Práctica 5: Búsqueda de información mediante la librería Lucene

Sistemas de Información

Grado de Informática
Dpto. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza
Escuela de Ingeniería y Arquitectura

11 de diciembre de 2019

1. Objetivos

Hoy en día numerosos Sistemas de información emplean técnicas de búsqueda y recuperación de información (Information Search and Retrieval -ISR-) para obtener documentos electrónicos y cualquier tipo de colección documental digital (metadatos de diferentes elementos, contenidos de bases de datos relacionales, etc.) con el objetivo de emplear la información y datos relevantes de la forma pertinente.

En esta práctica se presenta la librería **Apache Lucene**. Apache Lucene permite incorporar mecanismos de indexación y búsqueda de información en aplicaciones implementadas en diversos lenguajes de programación (Java, .Net, Python, etc.). En mayor detalle, los objetivos de esta práctica son:

- Familiarizarse con la librería Lucene probando el ejemplo proporcionado.
- Implementar una aplicación Java con interfaz de texto que permita:
    - La **indexación** de los contenidos de un determinado directorio.
    - La **incorporación** de un nuevo contenido al índice
    - La **búsqueda** de documentos mediante preguntas basadas en palabras clave.
- Añadir la funcionalidad de búsqueda basada en palabras clave en las noticias, preguntas y retos del portal medioambiental desarrollado en las prácticas anteriores (opcional).

El entorno de desarrollo propuesto emplea JDK 1.8.

2. Contenidos

La práctica consta de tres bloques. El primer bloque consiste en realizar pruebas con el código proporcionado y responder a una serie de preguntas relacionadas con la librería Lucene. En el segundo bloque, se pretende desarrollar una aplicación con interfaz de texto, que permita la indexación y búsqueda de documentos en un determinado directorio. Por último, el tercer bloque es opcional y requiere ampliar la aplicación Web construida en sesiones anteriores.

2.1 Introducción a la librería Lucene

Lucene permite la creación de un *índice inverso* en el que se añaden documentos para posteriormente buscarlos. Las clases fundamentales de la librería se enumeran a continuación:

- **IndexWriter**: Permite la construcción y actualización del índice inverso que va a emplear el sistema de recuperación de información. En Lucene los índices son auto-incrementales, es decir una vez creados se pueden ir añadiendo a ellos documentos susceptibles de ser indexados.
- **Document**: Los documentos son la unidad de indexación y búsqueda en Lucene. Un objeto de este tipo representa un único documento. No obstante, se debe señalar que un objeto de este tipo no tiene por qué corresponder con un fichero, sino que constituye la unidad de indexación del sistema de búsqueda construido, por ejemplo, podría representar un registro de una determinada base de datos, un párrafo de un texto... Es el elemento al que apunta el índice inverso. Los objetos Document se modelan como un conjunto de campos (Fields) de la forma (nombre, valor), y a la hora de indexar, podemos decidir qué campos de los documentos queremos indexar.
- **Field**: Cuando se crea un objeto Field además del nombre del campo y su valor, se debe indicar si se desea almacenar el valor del campo en el índice o no. En el caso de que el valor del campo sea razonablemente pequeño, Lucene permite que sea almacenado en el índice. Los campos almacenados en el índice, en lugar de utilizar el objeto Document para localizar los datos originales, pueden emplear directamente el índice. En general cuando se desea tokenizar el valor del campo se emplea la subclase TextField, mientras que si no se desea tokenizar se emplea StringField. Gracias a esta división en campos, es posible tratar con documentos que posean cierta estructura, por ejemplo, para indexar correos electrónicos, se pueden considerar cuatro campos que contengan: el remitente, los destinatarios, el título o subject y el contenido del correo; respectivamente.
- **Analyzer**: Los analizadores son los encargados de procesar los textos para obtener sus correspondientes representaciones internas (vista lógicas). Se emplean tanto en la creación de índices como en los procesos de búsqueda, ya que las queries deben ser procesadas con el mismo criterio que los documentos indexados. Lucene proporciona diversos analizadores, por ejemplo, el *SimpleAnalyzer* que simplemente divide el texto en palabras y convierte todo a minúsculas o el *StandardAnalyzer* que además elimina

stopwords (por defecto, las del idioma inglés).

- **IndexSearcher**: Constituye la base para realizar búsquedas en un índice de documentos. Para realizar una búsqueda se emplea el método search(), que proporciona los documentos que satisfacen las condiciones de la búsqueda indicadas en un objeto Query.
- **Query**: Es una clase abstracta que proporciona los mecanismos necesarios para formular las necesidades de información.
- **Hit**: Los objetos hit representan cada uno de los resultados obtenidos al realizar cierta consulta en un determinado índice. A través de estos objetos se puede acceder al documento al que se refiere el resultado. Además, cada hit tiene asociada una puntuación (score) que representa la relevancia del documento asociado al Hit para la pregunta formulada.

2.2 Bloque 1: Testeo de la librería Lucene

Para facilitar la familiarización con la biblioteca Lucene, se proporciona un código de ejemplo en el que se usa la librería. En el ejemplo, se indexan cuatro ficheros que contienen noticias relacionadas con el medio ambiente. (uno.txt, dos.txt, tres.txt y cuatro.txt) y se crea el índice en una carpeta. Posteriormente se realizan varias búsquedas sobre el índice y se muestran los resultados. El código de ejemplo está encapsulado en un proyecto "eclipse", para facilitar su manejo desde este IDE.

