

Introduction to



Lucia D. Krisnawati

Overview

- Introduction to Lucene & Solr
- Getting started
 - Indexing using Solr
 - Updating & deleting files
 - Searching using Solr
- Solr Configuration

What is Lucene?

- Lucene is:
 - NOT a crawler
 - see Apache Nutch
 - NOT an application
 - See PoweredBy on the Wiki
 - NOT a library for doing Google pageRank
 - An open source Java-based IR **library** enabling text based search
 - Metaphor: Lucene is an engine

What is Solr?

- Solr is:
 - An open source enterprise search server
 - Based on the Lucene Java search library
 - A web based application that processes HTTP request and returns HTTP responses
 - completed with XML/HTTP APIs, caching, replication, and web administration interface.
 - Metaphor: Solr is a car

Why Solr?

- Some reasons of using Solr:
 - Using many Lucene best practices
 - uncomplicated setup, configuration and Easy to extend
 - Providing faceted navigation, spell checking, highlighting, clustering, grouping, & any other search features
 - Supporting clients in:
 - HTTP
 - Java
 - Python
 - PHP
 - Ruby
 - JSON

Why Solr?

- Some reasons of using Solr:
 - Flexible index formats
 - New posting list codecs: block, simple text, Append (HDFS)
 - Good indexing performance
 - SolrCloud feature (Solr 4.x above)
 - Geospatial searches
 - Who uses Lucene/Solr?
 - Cisco, ebay, Boeing, AT& T, Ford and many, many others....!

Comparison to Database Technology

- The most important comparison to make is the data model
 - Data model is the organizational structure of data
- RDBMS:
 - Its data model is based on multiple tables with lookup keys between them
 - A **join** capability for querying across tables
 - A flexible data model
- Lucene Solr:
 - Has a more limiting **document oriented** data model
 - Analogous to a single table without join possibilities
 - Document-oriented databases have a rich nested structure similar to XML/JSON → MongoDB (NoSQL)
 - Has a flat document structure
 - Supporting multi-valued fields with an array of values

Comparison to Database Technology

- RDBMS:
 - Excell at:
 - insert/update efficiency, in-place schema changes,
 - multi-user access control, bulk data retrieval
 - Supporting rich ad-hoc query features
- Solr:
 - Falls short in all of above areas:
 - **No Updates:** if any part of a document in Solr needs to be updated, the entire document must be replaced.
 - **Slow commits:**
 - Solr's search performance & certain features are made due to extensive caches.
 - When a commit operation is done to finalize added documents, the caches are rebuilt.

Solr as NoSQL

- NoSQL : not only SQL
- Characteristics:
 - Non-traditional data stores
 - Not designed for SQL type query
 - Document oriented, data format agnostic (JSON, XML, CSV, binary)
- Versioning and optimistic locking
 - with Real Time GET, allows read/write/update with without conflict.
- Atomic updates:
 - Can add/remove/change and increment a field in existing index with/witout re-indexing

Important Terminologies in Solr

- A Lucene Index is a collection of documents
- A document is a collection of fields
- A field is a content along with metadata describing the content
- Field content can have several attributes, eg:
 - Tokenized – analyze the content, extracting Tokens and adding them to inverted index.
 - Stored – keep the content in a storage data structure for use by application.

```
<field name="id" type="string" indexed="true"
stored="true" required="true"
multiValued="false" />
```

Important Terminologies in Solr

- Solr Core:
 - A running instance of a Lucene index along with all Solr configurations required to use it
 - A single application may have 0 or more cores which are run in isolation
- Request Handler:
 - A Solr component that processes requests.
- Commit:
 - Solr always attempts to optimize the rate of incoming data that can be indexed by buffering data in memory before writing it to the index.
 - The downside is that data is not available for queries until it has been written to the index.
 - A commit operation is necessary to write all the buffered data to the index & make it available for queries.

