

# Information Retrieval – begleitendes Tutorium IV

Thomas Schmidt

# Rückblick

- Solr – Einführung und Installation
- Schema.xml
- Felder
- Analyzer
- Admin – UI
- Solritas – kurzer Einblick



# Heute

- Indexierung verschiedener Dateiformate in Solr
- DataImportHandler – Tutorial
- Solrconfig.xml (RequestHandler, SearchHandler, SearchComponents etc.)
- Literatur

# Installation (nochmal...)

- Solr-4.8.0.zip runterladen (Link im Grips oder googeln) und entpacken
- über Ausführen – cmd: Konsole öffnen
- Navigation zum solr-4.8.0 – Ordner über cd-Befehl
- zu example-Ordner navigieren, also cd example
- „java -jar start.jar“ eingeben
- Solr nun erreichbar unter: <http://localhost:8983/solr>

# Indexierung – verschiedene Dateiformate → verschiedene Lösungen

XML – formatiert

JSON

CSV

PDF

HTML

Office-Dateien

Datenbanken

RSS

XML ...usw.



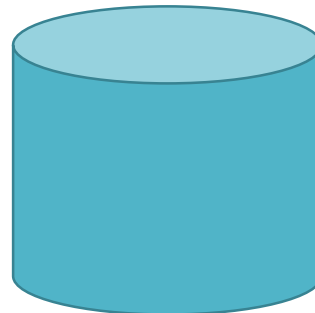
UpdateRequest-  
Handler



ExtractingRequest-  
Handler



DataImport-  
Handler



Solr –  
Index

# UpdateRequestHandler

- wird in solrconfig.xml registriert (später mehr dazu)
- Post.jar schickt Dokumente an den UpdateHandler /update

```
<requestHandler name="/update/json" class="solr.JsonUpdateRequestHandler">  
  <lst name="defaults">  
    <str name="stream.contentType">application/json</str>  
  </lst>  
</requestHandler>  
<requestHandler name="/update/csv" class="solr.CSVRequestHandler">  
  <lst name="defaults">  
    <str name="stream.contentType">application/csv</str>  
  </lst>  
</requestHandler>
```

# XML – im Feldformat von Solr

```
somedocs.xml  x
1  <add>
2  <doc>
3      <field name="id">1</field>
4      <field name="author">Florian Meier</field>
5  </doc>
6
7  <doc>
8      <field name="id">2</field>
9      <field name="author">Thomas Schmidt</field>
10 </doc>
11 </add>
```



Felder müssen in  
schema.xml definiert sein

```
java -jar post.jar somedocs.xml
(oder um alle Dateien zu indexieren:)
java -jar post.jar *.xml
```

# XML in das richtige Format bringen - XSLT

## XML Code:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Edited by XMLSpy -->
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  <cd>
    <title>Hide your heart</title>
    <artist>Bonnie Tyler</artist>
    <country>UK</country>
    <company>CBS Records</company>
    <price>9.90</price>
    <year>1988</year>
  </cd>
  <cd>
    <title>Greatest Hits</title>
    <artist>Dolly Parton</artist>
    <country>USA</country>
    <company>RCA</company>
    <price>9.90</price>
```



## Me »

## XSLT Code:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Edited by XMLSpy® -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

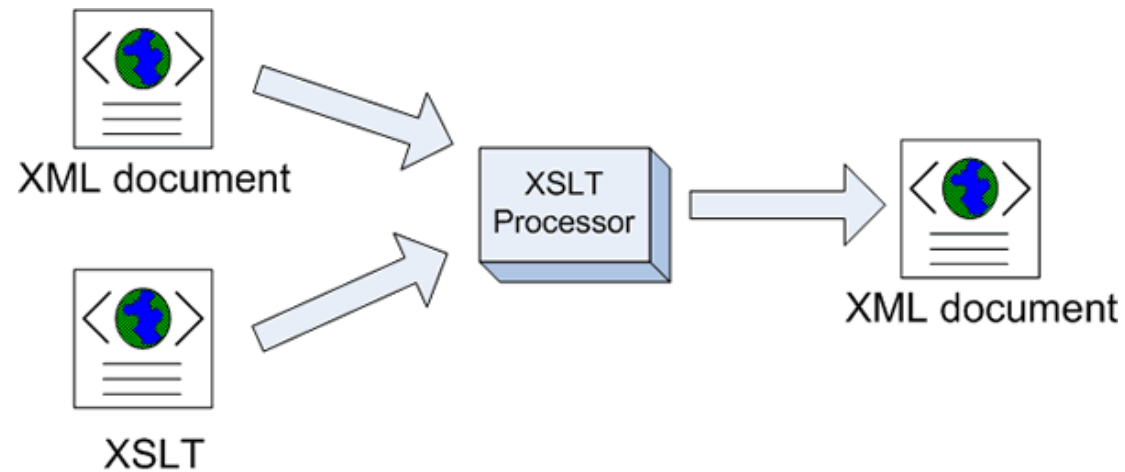
  <xsl:template match="/">
    <html>
      <body>
        <h2>My CD Collection</h2>
        <table border="1">
          <tr bgcolor="#9acd32">
            <th style="text-align:left">Title</th>
            <th style="text-align:left">Artist</th>
          </tr>
          <xsl:for-each select="catalog/cd">
            <tr>
              <td><xsl:value-of select="title"/></td>
              <td><xsl:value-of select="artist"/></td>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```



# Mehr Infos dazu...

<http://www.w3schools.com/xsl/default.asp>

<http://www.w3schools.com/XPath/>



# JSON

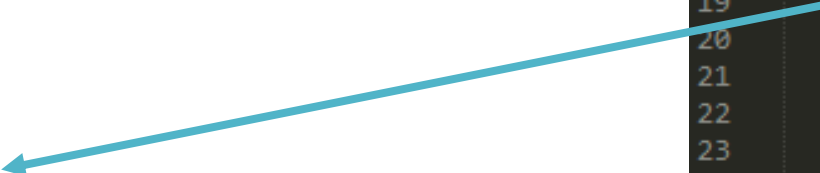
über /update/json - Handler

java -Dtype=application/json -jar post.jar somejson.json  
(oder um alle Dateien zu indexieren:)

java -Dtype=application/json -jar post.jar \*.json

Namen werden als  
Felder interpretiert,  
Werte als Inhalt des  
Feldes

```
1  [
2    {
3      "id" : "978-0641723445",
4      "cat" : ["book","hardcover"],
5      "name" : "The Lightning Thief",
6      "author" : "Rick Riordan",
7      "series_t" : "Percy Jackson and the Olympians",
8      "sequence_i" : 1,
9      "genre_s" : "fantasy",
10     "inStock" : true,
11     "price" : 12.50,
12     "pages_i" : 384
13   },
14   {
15     "id" : "978-1423103349",
16     "cat" : ["book","paperback"],
17     "name" : "The Sea of Monsters",
18     "author" : "Rick Riordan",
19     "series_t" : "Percy Jackson and the Olympians",
20     "sequence_i" : 2,
21     "genre_s" : "fantasy",
22     "inStock" : true,
23     "price" : 6.49,
24     "pages_i" : 304
25   },
26   {
27     "id" : "978-1857995879",
28     "cat" : ["book","paperback"],
29     "name" : "Sophie's World : The Greek Philosophers",
30     "author" : "Jostein Gaarder",
31     "sequence_i" : 1,
32     "genre_s" : "fantasy",
33   }
34 ]
```



# CSV

```
id,cat,name,price,inStock,author,series_t,sequence_i,genre_s
0553573403,book,A Game of Thrones,7.99,true,George R.R. Martin,"A Song of Ice and Fire",1,fantasy
0553579908,book,A Clash of Kings,7.99,true,George R.R. Martin,"A Song of Ice and Fire",2,fantasy
055357342X,book,A Storm of Swords,7.99,true,George R.R. Martin,"A Song of Ice and Fire",3,fantasy
0553293354,book,Foundation,7.99,true,Isaac Asimov,Foundation Novels,1,scifi
0812521390,book,The Black Company,6.99,false,Glen Cook,The Chronicles of The Black Company,1,fantasy
0812550706,book,Ender's Game,6.99,true,Orson Scott Card,Ender,1,scifi
0441385532,book,Jheræg,7.95,false,Steven Brust,Vlad Taltos,1,fantasy
0380014300,book,Nine Princes In Amber,6.99,true,Roger Zelazny,the Chronicles of Amber,1,fantasy
0805080481,book,The Book of Three,5.99,true,Lloyd Alexander,The Chronicles of Prydain,1,fantasy
080508049X,book,The Black Cauldron,5.99,true,Lloyd Alexander,The Chronicles of Prydain,2,fantasy
```

erste Zeile (Spalten) werde als Feldnamen interpretiert, Zeilen als jeweiliger Inhalt

über /update/csv - Handler

```
java -Dtype=application/csv -jar post.jar somecsv.csv
```

(oder um alle Dateien zu indexieren:)

```
java -Dtype=application/csv -jar post.jar *.csv
```

# ExtractingRequestHandler

```
<requestHandler name="/update/extract"
  startup="lazy"
  class="solr.extraction.ExtractingRequestHandler" >
  <lst name="defaults">
    <str name="lowernames">true</str>
    <str name="uprefix">ignored_</str>

    <!-- capture link hrefs but ignore div attributes -->
    <str name="captureAttr">true</str>
    <str name="fmap.a">links</str>
    <str name="fmap.div">ignored_</str>
  </lst>
</requestHandler>
```

*"Solr's ExtractingRequestHandler uses Tika to allow users to upload binary files to Solr and have Solr extract text from it and then index it."*



.docx, .pdf, .html etc

- Metadaten wie Autor, Titel, Subject
- Kompletter Inhalt geht in ein content-Feld


# Exkurs: Apache Tika

- Toolkit, dass strukturierten Text und Metadaten erkennt und extrahiert
- basiert auf verschiedenen Parser – Libraries
- unterstützt zahlreiche Formate (pdf, html, multimedia und, und, und...)




# Beispiel: PDF


```
java -Durl=http://localhost:8983/solr/update/extract -Dparams=literal.id=pdfdoc1 -Dtype=application/pdf -jar post.jar example.pdf
```



Dateityp muss  
spezifiziert werden



Handler muss  
spezifiziert werden



ID sollte angegeben  
werden (Feldtypen und  
Inhalte können über  
Literale übergeben  
werden)

# Mehr dazu...

ExtractHandler mit Tika → sehr mächtig und manipulierbar (Feldboosts, Xpath, Mapping and Capture, etc.)

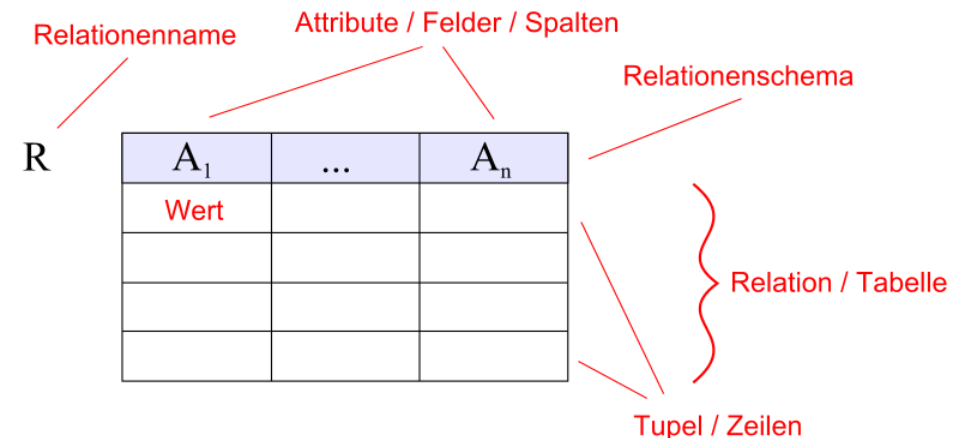
<http://wiki.apache.org/solr/ExtractingRequestHandler>

<http://tika.apache.org/>

<https://cwiki.apache.org/confluence/display/solr/Uploading+Data+with+Solr+Cell+using+Apache+Tika>

# DataImportHandler

- Daten aus relationalen Datenbanken auslesen (benötigen JDBC-Treiber)
- Solr-Dokumente über mehrere Spalten und Tabellen erstellen
- Full-Import
- Delta-Import (Aktualisierung von Daten)
- weitere Datentypen indexieren (xml, rss, usw.)
- Konfigurationsmöglichkeiten





# Beispiel: Indexierung von Datenbank

- Beispiele im Ordner: example/example-DIH/
- Beispiel für relationale Datenbank im Ordner: example/example-DIH/solr/db

```
<lib dir="../../../dist/" regex="solr-dataimporthandler-.*\.jar" />
```

```
<requestHandler name="/dataimport" class="org.apache.solr.handler.dataimport.DataImportHandler">  
  <lst name="defaults">  
    <str name="config">db-data-config.xml</str>  
  </lst>  
</requestHandler>
```



DataImportHandler in solrconfig.xml registrieren

→ Zugriff über Admin – UI möglich

(<http://localhost:8983/solr/dataimport>)

# SQL – Kenntnisse nötig!

<http://www.w3schools.com/sql/default.asp?PHPSESSID=300ae3404d5fa2612f238abeebb8869c>

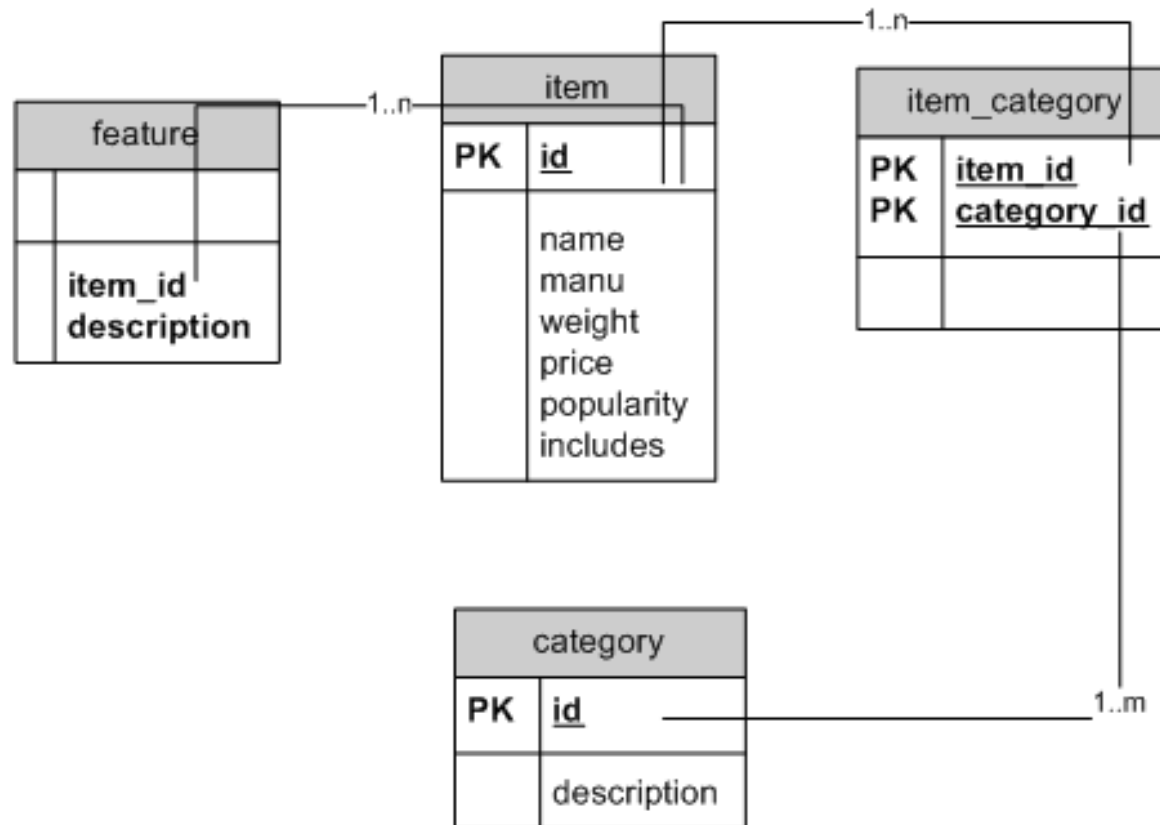
<http://sql.lernenhoch2.de/lernen/>

<http://www.schulserver.hessen.de/darmstadt/lichtenberg/SQLTutorial/>

viele Bücher → Regensburger Katalog



# Beispieldatenbank



# data-config.xml

```
<dataConfig>
<dataSource driver="org.hsqldb.jdbcDriver" url="jdbc:hsqldb:/temp/example/ex" user="sa" />
  <document name="products">
    <entity name="item" query="select * from item">
      <field column="ID" name="id" />
      <field column="NAME" name="name" />
      <field column="MANU" name="manu" />
      <field column="WEIGHT" name="weight" />
      <field column="PRICE" name="price" />
      <field column="POPULARITY" name="popularity" />
      <field column="INSTOCK" name="inStock" />
      <field column="INCLUDES" name="includes" />

      <entity name="feature" query="select description from feature where item_id='${item.ID}'">
        <field name="features" column="description" />
      </entity>
      <entity name="item_category" query="select CATEGORY_ID from item_category where item_id='${item.ID}'">
        <entity name="category" query="select description from category where id = '${item_category.CATEGORY_ID}'">
          <field column="description" name="cat" />
        </entity>
      </entity>
    </entity>
  </document>
</dataConfig>
```

# In der Realität – dataSource

```
<dataSource type="JdbcDataSource"  
            driver="com.mysql.jdbc.Driver"  
            url="jdbc:mysql://localhost/dbname"  
            user="db_username"  
            password="db_password"/>
```

# DIH - Konfiguration

- Datenquelle (dataSource)
- Entity (entity) → Root-Entity (zentrales Table/View), Sub-Entity (durch JOIN über Table von Root-Entity)
- mehrere Entities möglich → mehrere Datenbanken
- Transformer (transformer) → Inhalte verändern bevor diese indexiert werden
- Prozessoren (processor) → Daten extrahieren, verarbeiten (mittels transformers), indexieren

## Data-config.xml (anderes Beispiel)

```
<dataConfig>
  <dataSource type="FileDataSource" encoding="UTF-8" />
  <document>
    <entity name="page" processor="XPathEntityProcessor"
      stream="true" foreach="/mediawiki/page/" url="solr/wikipedia/solr_wikipedia/
      dewiki-latest-pages-articles.xml"
      transformer="RegexTransformer,DateFormatTransformer">
      <field column="id"      xpath="/mediawiki/page/id" />
      <field column="title"   xpath="/mediawiki/page/title" />
      <field column="revision" xpath="/mediawiki/page/revision/id" />
      <field column="user"    xpath="/mediawiki/page/revision/contributor/username"
      />
      <field column="userId"  xpath="/mediawiki/page/revision/contributor/id" />
      <field column="text"    xpath="/mediawiki/page/revision/text" />
      <field column="timestamp" xpath="/mediawiki/page/revision/timestamp"
        dateTimeFormat="yyyy-MM-dd'T'hh:mm:ss'Z'" />
      <field column="$skipDoc" regex="^#REDIRECT .*" replaceWith="true"
        sourceColName="text"/>
    </entity>
  </document>
</dataConfig>
```

# DIH – Full Import am Beispiel

Start der entsprechenden Solr-Instanz:

```
java -Dsolr.solr.home="./example-DIH/solr/" -jar start.jar
```

Full Import über Admin-UI oder URL-Eingabe:

<http://localhost:8983/solr/db/dataimport?command=full-import>





Dashboard

Logging

Core Admin

Java Properties

Thread Dump

db

Overview

Analysis

Config

Dataimport

Documents

Ping

Plugins / Stats

Query

Replication

● /dataimport

Command

full-import

☐ Verbose

☒ Clean

☒ Commit

☐ Optimize

☐ Debug

Entity

Start, Rows

0 10

Custom Parameters

key1=val1&key2=val2

Execute

Refresh Status

☐ Auto-Refresh Status

Last Update: 07:07:46

No information available (idle)

Raw Status-Output

Configuration

Debug-Mode Reload

Documentation

Issue Tracker

IRC Channel

Community forum

Solr Query Syntax

# Entity – Processors (Extraktion von Daten)

Name	Funktion
XPathEntityProcessor	Daten im XML-Format, die nicht Solr-konform sind
SQLEntityProcessor	Daten aus relationalen Datenbanken, z.B. SQL Default Entity Processor
TikaEntityProcessor	Ähnlich zu ExtractingRequestHandler PDF, Word, etc.
FileListEntityProcessor	Iteriert über Liste von Dateien, gibt sie jeweils an weitere Entity – Prozessors weiter
LineEntityProcessor	Liest den Inhalt einer Quelle zeilenweise ein
PlainTextEntityProcessor	Text wird unbehandelt in Feld plainText geschrieben

# Allgemeine Attribute für Entitys

Attributname	Funktion
datasource	Name der Datenquelle, muss vorher als Element deklariert sein
processor	Auswahl des Entity Processors
transformer	kommaseparierte Liste an Transformers
name	Name der Entity (required)
pk	Primary Key der Entity, der beim Delta-Import verwendet wird
onError	Fehlerbehandlung
preImportDeleteQuery	Query, die vor dem Import ausgeführt wird
postImportDeleteQuery	Query, die nach dem Import ausgeführt wird
rootEntity	zur Spezifizierung einer Wurzel-Entity

# Attribute für Entitys

→ jeder EntityProcessor besitzt zahlreiche individuelle Attribute

<http://wiki.apache.org/solr/DataImportHandler#EntityProcessor>



# Transformers (Manipulation von Daten)

Name	Funktion
ClobTransformer	Erzeugung von Strings aus „Clobs“
DateFormatTransformer	wandelt Datumsangabe in ein anderes Format um
HTMLStripTransformer	befreit den Inhalt eines Feldes von HTML-Markup
LogTransformer	Informationen auf der Konsole oder in Dateien loggen
NumberFormatTransformer	erzeugt aus String Zahl, Prozentzahl, Währung oder einen Integer
RegexTransformer	Extraktion von Daten gemäß Regular Expressions
ScriptTransformer	komplexe Veränderungen mit Hilfe von Programmiersprachen (z.B. JavaScript)
TemplateTransformer	Erzeugen NEUER Felder

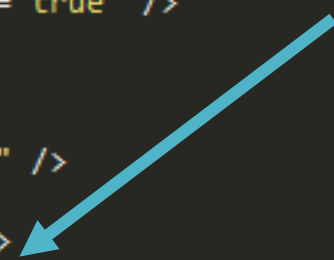
# Aktivierung von Transformer durch Attribute

```
<dataConfig>
  <dataSource type="URLDataSource" />
  <document>
    <entity name="slashdot"
      pk="link"
      url="http://rss.slashdot.org/Slashdot/slashdot"
      processor="XPathEntityProcessor"
      forEach="/rss/channel/item"
      transformer="DateFormatTransformer">

      <field column="source" xpath="/rss/channel/title" commonField="true" />
      <field column="source-link" xpath="/rss/channel/link" commonField="true" />
      <field column="subject" xpath="/rss/channel/subject" commonField="true" />

      <field column="title" xpath="/rss/channel/item/title" />
      <field column="link" xpath="/rss/channel/item/link" />
      <field column="description" xpath="/rss/channel/item/description" />
      <field column="creator" xpath="/rss/channel/item/creator" />
      <field column="item-subject" xpath="/rss/channel/item/subject" />
      <field column="date" xpath="/rss/channel/item/date" dateTimeFormat="yyyy-MM-dd'T'HH:mm:ss" />
      <field column="slash-department" xpath="/rss/channel/item/department" />
      <field column="slash-section" xpath="/rss/channel/item/section" />
      <field column="slash-comments" xpath="/rss/channel/item/comments" />

    </entity>
  </document>
</dataConfig>
```



```

<dataConfig>
  <document>
    <!--
      Note - In order to index attachments, set processAttachement="true" and drop
      Tika and its dependencies to example-DIH/solr/mail/lib directory
    -->
    <entity processor="MailEntityProcessor" user="email@gmail.com"
      password="password" host="imap.gmail.com" protocol="imaps"
      fetchMailsSince="2009-09-20 00:00:00" batchSize="20" folders="inbox" processAttachement="false"
      name="sample_entity"/>
  </document>
</dataConfig>

```

```

<dataConfig>
  <script><![CDATA[
    function CategoryPieces(row) {
      var pieces = row.get('category').split('/');
      var arr = new java.util.ArrayList();
      for (var i=0; i<pieces.length; i++) {
        arr.add(pieces[i]);
      }
      row.put('categorypieces', arr);
      row.remove('category');
      return row;
    }
  ]]></script>
  <document>
    <entity name="e" pk="id" transformer="script:CategoryPieces" query="select * from X">
      ....
    </entity>
  </document>
</dataConfig>

```

# Datenquellen

Name	Funktion
FileDataSource	Inhalt einer Datei wird als Quelle genommen
FieldReaderDataSource	Teile einer anderen Datenquelle mit einem anderen EntityProcessor verarbeiten
ContentStreamDataSource	nimmt alle Daten, die über POST gesendet werden
URLDataSource	URL zu einer Datei angeben, um Inhalt zu indexieren
JdbcDataSource	Datenquelle zur Indexierung relationaler Datenbanken



# Aktualisierung von Daten: Delta-Import

```
1 <dataConfig>
2   <dataSource driver="org.hsqldb.jdbcDriver" url="jdbc:hsqldb:./example-DIH/hsqldb/ex" user="sa" />
3   <document>
4     <entity name="item" query="select * from item"
5       deltaQuery="select id from item where last_modified > '${dataimporter.last_index_time}'">
6       <field column="NAME" name="name" />
7
8       <entity name="feature"
9         query="select DESCRIPTION from FEATURE where ITEM_ID='${item.ID}'"
10        deltaQuery="select ITEM_ID from FEATURE where last_modified > '${dataimporter.last_index_time}'"
11        parentDeltaQuery="select ID from item where ID=${feature.ITEM_ID}">
12        <field name="features" column="DESCRIPTION" />
13      </entity>
14
15      <entity name="item_category"
16        query="select CATEGORY_ID from item_category where ITEM_ID='${item.ID}'"
17        deltaQuery="select ITEM_ID, CATEGORY_ID from item_category where last_modified >
18          '${dataimporter.last_index_time}'"
19        parentDeltaQuery="select ID from item where ID=${item_category.ITEM_ID}">
20        <entity name="category"
21          query="select DESCRIPTION from category where ID = '${item_category.CATEGORY_ID}'"
22          deltaQuery="select ID from category where last_modified > '${dataimporter.last_index_time}'"
23          parentDeltaQuery="select ITEM_ID, CATEGORY_ID from item_category where
24            CATEGORY_ID=${category.ID}">
25          <field column="description" name="cat" />
26        </entity>
27      </entity>
28    </entity>
29  </document>
30</dataConfig>
```

# Mehr dazu...

<http://wiki.apache.org/solr/DataImportHandler>

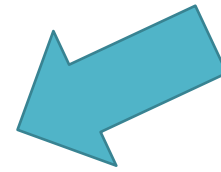
<https://cwiki.apache.org/confluence/display/solr/Uploading+Structured+Data+Store+Data+with+the+Data+Import+Handler>

# Solr-Konfiguration: solrconfig.xml



# Allgemeines

- zentrale Konfigurationsdatei neben schema.xml
- Index – Einstellungen
- Query – Einstellungen
- Request – Dispatcher – Einstellungen
- RequestHandler und SearchComponents
- UpdateHandler und UpdateRequestProcessorChain
- ResponseWriter
- Konfiguration der Admin – Oberfläche



besonders häufig benutzt

# Allgemeine Einstellungen

- Lucene-Version

```
<.luceneMatchVersion>4.4</luceneMatchVersion>
```

- externe Bibliotheken

```
<lib dir="../../../dist/" regex="solr-dataimporthandler-.*\.jar" />
```

- Datenverzeichnis für Index (per Default in data)
- Directory – Factory

# Index – Einstellungen

→ Default – Einstellungen reichen im Normalfall aus

```
<!-- <indexConfig> section could go here, but we want the defaults -->
```

- viele technische Daten (Speicher etc.)

<https://cwiki.apache.org/confluence/display/solr/IndexConfig+in+SolrConfig>

# Query – Einstellungen

→ Optimierung und Anpassung der Suche

z.B.

maxBooleanQueries

zahlreiche Cache-Informationen

zahlreiche Performance-Parameter

etc.



# