Jakarta Lucene

Eine Java-Bibliothek zur Suchindex-Erstellung

Seminararbeit
Tilman Schneider

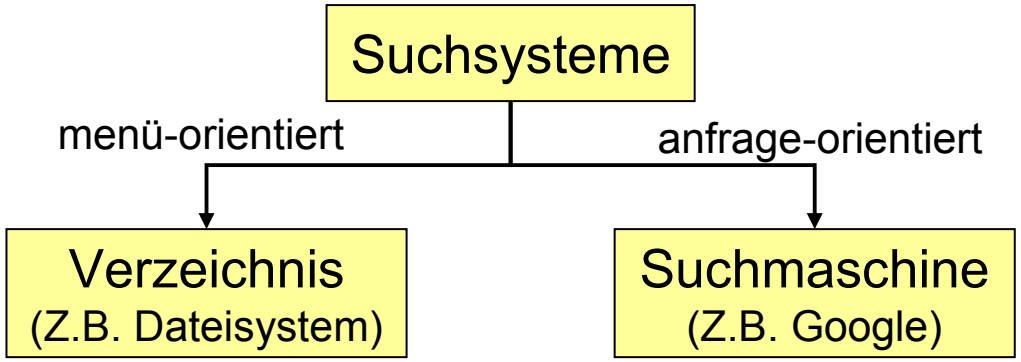
Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Definition: Suchmaschine



„Suchmaschine: Ein System zur Volltextsuche in großen Datenmengen“

Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Was ist Jakarta Lucene?

- Projekt der Apache Jakarta Gruppe
- 100% pure Java.
  Mittlerweile gibt es auch Implementierungen in C++, .NET, Python und Perl.
- Vielseitig einsetzbar
- Keine Applikation, sondern API

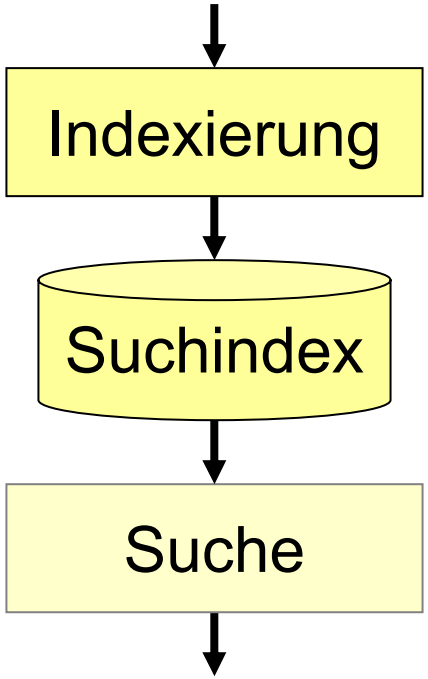
Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Wozu ein Suchindex?

- Herkömmliche Volltextsuche sehr langsam: O(n), n = Größe des Datenbestands
- Daher: Aufbereitung des Suchraums in einer geeigneten Datenstruktur

Gesamtablauf



- Durchforsten des Suchraums nach Texten
- Aufbereitung des Textes
- Eigentliche Indexierung

- Interpretation der Anfrage
- Eigentliche Suche
- Ergebnisaufbereitung

Umgekehrter Wortindex

Originaltext

```
Blaukraut bleibt Blaukraut und
Brautkleid bleibt Brautkleid
```

blaukraut	0, 17
bleibt	10, 42
brautkleid	31, 49
und	27

Suchindex

Textaufbereitung

Durchforsten des Suchraums nach Texten

↓

Textaufbereitung

Nicht übel, sprach der Dübel!

Nicht	übel	sprach	der	Dübel	Zerlegung
	übel	sprach		Dübel	Stopword-Filter
	ueb	sprach		dueb	Stemming

↓

Indexierung

Lucene: Documents

- Für jeden zu indizierenden Text wird ein Document-Objekt erzeugt.
- Ein Document ist eine Menge von Feldern
- Ein Feld ist ein Key/Value-Paar
- Beispiele für mögliche Felder:
  - Der Text selbst
  - Die URL des Textes
  - Das Datum der letzten Änderung
  - Die Größe des Textes

Lucene: Fields

- Für jedes Feld wird angegeben:
  - Ob es gespeichert wird (store-Flag)
  - Ob es indexiert wird (index-Flag)
  - Ob es vor der Indexierung zerlegt wird (token-Flag)

Fabrikmethode	store	index	token
`Field.Text(String name, String value)`	x	x	x
`Field.Text(String name, Reader value)`		x	x
`Field.Keyword(String name, String value)`	x	x	
`Field.UnIndexed(String name, String value)`	x		
`Field.UnStored(String name, String value)`		x	x

Codebeispiel: Datei-Indexierung 1

```
private static void addToIndex(IndexWriter index, File file) {
  InputStream is = new FileInputStream(file);

  // Wir erzeugen ein Document mit zwei Feldern:
  // Eines mit dem Pfad der Datei und eines mit seinem Inhalt
  Document doc = new Document();
  String fileName = file.getAbsolutePath();
  doc.add(Field.UnIndexed("path", fileName));
  doc.add(Field.Text("body", new InputStreamReader(is)));

  // Document zu Index hinzufügen
  index.addDocument(doc);
  is.close();
}
```

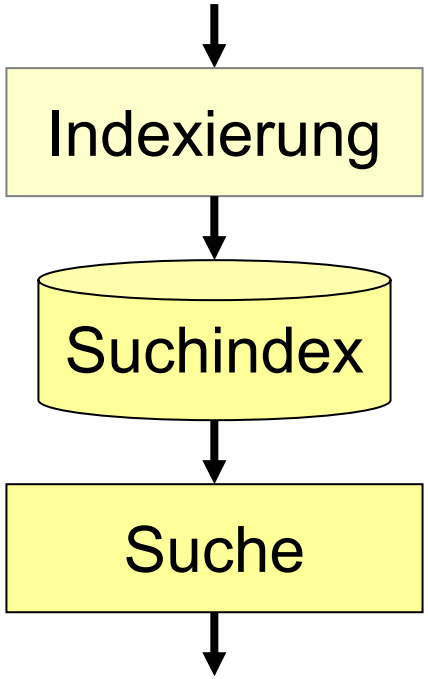
Codebeispiel: Datei-Indexierung 2

```
public static void main(String[] args) throws Exception {
  // Neuen Index erzeugen
  String indexPath = args[0];
  IndexWriter index
    = new IndexWriter(indexPath, new GermanAnalyzer(), true);

  // Gegebenes Verzeichnis durchsuchen
  File dir = new File(args[1]);
  crawlDirectory(index, dir);

  // Index optimieren und schließen
  index.optimize();
  index.close();
}

private static void crawlDirectory(IndexWriter index, File dir) {
  // Alle Dateien und Unterverzeichnisse zum Index hinzufügen
  File[] childArr = dir.listFiles();
  for (int i = 0; i < childArr.length; i++) {
    if (childArr[i].isDirectory()) {
      crawlDirectory(index, childArr[i]);
    } else {
      addToIndex(index, childArr[i]);
    }
  }
}
```

Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Gesamtablauf



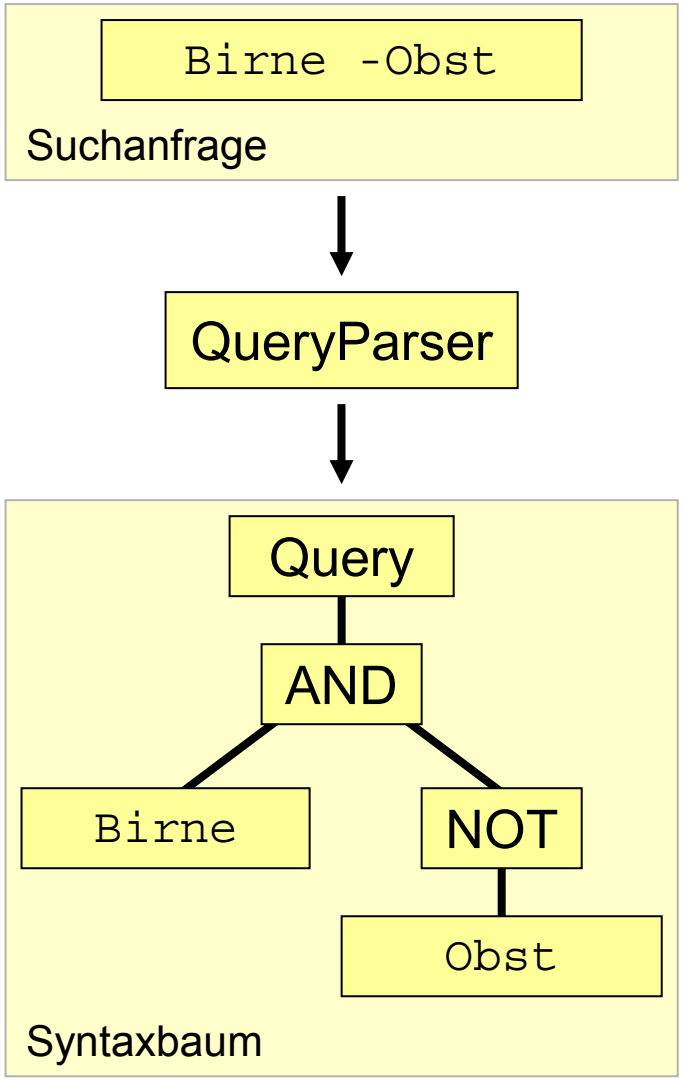
- Durchforsten des Suchraums nach Texten
- Aufbereitung des Textes
- Eigentliche Indexierung