Getting Started

- <https://lucene.apache.org/solr/tutorial.html>

- Unzip your Solr release

```
user:~solr$ unzip -g solr-version.zip
```

```
user:~solr$ tar xvzf solr-version.tgz
```

- Go to the solr directory and change your working directory to the 'example' directory:

- To the example path and type:

```
user:~solr/example$ java -jar start.jar
```

- Under windows, start the Web Server by running *start.bat* instead.
c:\Applications\solr\example > start.bat

- Solr can run in any Java Servlet Container & the example index includes an installation of Jetty

- The 'start.jar' command:

- launches Jetty with the Solr WAR
- Launches the example configs
- starts up the Jetty application server on port 8083

Getting Started

- Use your terminal to display the logging information from Solr.
- Solr is running in your port 8983
- Check it by:
 - Open your browser and type:
<http://localhost:8983/solr/>
- Your Solr server is running but it has no data or document at all
- Modifying a Solr index can be done by POSTing commands in variety of formats:
 - XML
 - JSON
 - CSV
 - JAVABIN

Solr Core Admin

localhost:8983/solr/#/~cores/collection1



+ Add Core

✂ Unload

🔄 Rename

🔄 Swap

🔄 Reload

🔧 Optimize

collection1

Core

startTime: about an hour ago
instanceDir: /home/lucia/solr-4.8.1/example/solr/collection1/
dataDir: /home/lucia/solr-4.8.1/example/solr/collection1/data/

Index

lastModified: -
version: 1
numDocs: 0
maxDoc: 0
deletedDocs: -
optimized: ✓
current: ✓
directory: org.apache.lucene.store.NRTCachingDirectory:NRTCachingDirectory(MMapDirectory@/home/lucia/solr-4.8.1/example/solr/collection1/data/index lockFactory=NativeFSLockFactory@/home/lucia/solr-4.8.1/example/solr/collection1/data/index; maxCacheMB=48.0 maxMergeSizeMB=4.0)

Dashboard

Logging

Core Admin

Java Properties

Thread Dump

Core Selector

Solr Admin User Interface



Dashboard

Logging

Core Admin

Java Properties

Thread Dump

collection1

Overview

Analysis

Dataimport

Documents

Files

Ping

Plugins / Stats

Query

Replication

Schema Browser

Statistics

Last Modified: about 9 hours ago

Num Docs: 15

Max Doc: 16

Heap Memory: 580

Usage:

Deleted Docs: 1

Version: 30

Segment Count: 1

Optimized:

optimize now

Current:

Instance

CWD: /home/lucia/solr-4.8.1/example

Instance: /home/lucia/solr-4.8.1/example/solr/collection1

Data: /home/lucia/solr-4.8.1/example/solr/collection1/data

Index: /home/lucia/solr-4.8.1/example/solr/collection1/data/index

Impl: org.apache.solr.core.NRTCachingDirectoryFactory

Replication (Master)

	Version	Gen	Size
Master (Searching)	1403429016445	12	9.7 KB
Master (Replicable)	-	-	-

+ Admin Extra

Healthcheck

Ping request handler is not configured with a healthcheck file.

Documentation

Issue Tracker

IRC Channel

Community forum

Solr Query Syntax

Solr Admin User Interface (UI)

- Pages describing each screen of admin UI:
 - **Dashboard** provides link for system-level information & Solr cores configured for this instance.
 - **Logging** explains the various logging level available and how to invoke them
 - **Core Admin** explains how to get management information about each core
 - **Java Properties** shows the java information about each core
 - **Thread Dump** lets you see detailed information about each thread, along with state information.

Solr Admin User Interface (UI)

