RECUPERACIÓN DE LA INFORMACIÓN

MEMORIA DE LA PRÁCTICA II

Autor:
Jesús Gabriel Gómez Tocino

NOVIEMBRE 2023

Índice

Página

1. Introducción

Este documento recoge el proceso de desarrollo y evaluación de los distintos ejercicios propuestos en la práctica 2, correspondientes a la herramienta ***Tika***.

2. Contenido de la carpeta entregada

Para cada ejercicio, existe una carpeta con los archivos usados y/o generados. Adicionalmente, se incluyen algunos ficheros de texto plano con los metadatos extraídos para facilitar su visualizado.

1. **Carpeta 1**
   - `Tika.pdf`: Documento PDF usado para extraer los metadatos.
   - `1.png`: Metadatos extraídos del documento anterior.
2. **Carpeta 2**
   - `Tika.pdf`: Documento PDF origen.
   - `Tika.doc`: Documento DOC destino.
3. **Carpeta 3**
   - `3.txt`: Metadatos extraídos del archivo web.
4. **Carpeta 4**
   - `comprimido.zip`: Archivo comprimido usado para extraer los metadatos.
   - `4.png`: Metadatos extraídos del archivo anterior.
5. **Carpeta 5**
   - `original.jpg`: Imagen original.
   - `correo.jpg`: Imagen enviada por correo.
   - `whatsapp.jpg`: Imagen enviada por WhatsApp.
6. **Carpeta 6**
   - `metadatosUCA.txt`: Archivo de texto plano que contiene los metadatos extraídos de la página principal de la UCA.
7. **Carpeta 7**
   - `celebs_fixed.rdf`: Documento (original) en formato RDF.
   - `celebs_fixed.doc`: Documento (transformado) en formato DOC.
8. **Carpeta 8**
   - `q.jpg`: Imagen 1 a comparar.
   - `r.jpg`: Imagen 2 a comparar.
   - `s.jpg`: Imagen 3 a comparar.
9. **Carpeta 10**
   - `imagen1.jpg`: Imagen descargada 1 a comparar.
   - `imagen2.jpg`: Imagen descargada 2 a comparar.
   - `imagen3.jpg`: Imagen descargada 3 a comparar.
10. **Carpeta 11**
    - `tikaIntelliJ.java`: Código desarrollado para extraer el contenido de un PDF desde Java (usando Tika).
    - `Tika.pdf`: Documento a extraer el contenido.
11. `Memoria.pdf`: Es el documento que usted está leyendo ahora mismo.

3. Ejercicios

3.1. Abrir un archivo PDF en modo ventana y decir quién es el autor.

1. Ejecutamos la aplicación en modo ventana con el comando: `java -jar tika-app-2.9.1.jar`.
2. Arrastramos el PDF a la ventana de la aplicación.
   - En este caso, hemos usado el propio archivo PDF del guión de la práctica para visualizar sus metadatos.
3. Establecemos el filtro de *metadata* y buscamos la etiqueta *creator*.
   - Para el PDF seleccionado, observamos varías líneas donde figura dicha etiqueta. Se adjunta a continuación la salida junto a las líneas relevantes resaltadas.

```
Content-Length: 246403
Content-Type: application/pdf
X-TIKA:Parsed-By: org.apache.tika.parser.DefaultParser
X-TIKA:Parsed-By: org.apache.tika.parser.pdf.PDFParser
X-TIKA:Parsed-By-Full-Set: org.apache.tika.parser.DefaultParser
X-TIKA:Parsed-By-Full-Set: org.apache.tika.parser.pdf.PDFParser
X-TIKA:digest:MD5: ddecbf64f8dff2b3276685379ea4c3f3
X-TIKA:digest:SHA256: 0b7fc83b9d3c0c327da394c0992abe0e79a97c3f2bbbb9340bc1aff833199eb6
access_permission:assemble_document: true
access_permission:can_modify: true
access_permission:can_print: true
access_permission:can_print_degraded: true
access_permission:extract_content: true
access_permission:extract_for_accessibility: true
access_permission:fill_in_form: true
access_permission:modify_annotations: true
dc:creator: Lo
dc:format: application/pdf; version=1.5
dc:language: es-ES
dc:title: Ejercicios XML y DTD.docx
dcterms:created: 2017-03-16T10:29:06Z
dcterms:modified: 2017-03-16T10:29:06Z
pdf:PDFVersion: 1.5
pdf:annotationSubtypes: Link
pdf:annotationTypes: null
pdf:charsPerPage: 1588
pdf:charsPerPage: 952
pdf:charsPerPage: 835
pdf:charsPerPage: 1928
pdf:charsPerPage: 1200
pdf:containsDamagedFont: false
pdf:containsNonEmbeddedFont: true
pdf:docinfo:created: 2017-03-16T10:29:06Z
pdf:docinfo:creator: Lo
pdf:docinfo:creator_tool: Microsoft® Office Word 2007
pdf:docinfo:modified: 2017-03-16T10:29:06Z
pdf:docinfo:producer: Microsoft® Office Word 2007
pdf:docinfo:title: Ejercicios XML y DTD.docx
pdf:encrypted: false
pdf:hasCollection: false
pdf:hasMarkedContent: true
pdf:hasXFA: false
pdf:hasXMP: false
pdf:num3DAnnotations: 0
pdf:overallPercentageUnmappedUnicodeChars: 0.0
pdf:producer: Microsoft® Office Word 2007
pdf:totalUnmappedUnicodeChars: 0
pdf:unmappedUnicodeCharsPerPage: 0
pdf:unmappedUnicodeCharsPerPage: 0
pdf:unmappedUnicodeCharsPerPage: 0
pdf:unmappedUnicodeCharsPerPage: 0
pdf:unmappedUnicodeCharsPerPage: 0
resourceName: Tika.pdf
xmp:CreatorTool: Microsoft® Office Word 2007
xmpTPg:NPages: 5
```

