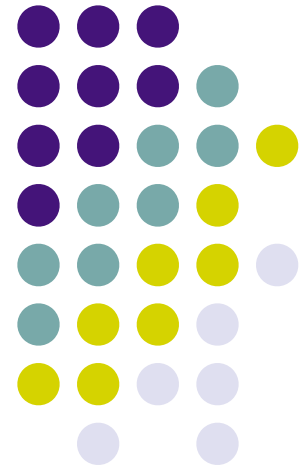


Better Search with Apache Lucene and Solr

Grant Ingersoll

November 19, 2007

Triangle Java User's Group



Intro



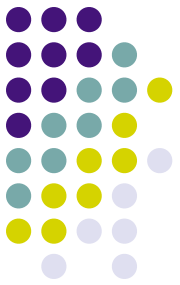
- Background
- Search Concepts
- Lucene
 - Indexing
 - Searching
 - Analysis
 - Miscellaneous
- Solr
 - Lucene on 'roids
 - Solr Setup
 - Indexing/Searching
 - Advanced Solr
- Lucene v. Solr

Background



- Lucene
 - Created by Doug Cutting in 1999 as a Source Forge project
 - Donated to the Apache Software Foundation (ASF) in 2001
- Solr
 - Created by Yonik Seeley while at CNET
 - Based on Lucene
 - Donated to ASF in 2006
- Nutch, Hadoop, Tika

Users

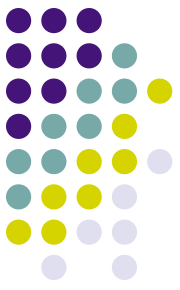


- Lucene
 - IBM Omnifind Y! Edition
 - Technorati
 - Wikipedia
 - Internet Archive
 - LinkedIn
- Solr
 - Netflix
 - CNET
 - Smithsonian
 - AOL:sports and music
- Many others



Search Concepts

- User inputs one or more keywords along with some operators and expects to get back a ranked list of documents relevant to the keywords
- User sorts through the documents, reading/using the most relevant
 - User's relevant docs does not always equal to the search engines



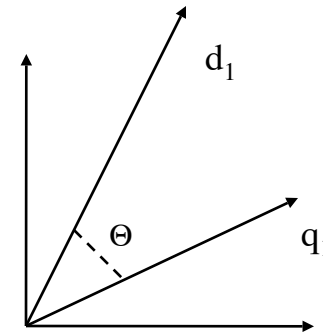
Search Concepts

- Several approaches developed over the years:
 - Boolean Model
 - Vector Space Model
 - Probabilistic Model
 - Language Modeling
 - Latent Semantic Indexing
- Vector Space Model is probably the most common and is generally fast
- Search is not a solved problem, despite Google's success!

Vector Space Model



- Goal: Identify documents that are similar to input query
- Represent each word with a weight w
- The words in the document and the query each define a Vector in an n -dimensional space
- Common weighting scheme is called **TF-IDF**
 - TF = Term Frequency
 - IDF = Inverse Document Freq.
- Intuition behind TF-IDF:
 - A term that frequently occurs in a few documents relative to the collection is more important than one that occurs in a lot of documents
- $\text{Sim}(q_1, d_1) = \cos \Theta$