- Interpretation der Anfrage
- Eigentliche Suche
- Ergebnisaufbereitung

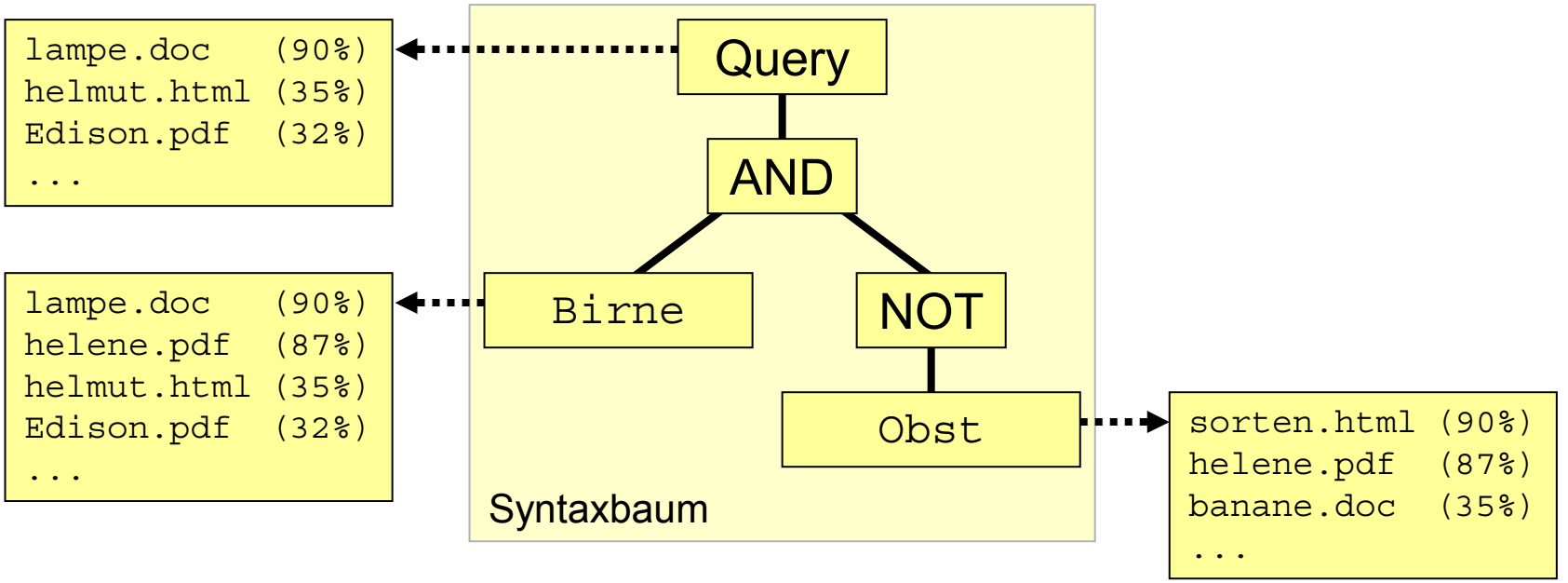
Lucene Querysyntax

- Lucene bietet mächtige QuerySyntax:
  - Bool‘sche Operatoren:
    „`Lucene and Seminar`“ oder „`+Lucene -Seminar`“
  - Suche in Feldern: „`Lucene title:Seminararbeit`“
  - Gruppierung: „`(Jakarta and Lucene) or Seminar`“
  - Wildcards: „`te?t`“ oder „`text*`“
  - Fuzzy-Suche: „`Maier~`“
  - ...