Podemos observar que la creadora de los documentos es Lo (Lorena Gutiérrez Madroñal),

3.2. Almacenar el contenido de un archivo PDF en un archivo DOC.

Para este ejercicio, hemos vuelto a usar el archivo PDF del guion de prácticas, llamado `Tika.pdf`. Para convertir dicho archivo a formato DOC, ejecutamos el siguiente comando:

- `java -jar tika-app-2.9.1.jar Tika.pdf > Tika.doc`

Consulte la carpeta entregada para visualizar el archivo generado.

3.3. Ver los metadatos de un archivo que esté subido en una página web.

Para este ejercicio se ha usado un artículo científico, alojado en RODIN.
El comando a utilizar es el siguiente:

- *java -jar tika-app-2.9.1.jar –metadata https://rodin.uca.es/bitstream/handle/10498/28927/APC_2023_075.pdf*

Se adjunta a continuación la salida obtenida:

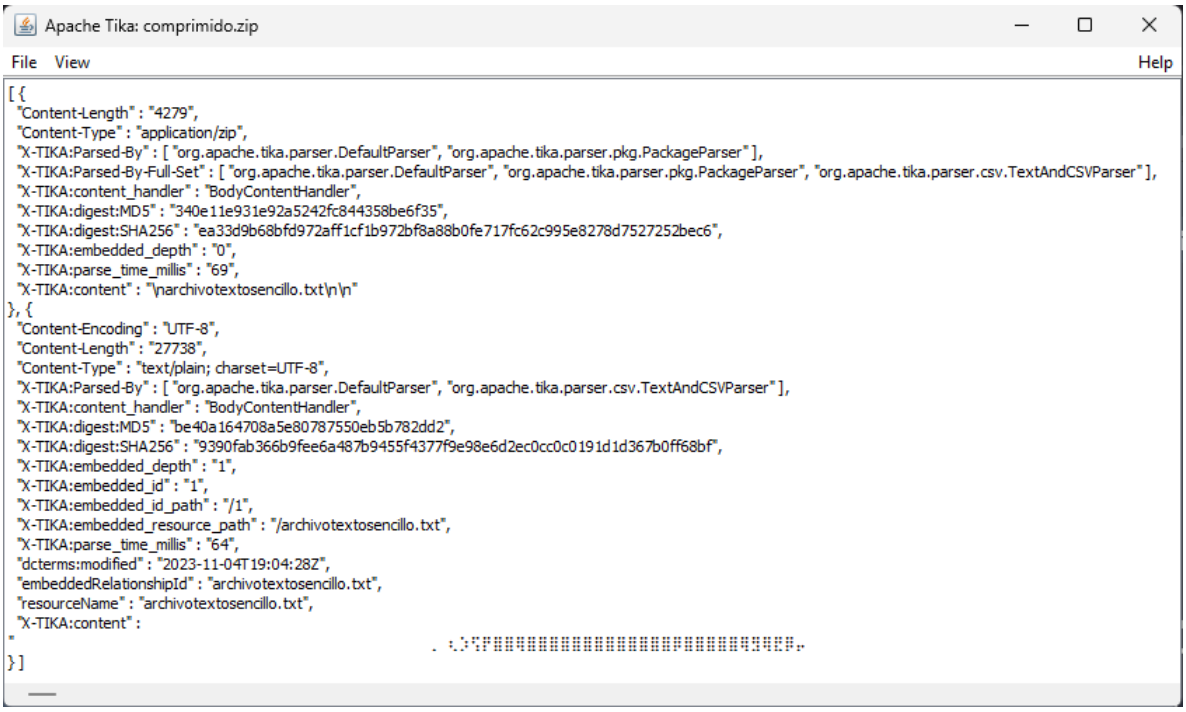
Content-Length:	2278764
Content-Type:	application/pdf
CreationDate–Text:	7 de Junio de 2023
CrossMarkDomains[1]:	elsevier.com
CrossMarkDomains[2]:	sciencedirect.com
CrossmarkDomainExclusive:	true
CrossmarkMajorVersionDate:	23 de Abril de 2010
ElsevierWebPDFSpecifications:	7.0
dc:creator:	Jesús Rosa-Bilbao, Juan Boubeta-Puig, Adrian Rutle
dc:description:	Internet of Things, 22 (2023) 100802. doi:10.1016/j.iot.2023.100802
dc:format:	application/pdf; version=1.7
dc:language:	en-US
dc:subject:	Air quality, Blockchain, Complex event processing, Event-driven architecture, Internet of Things, Low-code
dc:title:	CEPEDALoCo: An event-driven architecture for integrating complex event processing and blockchain through low-code