$$\mathbf{d}_j = \langle w_{1,j}, w_{2,j}, \dots, w_{n,j} \rangle$$

$$\mathbf{q} = \langle w_{1,q}, w_{2,q}, \dots, w_{n,q} \rangle$$

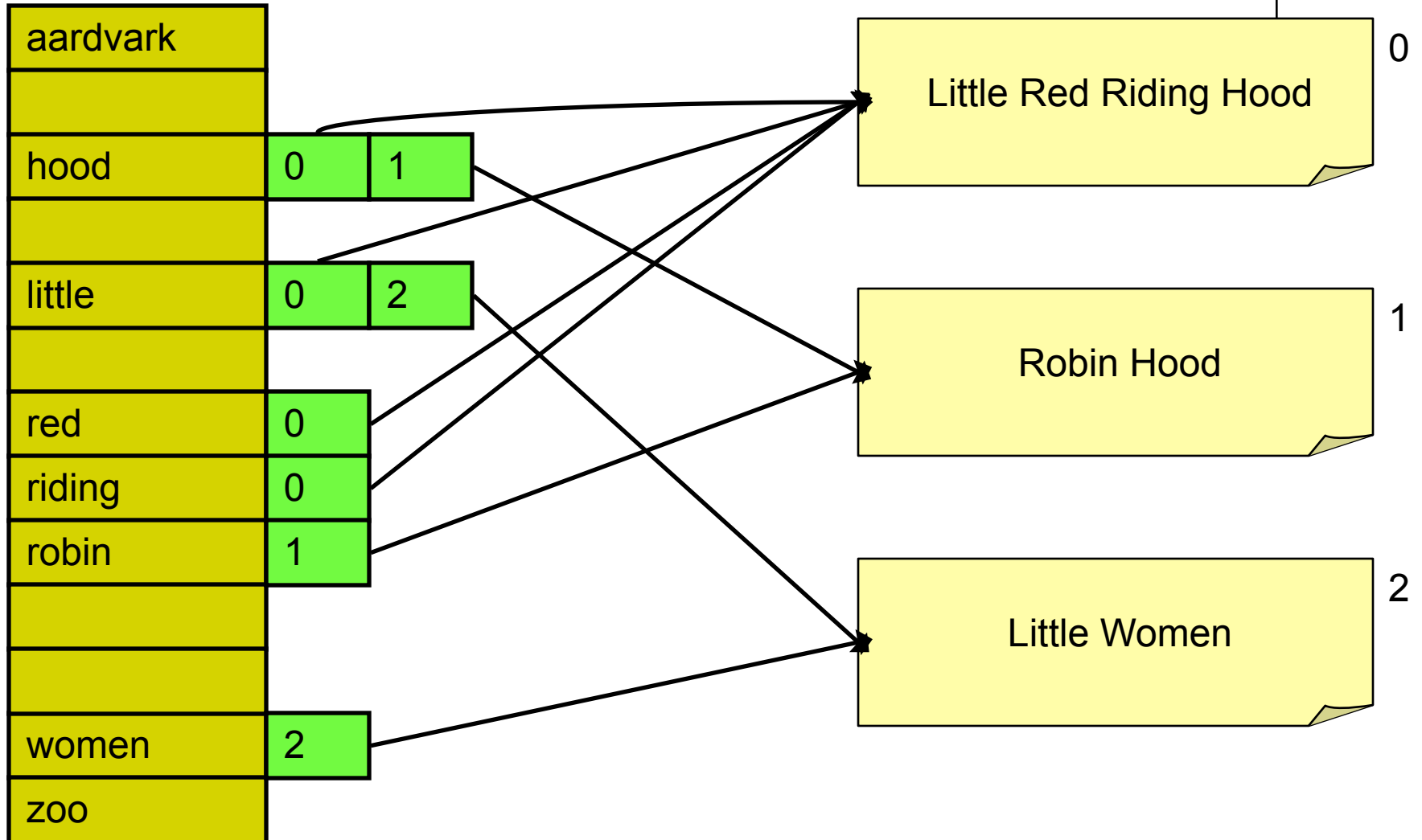
w = weight assigned to term

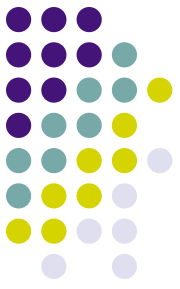


Making Content Searchable

- Search engines generally:
 - Extract Tokens from Content
 - Optionally transform said tokens depending on needs
 - Stemming
 - Expand with synonyms (usually done at query time)
 - Remove token (stopword)
 - Add metadata
 - Store tokens and related metadata (position, etc.) in a data structured optimized for searching
 - Called an Inverted Index

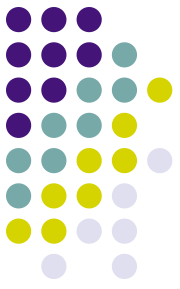
Inverted Index





On to Lucene

- Provides modified Vector Space Model implementation of search
 - Boolean + VSM
- Written in Java, but has been ported to many languages:
 - C/C++
 - Ruby
 - Python
 - .NET



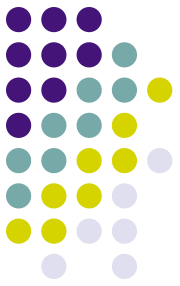
Lucene is

- NOT a crawler
 - See Nutch
- NOT an application
 - See PoweredBy on the Wiki
- NOT a library for doing Google PageRank or other link analysis algorithms
 - See Nutch and Hadoop
- A library for enabling text based search