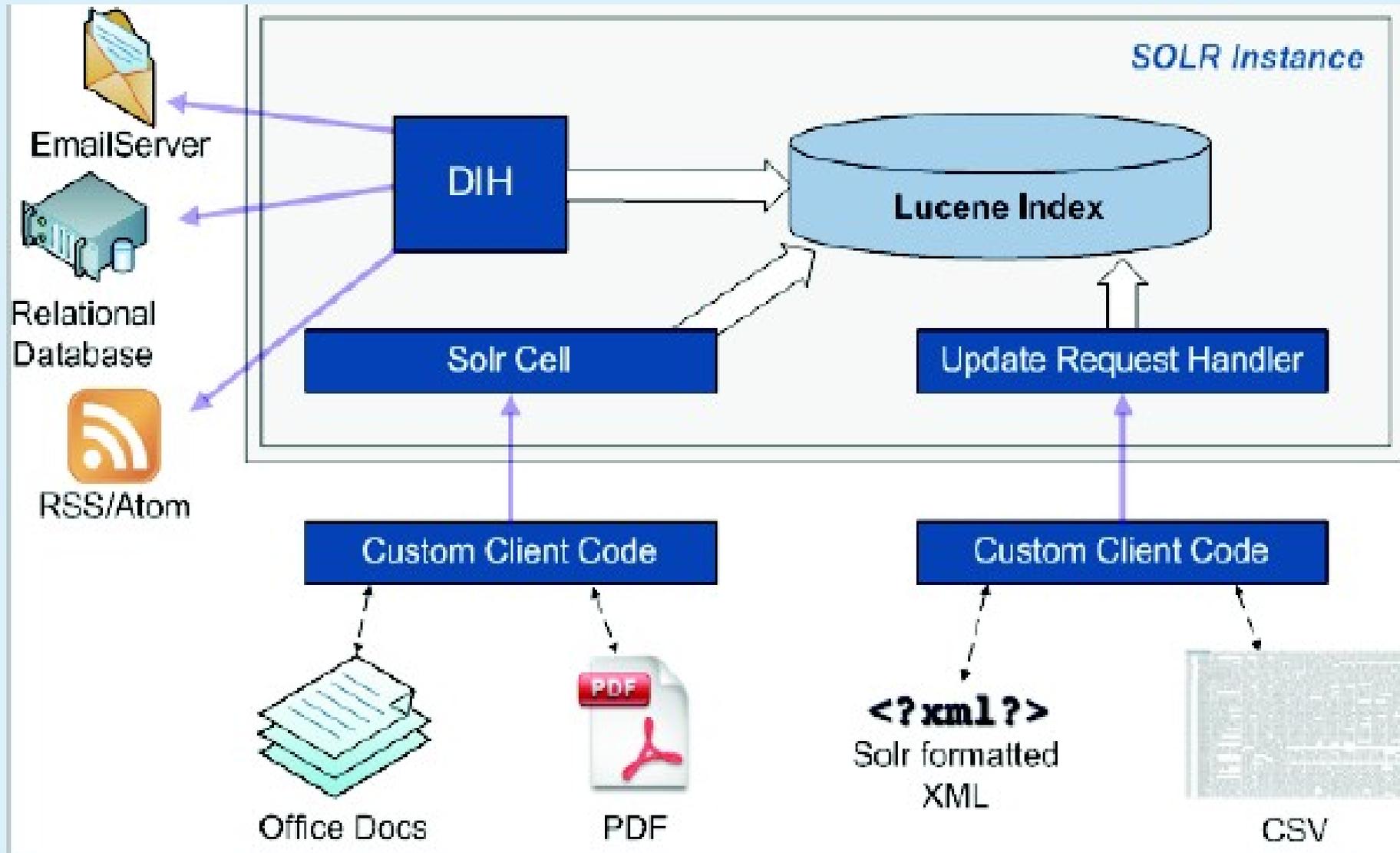
- Core-Specific Tools is a section explaining each named core:
 - **Analysis** lets us analyze the data found in specific fields
 - **Dataimport** shows information about the current status of the Data Import Handler
 - **Documents** provides a form allowing us to execute various Solr indexing commands directly from browser
 - **Files** shows the current core configuration files such as *solrconfig.xml* & *schema.xml*
 - **Ping** lets us ping a named core & determine whether it is active
 - **Plugins/Stats** shows statistics for plugins & other installed components
 - **Query** Let us submit a structured query
 - **Replication** shows the current replication status for the core
 - **Schema Browser** displays schema data in a browser window

Getting started to Indexing

- An easiest way to indexing:
 - Open a new terminal window
 - Go to *exampledocs* directory that contains sample files & SimplePost Tool, a java-based command line tool, *post.jar*
 - choose some files and run „java -jar post.jar“:

```
user:~solr/example/exampledocs$ java -jar post.jar  
doc_name.xml
```
 - To check that you have successfully indexed those document:
 - go to admin interface to „query“ tab, and enter a query(ies) relating to your indexed documents.
 - Click „execute query“ button.
 - You will see the result in the format you choose

Solr Indexing Option



[Hatcher, 2011]