Interpretation der Suchanfrage



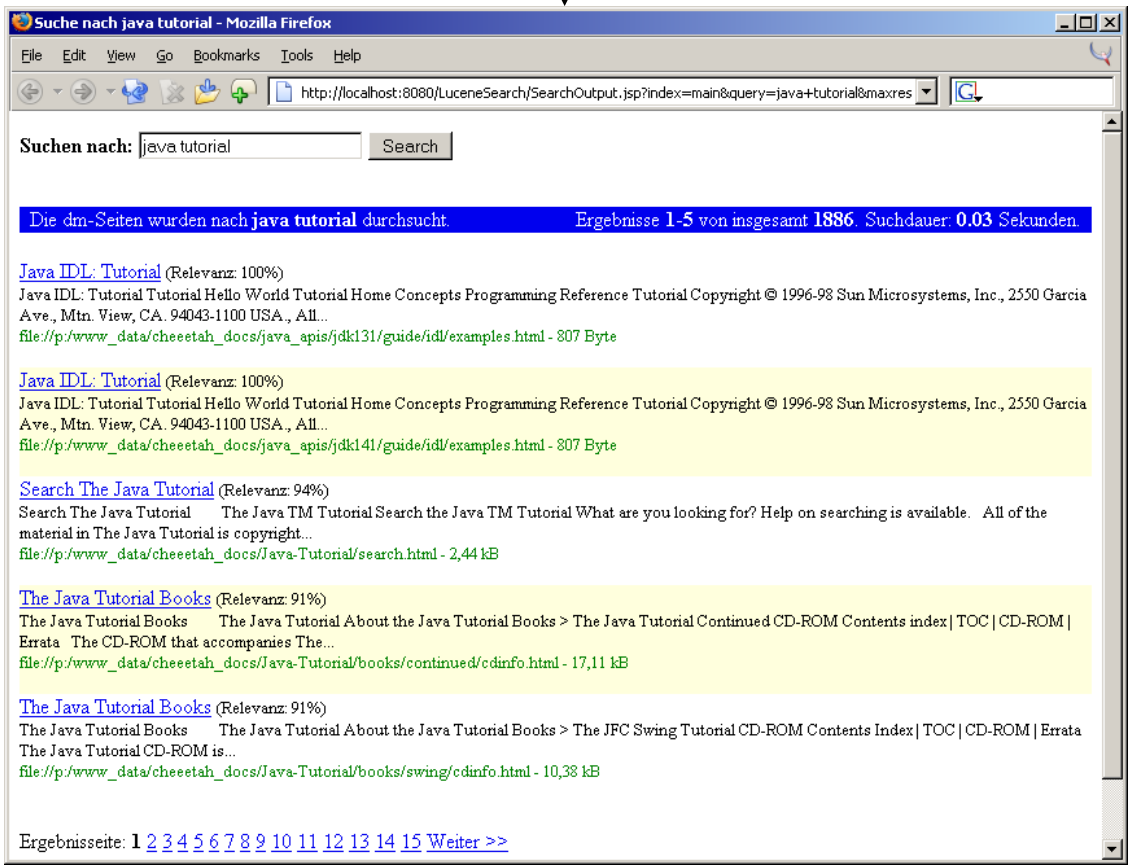
Eigentliche Suche



Ergebnisaufbereitung

```
lampe.doc    (90%)
helmut.html  (35%)
Edison.pdf   (32%)
...
```

Darstellungsschicht
(z.B. JSP-Seite)



Codebeispiel: Suche

```
public class SimpleSearcher {

  public static void main(String[] args) throws Exception {
    String indexPath = args[0];
    String queryString = args[1];

    Query query = QueryParser.parse(queryString, "body",
                                    new GermanAnalyzer());

    Searcher searcher = new IndexSearcher(indexPath);
    Hits hits = searcher.search(query);

    for (int i = 0; i < hits.length(); i++) {
      System.out.println(hits.doc(i).get("path")
                         + " (score: " + hits.score(i) + ")");
    };
  }

}
```

Agenda

- Definition: Suchmaschine
- Vorstellung von Jakarta Lucene
- Erstellung eines Suchindex
- Suchen mit dem Suchindex
- Fazit

Fazit

- Vorteile von Lucene:
  - Einfache API
  - Trotzdem hohe Funktionaliät
  - Sehr ressourcenschonend (Kleiner Index, Wenig Ansprüche an CPU und Arbeitsspeicher)
  - Alle Module vom Analyzer über den QueryParser bis zur Persistenz des Index sind austauschbar.
  - Vielseitig: Lässt sich in jedes Projekt integrieren, das eine Suchfunktion benötigt.
- Nachteil:
  - Lucene bietet nur den Kern einer Suchmaschine, die Darstellung des Suchergebnis und die Aufbereitung des Suchraums muss man selbst übernehmen.

Regain

- Open Source Projekt
- Bietet Crawler für Dateisystem und HTTP
- Kann viele geläufige Dateiformate lesen: HTML, XML, Excel, Powerpoint, Word, PDF and RTF
- Bietet TagLibrary für einfache Einbindung der Suchergebnisse in die eigene JSP-basierte Website.

Referenzen

- Lucene: http://jakarta.apache.org/lucene
- Regain: http://regain.sf.net
- Java World-Artikel über Lucene:
  http://www.javaworld.com/javaworld/jw-09-2000/jw-0915-lucene.html
- Lucene-Tutorial: http://darksleep.com/lucene