Vocab



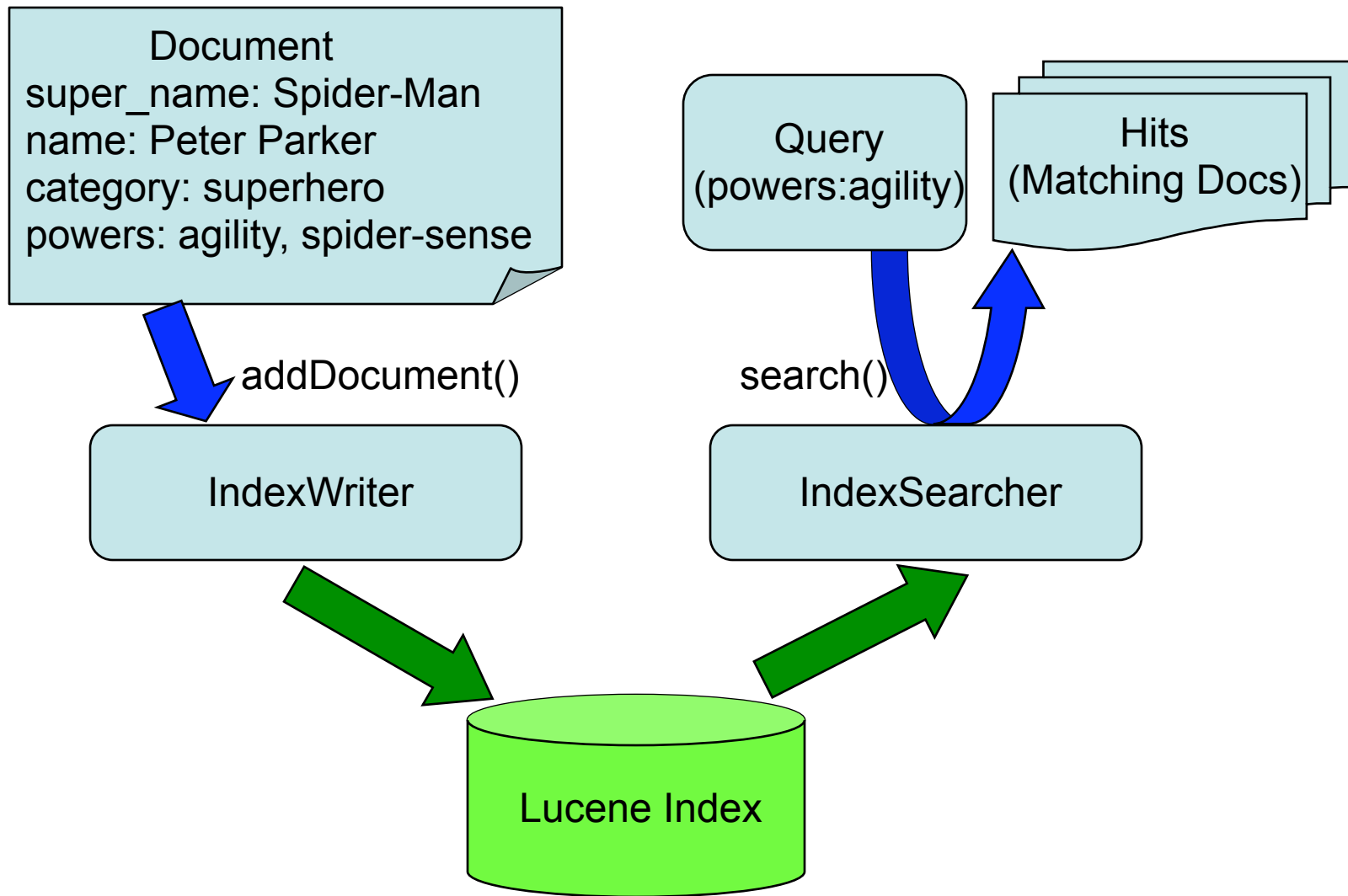
- A Lucene Index is a collection of Documents
- A Document is a collection of Fields
- A Field is content along with metadata describing the content
- Field content can have several attributes
 - Tokenized - Analyze the content, extracting Tokens and adding them to the inverted index
 - Stored - Keep the content in a storage data structure for use by application