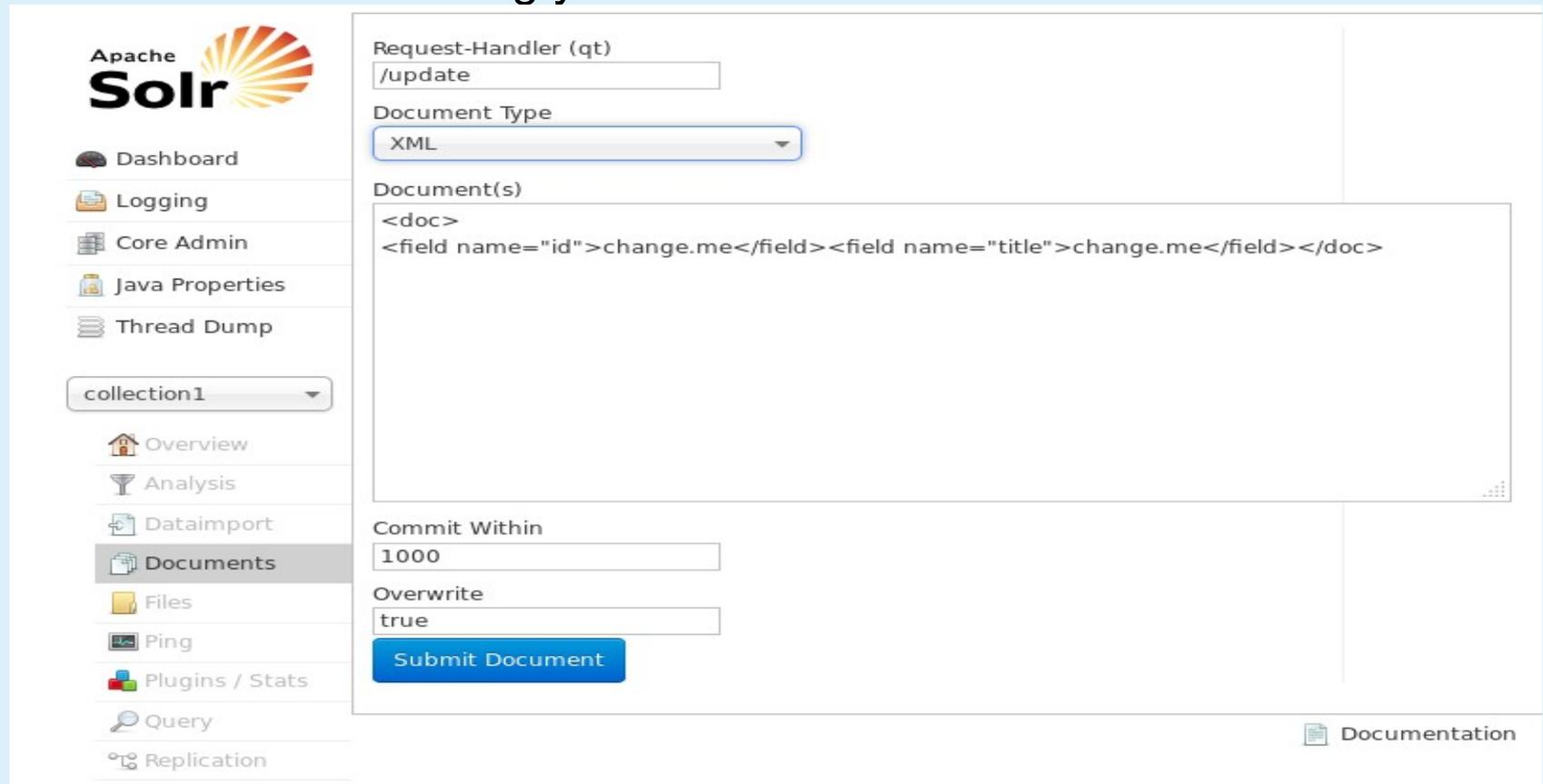
Indexing through Request Handler

- Updating a Solr Index with XML
- Techniques:
 - /update POST to with post.jar command

```
<add>
  <doc>
    <field name="id">rawxml1</field>
    <field name="content_type">text/xml</field>
    <field name="category">index example</field>
    <field name="title">Simple Example</field>
    <field name="filename">addExample.xml</field>
    <field name="text">A very simple example of
      adding a document to the index.</field>
  </doc>
</add>
```

Indexing through Request Handler

- Using Admin Interface:
 - Go to tab Documents
 - Choose the document type:
 - File upload (from your file system)
 - Creating your own document on the chosen format



Apache Solr

- Dashboard
- Logging
- Core Admin
- Java Properties
- Thread Dump

collection1

- Overview
- Analysis
- Dataimport
- Documents**
- Files
- Ping
- Plugins / Stats
- Query
- Replication

Request-Handler (qt)
/update

Document Type
XML

Document(s)
<doc>
<field name="id">change.me</field><field name="title">change.me</field></doc>

Commit Within
1000

Overwrite
true

Submit Document

Documentation 21

Indexing CSV Files

- Beside using request-handler, indexing csv files to Solr can be done by

- Sending files over HTTP:

```
cd example/exampledocs
```

```
curl http://localhost:8983/solr/update/csv --data-binary  
@books.csv -H 'content-type:text/csv; charset=utf-8'
```

