman,Regular"&12&A

&"Times New Roman,Regular"&12Page &P
```

Presentation

```
------------ Extracting the content from a Presentation:
Testing RESTful Webservices using the REST-assured Framework

Table of Contents

    Prerequisites

    REST-assured and Maven

    Verify JSON via GET

    JsonPath

    Groovy Closures - The JSON

    Groovy Closures - The Test

    Verifying XML, Xpath, Schema

    Request Parameters

    Status Codes, Headers

    [..]
```

MP3

```
------------ Extracting the content from a MP3:
hasCode Sample MP3
Micha Kops
hasCode.com Best of Tutorials ;), track 1
2012
Silent
XXX - Comment
An example mp3 for my tutorial on content extraction with Apache Tika.
```

Java Class

```
------------ Extracting the content from a Java Class:
package com.hascode.tutorial;
public synchronized class ConcretePDFExtractor {
    public void ConcretePDFExtractor();
    public static void main(String[]) throws java.io.IOException, org.xml.sax.SAXException, org.apache.tika.exception.TikaException;
}
```

HTML

```
------------ Extracting the content from a HTML File:

    hasCode.com

    Now with an improved layout.
```

Content Type Detection

Apache Tika offers different implementations for content type detection like Mime Magic Detection, Resource Name Based Detection, Known Content Type Detection, The

default Mime Types Detector, Container Aware Detection and Language Detection.

More detailed information about the possibilities of content detection is to be found in the Apache Tika Documentation.

```
package com.hascode.tutorial;

import java.io.IOException;

import org.apache.tika.detect.DefaultDetector;
import org.apache.tika.detect.Detector;
import org.apache.tika.metadata.Metadata;
import org.apache.tika.mime.MediaType;

public class ContentDetectionExample {
	public static void main(final String[] args) throws IOException {
		detectContentFromFile("/demo.pdf");
		detectContentFromFile("/demo.docx");
		detectContentFromFile("/demo.xlsx");
		detectContentFromFile("/demo.odp");
		detectContentFromFile("/demo.png");
		detectContentFromFile("/demo.avi");
		detectContentFromFile("/demo.mp3");
		detectContentFromFile("/com/hascode/tutorial/ConcretePDFExtractor.class");
		detectContentFromFile("/demo.html");
	}

	private static void detectContentFromFile(final String fileName)
			throws IOException {
		Detector detector = new DefaultDetector();
		MediaType type = detector.detect(
				ContentDetectionExample.class.getResourceAsStream(fileName),
				new Metadata());
		System.out.println(String.format(
				"detected media type for given file %s: %s", fileName,
				type.toString()));
	}

}
```

Running this prints the following output:

```
detected media type for given file /demo.pdf: application/pdf
detected media type for given file /demo.docx: application/zip
detected media type for given file /demo.xlsx: application/zip
detected media type for given file /demo.odp: application/zip
detected media type for given file /demo.png: image/png
detected media type for given file /demo.avi: video/x-msvideo
detected media type for given file /demo.mp3: audio/mpeg
detected media type for given file /com/hascode/tutorial/ConcretePDFExtractor.class: application/java-vm
detected media type for given file /demo.html: text/html
```

Tutorial Sources

Please feel free to download the tutorial sources from my Bitbucket repository, fork it there or clone it using Mercurial:

```
hg clone https://bitbucket.org/hascode/tika-examples
```

Resources

- Apache Tika Project Website
- List of supported document formats
- Apache Tika API Javadocs
- Misko Hevery: Guide: Writing Testable Code Website and the PDF Version from Blaine R. Southam
- Dublin Core Metadata Initiative: Metadata Elements

Tags: Apache, content extraction, formats, Java, lucene, maven, parser, search, tika

This entry was posted on Sunday, December 2nd, 2012 at 4:44 pm and is filed under Java. You can follow any responses to this entry through the RSS 2.0 feed. You can leave a response, or trackback from your own site.