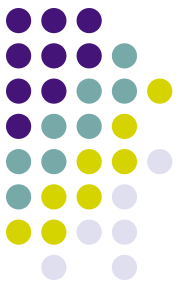


Getting Started

- Download Lucene from <http://lucene.apache.org/java>
- Indexing Side:
 - Write code to add Documents to index
- Search Side
 - Write code to transform user query into Lucene Query instances
 - Submit Query to Lucene to Search
 - Display Results

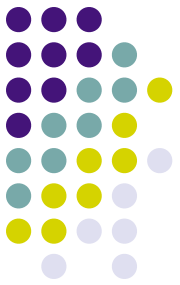
Basic Application





Indexing

- Process of preparing and adding text to Lucene
 - Optimized for searching
- Key Point: Lucene only indexes Strings
 - What does this mean?
 - Lucene doesn't care about XML, Word, PDF, etc.
 - There are many good open source extractors available
 - It's our job to convert whatever file format we have into something Lucene can use



How to Index

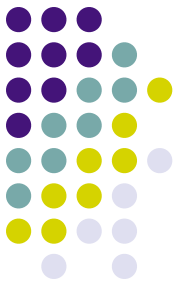
- Look at Sample Code in QuickExampleTest.java
- Available at <http://www.lucenebootcamp.com> under the Subversion repository
 - Other resources available there as well

Searching



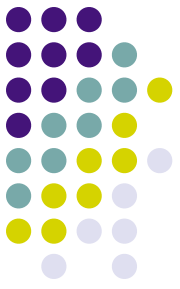
- Lucene Query Parser converts strings into Java objects that can be used for searching
 - See <http://lucene.apache.org/java/docs/queryparsersyntax.html>
- Query objects can also be constructed programmatically
- Native support for many types of queries
 - Keyword
 - Phrase
 - Wildcard
 - Many more

Searching



- Look again at QuickExampleTest.java for examples of how to search

Analysis



- Analysis is the process of converting raw text into indexable Tokens
- In Lucene, this is done by the *Analyzer*, *Tokenizer* and *TokenFilter* classes
- The *Tokenizer* is responsible for chunking the input into Tokens
- *TokenFilters* can further modify the Tokens produced by the *Tokenizer*, including:
 - Removing it
 - Stemming
 - Other



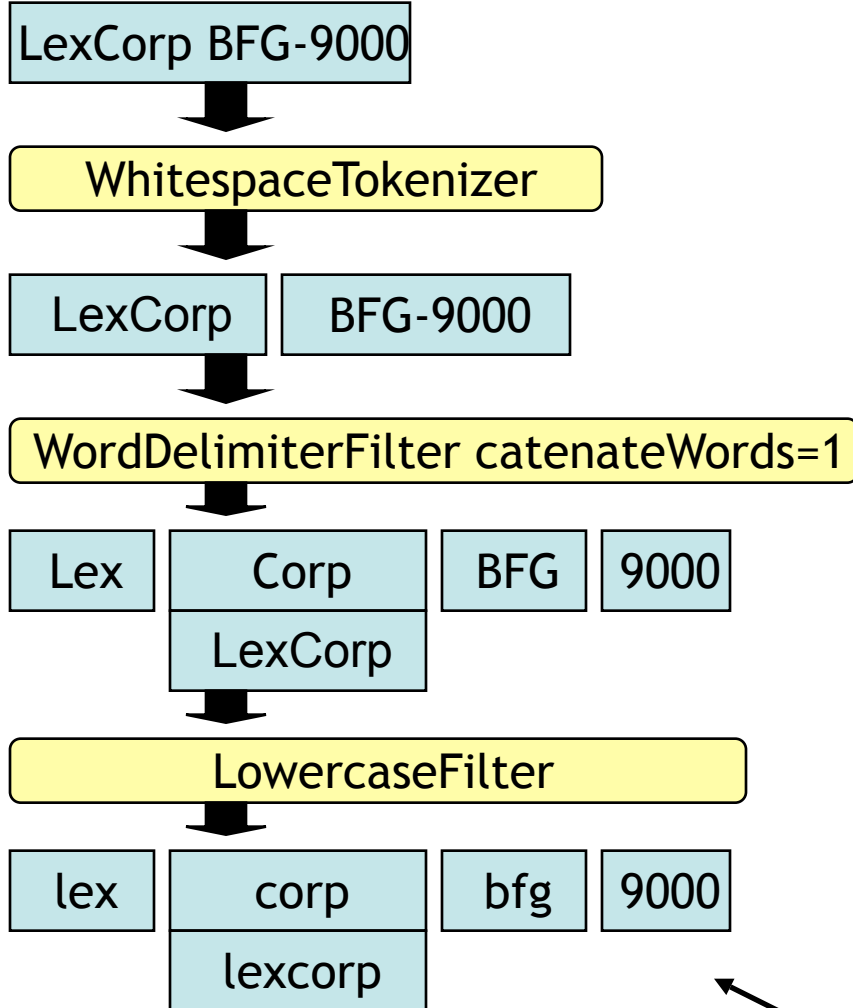
Analysis

- Lucene provides many Analyzers out of the box
 - StandardAnalyzer
 - WhitespaceAnalyzer
 - Many others, including support for other languages
- Easy to add your own
- Analysis is done on both the content to be indexed and the query
 - Same type of Analyzer should be used

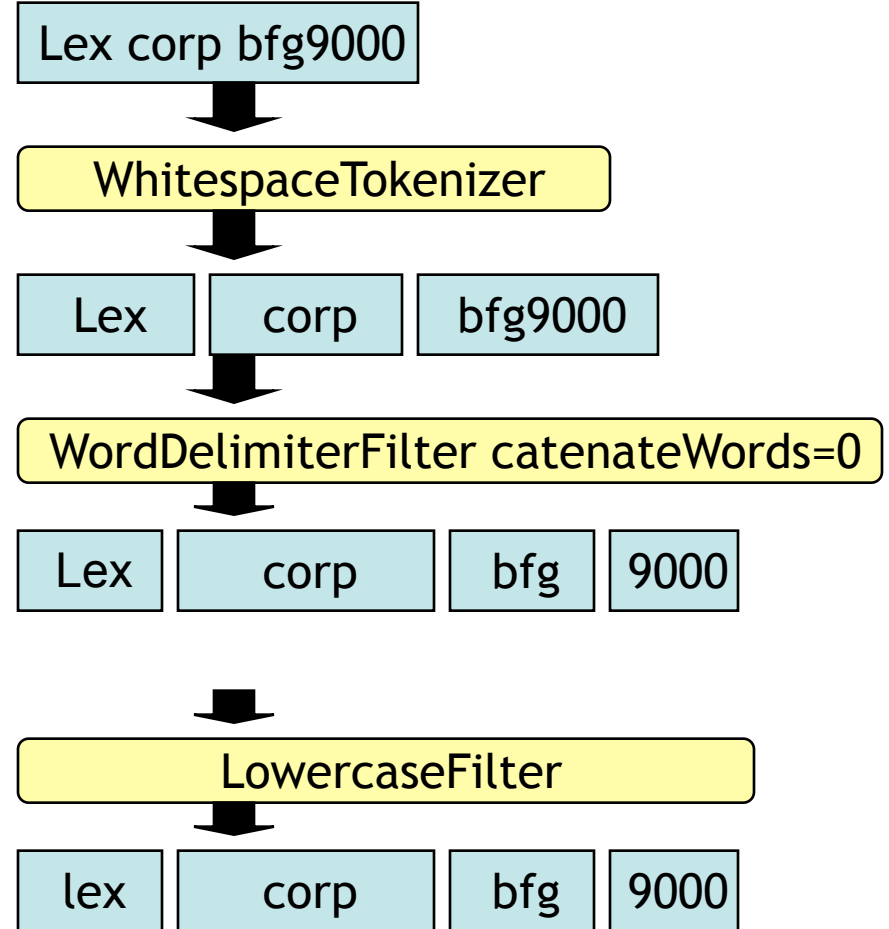
Analysis and Search Relevancy



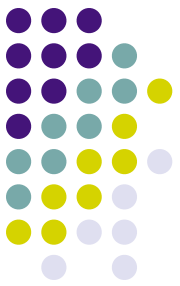
Document Indexing Analysis



Query Analysis

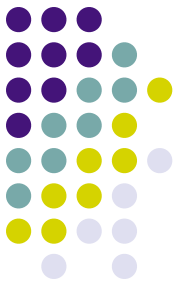


← A Match! →



Miscellaneous

- Lucene comes with many contributed modules that are not part of the core, but solve common problems:
 - Located in contrib directory in the distribution
- Popular contribs:
 - Highlighter
 - Spell Checking
 - Analyzers
 - More Like This (finds similar pages)
- Luke:
 - <http://www.getopt.org/luke/>