- Or streaming from the file system:

```
cd example/exampledocs
```

```
curl http://localhost:8983/solr/update/csv?stream.file=  
exampledocs /data.csv&stream.contentType=text/csv;  
charset=utf-8
```

Updating Documents

- Solr uses the „UniqueKey“ to determine the „identity“ of a document
- Adding the same document to the index with the same uniqueKey as an existing document means the new document will replace the original.
- An „update“ is actually 2 steps, internally:
 - Delete a document with that id
 - Add the new document
 - So documents are „replaced“, not deleted
 - No field-level updating – a whole document has to be replaced

Deleting Documents

- Document can be deleted using SimplePost Tool that sends raw XML to a Solr port:
 - Using a delete by id:

```
<delete><id>001</id></delete>
```


user:~solr/example/exampledocs\$ java -Dcommit=false
-Ddata=args -jar post.jar "<delete><id>001</id></delete>"
 - Using a delete by query:

```
<query><delete>name:information</delete></query>
```


user:~solr/example/exampledocs\$ java -Ddata=args
-Dcommit=yes -jar post.jar
"<query><delete>name:information</delete></query>"

Deleting Documents

- When a document is deleted it still exists in an index segment:
 - The example configuration uses Solr's „autoCommit“ → automatically persist this change to the index
 - Check in the admin GUI, 'plugin/stats' for *updateHandler*
 - If *deleteByld* value drops as the *cumulative_deletesByld* & *autocommit* values increase, the delete to disk has been done.
- You can force a new searcher to be opened to reflect these changes by sending an explicit commit command:

```
java -jar post.jar -
```

Searching in Solr

- The search query is processed by a Request Handler:
 - Request Handler calls a query parser
 - Query parser interprets query's term & parameters
 - Input to a query parser can include:
 - Search strings – common terms
 - Parameters for fine tuning, eg. Boolean logic
 - Parameters for controlling the presentation of the query response, eg. Specifying the order in which results are displayed.
 - Solr supports:
 - Highlighting to relevant terms
 - Snippets → 3-4 lines of texts offering a description of a search result
 - Faceting → arrangement of search results into categories which are based on index terms.

Searching in Solr: Faceting

To see the faceting, access the Velocity sample search UI: <http://localhost:8983/solr/browse>

Faceting

The screenshot shows the Amazon.com search results page for the query "java programming". The page is divided into two main sections: a left sidebar for faceted navigation and a main content area for search results.

Faceted Navigation (Left Sidebar):

- Department:** Any Department, Books (5,216), Kindle Store (432), Software (79), Automotive (8), Movies & TV (7), VHS (7), Everything Else (5). [+ See All 9 Departments](#)
- Shipping Option (whats this?):** Any Shipping Option, Prime Eligible, Free Super Saver Shipping
- Java Programming**
- Format:** Any Format, Audiobooks (2), HTML (42), Kindle Books (96), PDF (14), Printed Books (2,487)
- Binding:** Any Binding, Hardcover (213), Paperback (2,245), School & Library Binding (4)

Search Results (Main Content Area):

Search: All Departments | java programming

"java programming"

Related Searches: [java](#), [java programming language](#), [javascript](#).

Select Results from All Departments

- Head First Java, 2nd Edition** by Kathy Sierra and B...
Buy new: ~~\$44.95~~ **\$29.67**
49 new from \$25.66 45 used from \$23.72
Get it by **Tuesday, May 18** if you order in the next **45 hours** and Eligible for **FREE Super Saver Shipping**.
★★★★★ (263)
Excerpt - Front Matter: "... The **Java Programming Language J**
[Surprise me!](#) See a random page in this book.
Books: See all 5,216 items
- Books**
Beginner's Guides to Java Programming
Java Servlets Software Programming
JavaBeans
- Effective Java (2nd Edition)** by Joshua Bloch (Paper...
Buy new: ~~\$54.99~~ **\$42.89**
37 new from \$38.45 19 used from \$38.60
Get it by **Tuesday, May 18** if you order in the next **44 hours** and Eligible for **FREE Super Saver Shipping**.
★★★★★ (42)
Excerpt - Front Matter: "... and vice president, Sun Microsystems
[Surprise me!](#) See a random page in this book.