dcterms:created:	2023-06-07T06:19:20Z
dcterms:modified:	2023-06-07T06:30:56Z
doi:	10.1016/j.iot.2023.100802
meta:keyword:	Air quality, Blockchain, Complex event processing, Event-driven architecture, Internet of Things, Low-code
pdf:PDFVersion:	1.7
pdf:annotationSubtypes:	Link
pdf:charsPerPage:	3228, 5306, 4873, 3978, 3419, 2988, 4775, 3958, 3426, 365, 2450, 2441, 2298, 4715, 7792, 3502
pdf:docinfo:created:	2023-06-07T06:19:20Z
pdf:docinfo:creator_tool:	Elsevier
pdf:docinfo:custom:	CreationDate–Text: 7 de Junio de 2023 CrossMarkDomains[1]: elsevier.com CrossMarkDomains[2]: sciencedirect.com CrossmarkDomainExclusive: true CrossmarkMajorVersionDate: 23 de Abril de 2010 ElsevierWebPDFSpecifications: 7.0 doi: 10.1016/j.iot.2023.100802
pdf:docinfo:producer:	Acrobat Distiller 8.1.0 (Windows)
pdf:encrypted:	false
resourceName:	APC_2023_075.pdf
robots:	noindex
xmp:CreateDate:	2023-06-07T04:19:20Z
xmp:CreatorTool:	Elsevier
xmp:MetadataDate:	2023-06-07T04:30:56Z
xmp:ModifyDate:	2023-06-07T04:30:56Z
xmpMM:DocumentID:	uuid:1b499bed-4ce8-4c0c-b682-0d58baae1cbe
xmpTPg:NPages:	16

3.4. Comprimir un archivo de texto sencillo y abrir con *Tika* el archivo comprimido.

En este caso, se obtiene un mejor resultado si se usa la aplicación de ventana, por lo que los pasos son los siguientes:

1. Ejecutamos la aplicación gráfica: `java -jar tika-app-2.9.1.jar`.
2. Arrastramos el archivo comprimido a la ventana.
3. En el apartado *View* seleccionamos la opción *Recursive JSON*, ya que este ofrece un resultado más preciso y profundo que el resto.