Luke is your friend

Luke - Lucene Index Toolbox, v 0.7 (2007-02-20)

File Tools Settings Help

Overview Documents Search Files Plugins

Index name: /Users/grantingersoll/projects/apachecon/europe2007/index
Number of fields: 3
Number of documents: 21578
Number of terms: 103205
Has deletions?: No
Index version: 1172436418142
Last modified: Sun Feb 25 15:51:12 EST 2007
Directory implementation: org.apache.lucene.store.FSDirectory

Re-open
Close

Select fields from the list below, and press button to view top terms in these fields. No selection means all fields.

Available Fields: <body> <date> <title>

Show top terms >>

Number of top terms: 50

Hint: use Shift-Click to select ranges, or Ctrl-Click to select multiple fields (or unselect all).

Top ranking terms. (Right-click for more options)

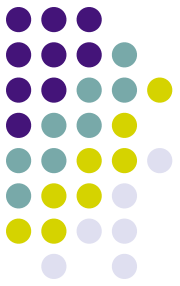
No	Rank	Field	Text
1	19043	<body>	3
2	18883	<body>	reuter
3	15345	<body>	said
4	8291	<body>	mln
5	7960	<body>	its
6	7651	<body>	dtrs
7	7590	<body>	from
8	6002	<body>	has
9	5961	<body>	pct
10	5846	<body>	year
11	5479	<body>	company
12	5045	<body>	which
13	4724	<body>	inc
14	4416	<body>	corp
15	4221	<body>	would
16	3996	<body>	were
17	3939	<body>	one
18	3938	<body>	us
19	3877	<body>	have

Index name: /Users/grantingersoll/projects/apachecon/europe2007/index

Solr



"Solr is an open source enterprise search server based on the Lucene Java search library, with XML/HTTP APIs, caching, replication, and a web administration interface."



Why Solr?

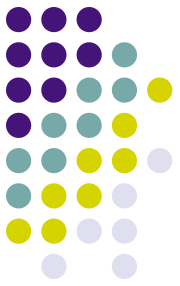
- Uses many Lucene best practices
- Makes setup and configuration a snap
- Easy to extend
- Supports clients in:
 - HTTP
 - Java
 - Ruby
 - JSON
 - PHP
 - Python



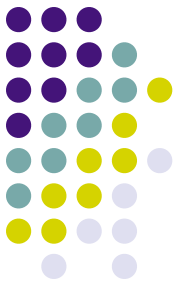
Getting Started

- <http://lucene.apache.org/solr/tutorial.html>
- In the Solr directory:
 - cd example
 - java -jar start.jar
- Indexing:
 - cd exampledocs
 - java -jar post.jar *.xml
- Browse to
 - <http://localhost:8983/solr>

Solr Setup



- **schema.xml**
 - Describe your data and how it is processed
 - Adds semantics to Lucene Fields to handle types other than Strings (int, double, long, custom, etc.)
- **solrconfig.xml**
 - Describe how clients interact with your data
 - Specifies:
 - Data Location
 - Performance Options
 - Types of actions allowed



Indexing in Solr

- Send XML add commands over HTTP
 - Other options exist as well, but XML is most common

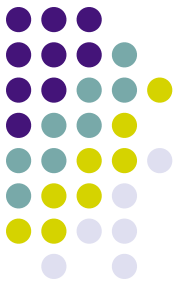
```
<add><doc>
```

```
<field name="id">canes</field>
```

```
<field name="name">Carolina
```

```
  Hurricanes</field> </doc></add>
```

- Can also delete, update documents



Searching

- Send HTTP Get or Post requests
- Query parameters specify options
- <http://localhost:8983/solr/select/?q=iPod>
- Try <http://localhost:8983/solr/admin/form.jsp>
- Solr supports:
 - Faceting
 - Highlighting
 - More Like This
 - Spell Checking
 - Other

Faceting



amazon.com Hello, Grant Ingersoll. We have [recommendations](#) for you.