Searching in Solr: Highlighting & Faceting

www.linguee.de/englisch-deutsch/uebersetzung/lucene+search+engine.html

High School Exchange Year
Verbringe ein Schuljahr oder -halbjahr im Ausland und erlebe die Zeit deines Lebens!
GRATIS BROSCHÜRE

Ober Linguee | Linguee in English | Mitmachen | Einloggen | Apps | Werbung | Feedback | Hilfe

Deutsch ↔ Englisch

Lucene search engine

Suchen

Redaktionelles Wörterbuch

Kein exakter Treffer.

Nicht exakte Treffer:

- search engine** Substantiv
 - Suchmaschine *f* (Computer)
 - Suchroboter *m* (Computer)
- engine** Substantiv
 - Motor *m* · Triebwerk *nt* · Maschine *f* · Lokomotive *f* · Lok *f* · Kraftmaschine *n*
- v-engine** Substantiv
 - V-Motor *m* (Autos) (Technik)
- V-engine** Substantiv
 - V-Motor *m*
- search** Substantiv
 - Suche *f* · Recherche *f* · Durchsuchung *f* · Suchaktion *f*

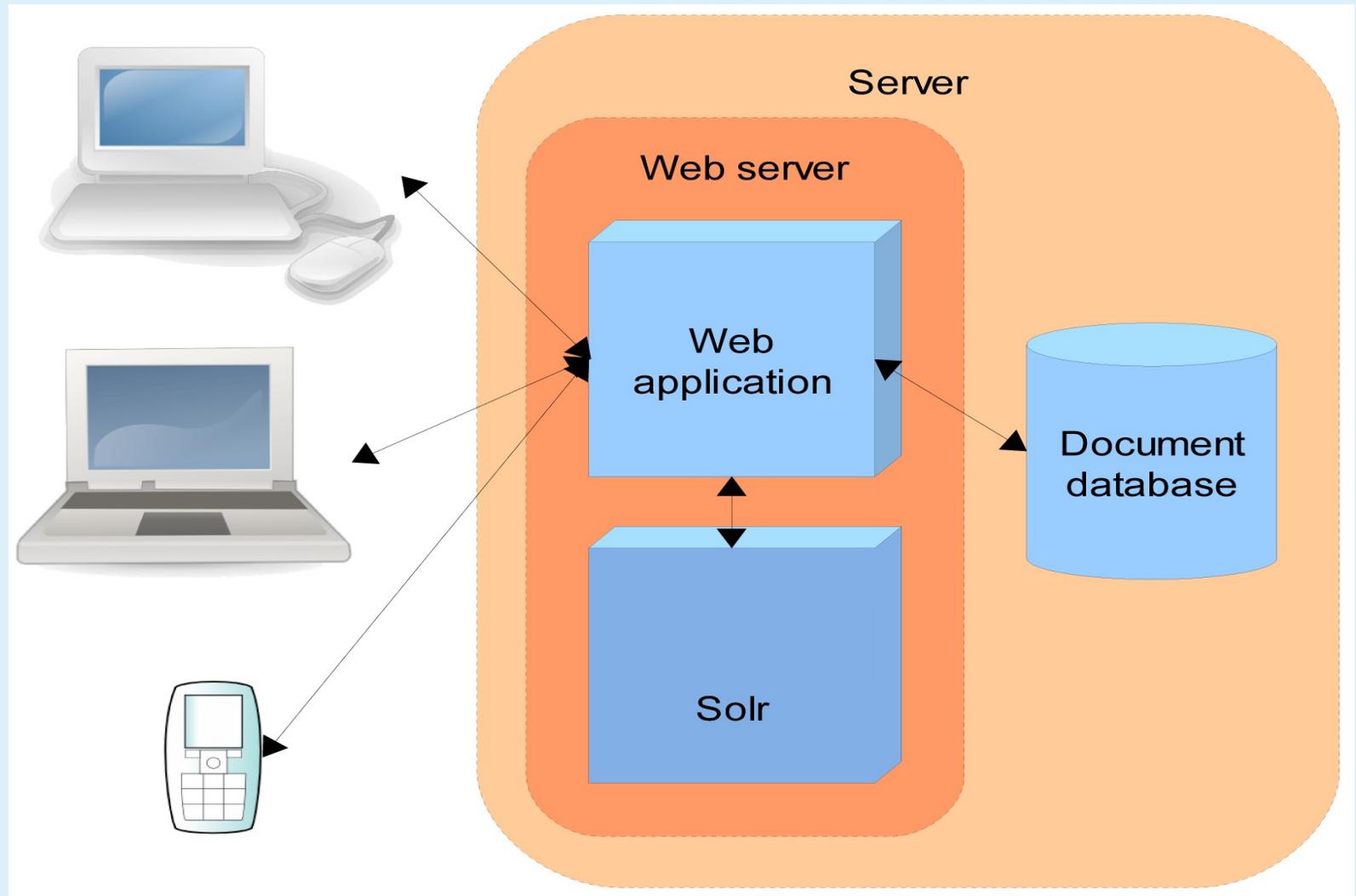
Übersetzungsbeispiele aus fremden Quellen für 'Lucene search engine':

Englisch	Deutsch
Ensures the easy retrieval of documents via the Lucene search engine which is even able to search and find documents within PDF files. <small>↳ finanzinformatik.ch</small>	? Sicherstellen der leichten Wiederauffindbarkeit von Dokumenten über die Lucene Suchmaschine, die selbst innerhalb von PDF Dateien sucht und findet . <small>↳ finanzinformatik.ch</small>
TET connector for the Lucene Search Engine <small>↳ pdflib.com</small>	TET-Konnektor für die Suchmaschine Lucene <small>↳ pdflib.com</small>
Net, which is port of the Java Lucene search engine to the C# and . <small>↳ company.hulbee.com</small>	Net, welches die Java Lucene Suchmaschine verbindet mit C# und . <small>↳ download.hulbee.com</small>
Also manage search indexes for the integrated Lucene search engine , define scheduled tasks to perform, e.g. to update the search indexes regularly every night. <small>↳ alkacon.com</small>	Außerdem können Sie Suchindexe für die integrierte Lucene-Suchmaschine verwalten oder geplante Tasks für die Durchführung definieren, z.B. die Aktualisierung der [...] <small>↳ alkacon.com</small>
It is based on the Zend implementation of Lucene search engine , and allows to browse the content area. <small>↳ snm-portal.com</small>	Sie setzt auf die Zend-Implementierung der Suche Engine Lucene auf, und ermöglicht auch den Content-Bereich zu durchsuchen . <small>↳ snm-portal.de</small>

Searching in Solr

- Searching in Solr can be done by:
 - Sending HTTP Get or Post requests
 - <http://localhost:8983/solr/select?q=dell>
 - The Query Form provided in the Web Admin
- Sorting:
 - Solr provides a simple method to sort on 1 or more indexed fields.
 - Use the „sort“ parameter:
 - ...?q=lcd&sort=price asc
- Highlighting:
 - ...?q=lcd&fl=name,price&hl=true&hl.fl=name,price

Solr's Use Case scenario



<https://cwiki.apache.org/confluence/display/solr/A+quick+overview>

Solr's Use Case Scenario

- Solr runs alongside another application in web serve, eg. an online store application.
- Solr makes it easy to add capability to search through, eg the online store through the following steps:
 - Define schema:
 - The schema tells Solr about the contents of documents it will be indexing:
 - The schema would define fields for: product name, description, price, manufacturer, etc.
 - Deploy Solr to your application server
 - Feed Solr the documents for which your users will search
 - Expose search functionality in your application

Solr Configuration

- Solr is configured using 3 main files:
 1. solr.xml:
 - Specifying configuration options for Solr core
 - Allowing to configure multiple cores
 2. solrconfig.xml:
 - controlling high-level behaviour
 - defining Solr's behaviour as it indexes content and responds to queries
 - Being able to specify an alternate location for the data directory
 - an example of solrconfig.xml can be found in Solr Administration UI, tab Config.

Solr Configuration

- Solr is configured using 3 main files:
 1. solrconfig.xml
 2. core.properties
 3. schema.xml:
 - Describing the documents indexed by Solr.
 - Defining a document as a collection of fields

Solr Configuration: solr.xml

- The default format → solr/example/solr/solr.xml
- Solr cores are configured by placing a file name *core.properties* in subdirectory under solr.home.
 - Cores maybe anywhere in the tree with an exception that they may not be defined under the existing core.

This is not allowed:

```
./cores/collection1/core.properties
```

```
./cores/colection1/coremore/collection2/core.properties
```

but this is legal/allowed:

```
./cores/somecores/collection1/core.properties
```

```
./cores/somecores/collection2/core.properties
```

- A minimal *core.properties* file looks like this:
name=collection1

Solr Configuration: solr.xml

- Solr.xml parameters:
 - The <solr> element:
 - The root element of solr.xml
 - There are no attribute that can be specified in the <solr>
 - Nodes: adminHandler, collectionsHandler, infoHandler, coreLoadThreads, etc (see cwiki.apache.org for node functions)
 - <solrcloud>: defines several paremeters that relate to solrCloud.
 - <logging>: defines classes to use for logging
 - <logging><watcher>: defines the size & threshold of log events
 - <shardHandlerFactory>: costumize share handlers defined in solr.xml

Solr Configuration: solr.xml

- The `core.properties` file:
 - Is a simple java properties where each line is a key=value pair
 - Use hash(#) or bang (!) characters to specify comment-to-end-of-line.
 - The recognized properties:
 - name → specifying the name of the SolrCore
 - config → specifying the configuration file name for a given core, default is `solrconfig.xml`.
 - Schema → specifying schema file name for a given core, default is `schema.xml`.
 - Datadir → specifying core's data directory as a path relative to the instance dir
 - Properties → specifying the name of properties file for this core. The value can be an absolute pathname to the value of `instanceDir`.

Configuring Solrconfig.xml

- The solrconfig.xml file is found in solr/conf directory
- In solrconfig.xml, the important features that need to configure are:
 - Request handler
 - Listeners (processes that listen for particular query-related events).
 - The Request Dispatcher for managing HTTP communications
 - The Admin Web interface
 - Parameters related to replication and duplication

Configuring Solrconfig.xml

- Request Handler:
 - Processes requests coming to Solr.
 - The requests might take in the form of queries or index updates.
 - Every request handler is defined with a name and a class.
 - The name of the request handler is referenced with the request to solr, eg. If '/select' is appended to the end, then a query can be made:

`http://localhost:8983/solr/collection1/select?q=solr`

- The primary request handler defined is SearchHandlers.
- The default solrconfig.xml for request handler looks like:

```
<requestHandler name="/select" class="solr.SearchHandler">
  <lst name="defaults">
    <str name="echoParams">explicit</str>
    <int name="rows">10</int>
    <str name="df">text</str>
  </lst>
</requestHandler>
```

Configuring Solrconfig.xml

- Request Handler:
 - The default example defines the following parameters:
 - rows → how many search results to return, eg. 10 rows
 - df → the default field to search is 'text' field
 - EchoParams → the parameters defined in the query should be returned when debug information is returned.
 - Other options for SearchHandler besides defaults:
 - appends: allows definition of parameters that are added to user query, eg. We define fq for filter query

```
<lst name="append">  
  <str name="fq">inStock:true</str>  
</lst>
```

Configuring Solrconfig.xml

- Request Handler:
 - Other options for SearchHandler besides defaults:
 - Invariants: allows definition of parameters that can't be overridden by a client.
 - The values defined in 'invariants' is always used regardless of the values specified by user, client in 'defaults' or in 'appends', eg

```
<lst name="invariants">
```

```
  <str name="facet.field">cat</str>
```

```
  <str name="facet.query">price:[* to 500]</str>
```

```
</lst>
```

Configuring Schema.xml

- Schema.xml is usually the first file to configure.
- The schema declares:
 - What kind of fields there are
 - Which fields should be used as unique/primary key
 - Which fields are required
 - How to index and search each field
- The most important tags in schema.xml are:
 - `<fieldtypes>` : Specifying and defining all types of fields
 - `<field>` : Defining your document structures

Configuring Schema.xml

```
<schema name="example" version="1.5">
  <field name="content" type="text_general" indexed="false" stored="true"
    multiValued="true"/>
  <field name="text" type="text_general" indexed="true" stored="false"
    multiValued="true"/>
  <fieldType name="text_general" class="solr.TextField" positionIncrementGap="100">
    <analyzer type="index">
      <tokenizer class="solr.StandardTokenizerFactory"/>
      <filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt" />
      <filter class="solr.LowerCaseFilterFactory"/>
    </analyzer>
    <analyzer type="query">
      <tokenizer class="solr.StandardTokenizerFactory"/>
      <filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt" />
      <filter class="solr.LowerCaseFilterFactory"/>
    </analyzer>
  </fieldType>
</schema>
```

References

- Smiley, D. & Pugh, E. (2011). Apache Solr 3 Enterprise Search Server. Birmingham: Packt Publishing.
- Solr Wiki: <http://wiki.apache.org/solr/>
- Apache solr Tutorial: <https://lucene.apache.org/solr/tutorial.html>