Para un archivo de texto sencillo, el resultado que obtenemos es el siguiente:



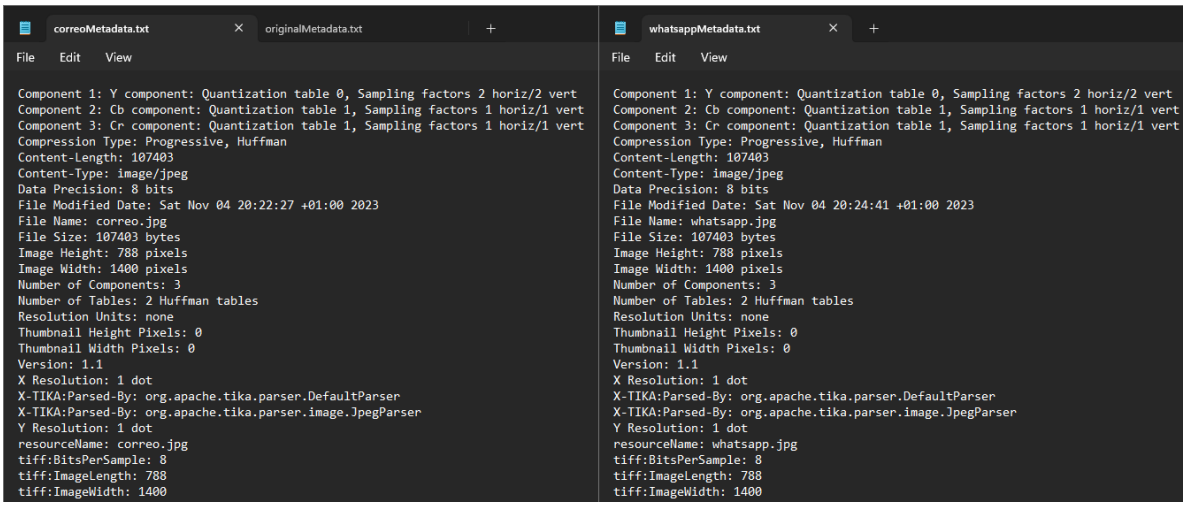
Podemos ver que obtenemos información detalla del comprimido, así como una previsualización del contenido del archivo de texto incrustado.

3.5. Pasar por correo y *WhatsApp* una foto y comparar los metadatos contrastando las diferencias.

Tras obtener los metadatos para las 3 fotos (original, correo y WhatsApp), se puede observar, tras varios intentos buscando resultados diferentes, que la salida es idéntica en los 3 casos.

- El comando utilizado es el siguiente: `java -jar tika-app-2.9.1.jar imagen.jpg`.

Se adjunta una captura que respalda el escenario descrito:



3.6. Ver los metadatos de la página principal de la UCA y guardarlo en un archivo.txt.

El comando a utilizar es el siguiente:

- `java -jar tika-app-2.9.1.jar --metadata https://www.uca.es/ > metadatosUCA.txt`.

Salida obtenida:

Content-Encoding:	UTF-8
Content-Language:	es-ES
Content-Length:	134162
Content-Type:	text/html; charset=UTF-8
X-TIKA:Parsed-By:	org.apache.tika.parser.DefaultParser
X-TIKA:Parsed-By:	org.apache.tika.parser.html.HtmlParser
X-UA-Compatible:	IE=EDGE
dc:title:	Portal UCA Portal principal de la Universidad de Cádiz
generator:	WPML ver:4.4.9 stt:1,4,2;
og:description:	Campus Jazz Cádiz / Puerto Real 2020
og:image:	
og:title:	Campus Jazz Cádiz / Puerto Real 2020
og:type:	website
og:url:	https://www.uca.es/campus-jazz-cadiz-puerto-real-2020-2/
robots:	max-image-preview:large
viewport:	width=device-width

3.7. Pasar un archivo RDF a DOC. ¿Pasar el archivo DOC o RDF al formato PDF dará error?

3.7.1. RDF → DOC

- `java -jar tika-app-2.9.1.jar celebs_fixed.rdf > celebs_fixed.doc`

3.7.2. RDF/DOC → PDF

Esta conversión no es posible, ya que no existe ningún protocolo implementado en *Tika* que permita realizar tal conversión.
RDF es un formato para datos estructurados, como XML, mientras que PDF tiene un formato para documentos formateados visualmente, Para llevar a cabo tal conversión, primero deberíamos formatear la información estructurada en información formateada visualmente.

3.8. Decir cuáles de las imágenes proporcionadas se sacaron con un producto perteneciente a Apple Computer Inc.

Tras analizar los metadatos de las 3 imágenes, observamos lo siguiente:

1. `q.jpg`: Foto tomada con un dispostivo Apple Computer Inc.
2. `r.jpg`: Foto tomada con un dispositivo Apple Computer Inc.
3. `s.jpg`: Foto tomada con una cámara digital FinePix A500.

Por tanto, las imágenes 1 y 2 se sacaron con un producto perteneciente a Apple Computer Inc.

3.9. Describir el procedimiento para, primero, guardar el contenido de una web cualquiera en un archivo HTML, y luego este convertirlo en DOC. Además, indique cómo comprobar los metadatos de este último.

Procedimiento a seguir:

1. Se almacena el contenido de una página cualquiera en un archivo DOC:
   - `java -jar tika-app-2.9.1.jar --text URL >archivo.html`
2. Se convierte el archivo HTML a un documento en formato DOC:
   - `java -jar tika-app-2.9.1.jar --text archivo.html >archivo.doc`
3. Se consulta los metadatos del documento DOC generado:
   - `java -jar tika-app-2.9.1.jar --metadata archivo.doc`

3.10. Descargar tres imágenes y contrastar las diferencias que encontramos en los metadatos.

Metadato	Imagen 1	Imagen 2	Imagen 3
Número de Tablas Huffman	2	2	2
Tipo de Compresión	Progressive, Huffman	Progressive, Huffman	Progressive, Huffman
Precisión de Datos	8 bits	8 bits	8 bits
Número de Componentes	3	3	3
Dimensiones de la Imagen	512x509 pixels	728x977 pixels	1124x1074 pixels
Tamaño del Archivo	38906 bytes	64074 bytes	70863 bytes
Fecha de Modificación del Archivo	05 Nov 03:29:05 2023	05 Nov 03:29:01 2023	05 Nov 03:28:53 2023
Nombre del Archivo	imagen1.jpg	imagen2.jpg	imagen3.jpg
Tipo de Contenido	image/jpeg	image/jpeg	image/jpeg
Unidades de Resolución	none	(No especificado)	inch
Resolución en X	72 dots	(No especificado)	72 dots
Resolución en Y	72 dots	(No especificado)	72 dots
Analizado Por	Tika	Tika	Tika

Tabla comparativa de las 3 imágenes descargadas

3.10.1. Similitudes

1. Todas las imágenes han usado el mismo tipo de compresión y codificación (Progressive, Huffman, 2 tablas).
2. Todas las imágenes tienen 8 bits de profundidad/precisión.
3. Todas las imágenes son JPEG, por lo que las 3 tienen 3 números de componentes, característica común en los archivos JPEG.

3.10.2. Diferencias

1. Todas las imágenes son de distinto tamaño y dimensiones.
2. Para la segunda imagen no se proporciona información relevante a las otras dos imágenes (No especificado).

3.11. Describir los pasos para integrar *Tika* en *Eclipse*.

En este caso, se ha descrito el procedimiento a seguir para **integrar *Tika* en** el IDE ***IntelliJ***, de *JetBrains*. El procedimiento de configuración para *Ecplise* es muy similar, siendo la única motivación de su desuso el evitar instalar y configurar un nuevo entorno desde cero.
Pasos a seguir:

1. **Crear o abrir un proyecto**
   - Debemos asegurarnos de que está seleccionado o establecido un JDK. Esto es esencial para compilar y ejecutar luego el programa.
2. **Configuración del proyecto**
   En este paso, importaremos el módulo *Tika* para su uso en el autocompletado, compilación y ejecución del código.
   a) En *IntelliJ*, abrimos la estructura del proyecto yendo a $File > Project\,Structure > Modules$ y pulsamos sobre la sección *Dependencies*.
   b) Haga click en el signo '+' y seleccione la opción '*JARs or directories*'.
   c) Busque y seleccione el archivo '`tika-app-2.9.1.jar`'.
   d) Una vez añadido, asegúrese de que está marcado como '*Compile*' y luego pulse '*OK*'.
3. **Crear una clase Java para el procesamiento y extracción del contenido del PDF**
   Para ello, deberemos desarrollar un código que utilice las librerías de *Tika*.
   Su código podría ser algo parecido a esto:

```java
import org.apache.tika.Tika;
import org.apache.tika.exception.TikaException;
import java.io.IOException;
import java.nio.file.*;

public class tikaIntelliJ {
    public static void main(String[] args) throws IOException, TikaException{
        Tika tika = new Tika();

        String contenido = // Extraer el contenido del archivo PDF
                tika.parseToString(Files.newInputStream(Paths.get("Tika.pdf")));

        System.out.println("Contenido del PDF:");
        System.out.println(contenido); // Imprimir el contenido extraido
    }
}
```

4. **Ejecutar el código y obtener el contenido del documento PDF**
   Llegados a este paso, bastaría con pulsar el botón de ejecutar para obtener la salida esperada.