Grant's Amazon.com Today's Deals Gift Cards

Shop All Departments

Search Amazon.com Wii

Category

Any Category

- Books (16,456)
- Video Games (795)
- Electronics (419)
- Everything Else (166)
- Apparel (80)
- Toys & Games (65)
- Sports & Outdoors (24)
- Home & Garden (23)
- Health & Personal Care (19)
- Home Improvement (15)
- Software (9)
- Automotive (9)

"Wii"

Related Searches: [nintendo wii](#), [ps3](#), [wii game](#)

Showing Top Results



Explore over 280 products tagged with Wii. See what the community is talking about.

1.

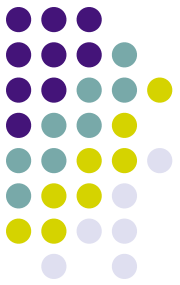


Wii by Nintendo (93 Used & new from \$149.99)

★★★★☆ (753)

694 customers tagged this product

Video Games: See all 1,234 results



Highlighting

- From Digg.com

2 diggs

[Full-Text Search for Database Using Lucene Search Engine](#)


blog.scalingweb.com — Shows you how to create full-text index, thus making your database : which scales you'll have to do it yourself. This post is based on real experience of optimiz

[digg it](#) [0 Comments](#) [Share](#) [Bury](#)  [Shmo](#) — submitted 13 days ago

1 digg

[Visvo search startup](#)

yousefourabi.com — Why does search startup matter? Three reasons: 1) Each search result using open Source Search Technology Nutch, and Hadoop both of the Lucene project 3) 1 (Tech Industry News)

[digg it](#) [1 Comment](#) [Share](#) [Bury](#)  [yourabi](#) — submitted 13 days ago

Advanced Solr

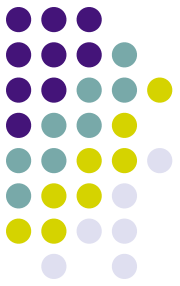


- Replication
 - Use case:
 - Updates can wait (minutes instead of seconds)
 - High query volume
 - Load-balanced environment
- Caching
 - Solr provides intelligent caching of objects such as:
 - Query Filters
 - Documents
 - Search Results
 - User Defined



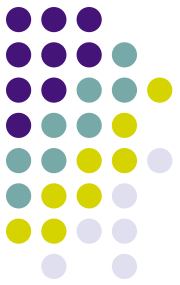
Advanced Solr

- In Lucene, an IndexReader (used for searching by the IndexSearcher) represents a point-in-time view of the index
 - Updates occurring after opening the IndexReader are not available for searching
 - Applications must decide when to make changes available for search
- Solr can open up new IndexSearcher instances and warm them before bringing them online



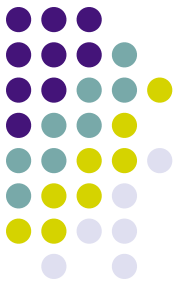
Lucene v. Solr

- Lucene and Solr share many common features and terminologies
- Both have friendly Apache Software Foundation license
- Both have an extensive, knowledgeable community
- Both are fast, scalable, thread-safe, etc.
- Both have good documentation
- Both are battle-tested



Lucene v. Solr

- Lucene
 - Embedded/
lightweight
 - No Container
 - Want low-level control over all aspects of process
 - Thick clients?
 - Distributed?
 - Need to use features not available in Solr
 - JDK 1.4
- Solr
 - Server-side
 - HTTP as lingua-franca
 - Want ease of setup and configuration
 - Non-Java clients
 - Replication/
Caching OOTB
 - JDK 1.5



Resources

- <http://lucene.apache.org>
 - /solr
 - /java
- java-user@lucene.apache.org
- solr-user@lucene.apache.org
- *Lucene In Action* by Erik Hatcher and Otis Gospodnetić
- Search Smarter with Apache Solr
 - <http://www.ibm.com/developerworks/java/library/j-solr1/>
 - <http://www.ibm.com/developerworks/java/library/j-solr2/>



Resources

- Other presentations:
 - <http://people.apache.org/~gsingers>
 - <http://people.apache.org/~yonik>
 - <http://people.apache.org/~hossman>
- <http://www.lucenebootcamp.com>
- My email:
 - User: trainer
 - Domain: lucenebootcamp.com