Realizar las siguientes consultas e indicar cuál es el resultado de la búsqueda para cada una de ellas:

1. contaminación
2. cambio climático
3. por
4. aeropuerto

Podéis probar a hacer variaciones sobre las consultas. Por ejemplo,

- ¿Qué pasa si utilizamos el "StandardAnalyzer" en lugar del "SimpleAnalizer"? ¿Qué función tiene el fichero "stopwords.txt"?
- Qué ocurre si en la búsqueda ponemos "contaminacion" o "cambio climatico" (sin tildes)?
- ¿Y si hacemos esta búsqueda utilizando el "SpanishAnalizer"? ¿Por qué ocurre esto?
- ¿Qué ocurre si re-indexamos todos los ficheros cada vez que ejecutamos el programa, en lugar de, simplemente, reabrir el índice creado previamente?

2.3 Bloque 2: Indexación y búsqueda de documentos de un determinado directorio

Desarrollar una aplicación que realice búsquedas de documentos en un determinado directorio. El directorio puede contener subdirectorios con documentos que también deben ser indexados. El tamaño del directorio puede ser considerable. El índice se creará en un directorio. Además, en general, los documentos están escritos en castellano (utilizar un analizador adecuado para este lenguaje).
La aplicación ofrecerá un menú inicial en modo texto:

1.- Indexar un directorio
2.- Añadir un documento al índice
3.- Buscar término
4.- Salir

La primer opción solicitará un directorio a indexar.
La segunda opción solicitará un fichero que añadir al índice
La tercera opción solicitará un término a buscar, y mostrará los resultados.

Tras cada una de las operaciones, la aplicación volverá a mostrar el menú inicial, hasta que el usuario pulse la opción 4.

2.4 Bloque 3: Búsqueda de datos almacenados en una base de datos basada en palabras clave (opcional)

Modificar la aplicación web del portal medioambiental realizado en las prácticas anteriores, de manera que incorpore un buscador sobre el contenido del mismo, basado en palabras clave.
Para ello, será necesario que las funciones de inserción de contenido (noticias, foros, páginas estáticas...) en la base de datos añadan una llamada a una función de indexación de contenido. Esta función deberá indexar Documents que incorporen, en distintos campos (Fields) la información necesaria:

- Tipo de documento (tabla / directorio...): Noticia, Registro B.D., Pregunta, Comentario...
- Identificador único del documento en la tabla correspondiente (que permita automatizar el acceso al dato).
- Contenido del documento que queremos indexar

Eso permitirá, al hacer una búsqueda sobre el índice, recuperar documentos y saber en qué tabla y con qué ID único debemos buscarlos.

3. Entrega de la práctica

La práctica se realizará en grupos de tres personas. Cuando se finalice la práctica se debe entregar en un fichero denominado práctica5_NIA1_NIA2_NIA3.tar o práctica5_NIA1_NIA2_NIA3.zip (donde NIA1, NIA2 y NIA3 son los NIAs de los autores de la práctica) con el siguiente contenido:

1. Un fichero de texto denominado autores.txt que contendrá el NIA, los apellidos y el nombre de los autores de la práctica en las primeras líneas del fichero. Por ejemplo:

NIA	Apellidos	Nombre
545689	Rodríguez Quintela	Sabela
745689	Caamaño Cives	Antón
677893	López Marín	Adrián

2. Un directorio denominado proyectoObligatorio que contendrá los ficheros fuente del programa de indexación y búsqueda de términos en un índice.
3. (opcional) Un directorio llamado proyectoOpcional con los ficheros fuente modificados en el proyecto de portal medioambiental, para incluir la funcionalidad de búsqueda indexada.
4. Un fichero denominado memoriaPractica5.pdf cuya extensión no excederá de las 10 páginas donde se indiquen:
    a. los diseños realizados
    b. Las respuestas y reflexiones del bloque 1 de la práctica
    c. La metodología de trabajo empleada para el desarrollo de la práctica (recursos, herramientas utilizadas, distribución del trabajo, horas de trabajo, etc.), las dificultades encontradas durante la realización de la práctica.

Al descomprimir el fichero .tar se deben extraer los ficheros y directorios indicados en el directorio practica5_Nip1_Nip2_y_Nip3.tar. Es importante seguir las convenciones de nombrado y la estructura de ficheros y directorios descrita.

Para la realización de esta práctica se han planificado una sesión

Por tanto, la fecha límite de entrega es el día anterior (a las 23:59) a la novena sesión de prácticas en el laboratorio. Para la entrega del fichero .tar, se utilizará el enlace correspondiente disponible en la página de la asignatura en el Anillo Digital Docente (ADD)

de la Universidad de Zaragoza en la plataforma Moodle 2 (http://moodle2.unizar.es). En caso de que no sea posible realizar la entrega a través del ADD se enviará dicho tar al profesorado de la asignatura mediante correo electrónico.

4.- Procedimiento de corrección y recomendaciones

Una vez realizada la práctica y entregada esta, cada grupo debe presentársela al profesorado de prácticas en la sesión de prácticas correspondiente al inicio de la práctica 6. Al realizar la presentación el profesorado le formulará cuestiones sobre las decisiones de diseño e implementación que se ha realizado el grupo de trabajo.

La práctica debe entregarse en los términos indicados anteriormente, funcionar correctamente y no haber sido copiada. También es importante someter código limpio (donde se ha evitado introducir mensajes de depuración y comentarios que no proporcionan información al usuario). Los criterios de corrección se dividen en dos bloques:

- *Requisitos mínimos a cumplir (10 puntos)*
    - Análisis realizado en el bloque 1 de la práctica (3 puntos).
    - Aplicación de indexado y búsqueda sobre directorio (4 puntos)
    - Memoria técnica de realización el proyecto (3 puntos).
- *Bloque opcional (2 puntos)*
    - Desarrollo e integración del sistema de búsqueda propuesto en el bloque opcional (2 puntos).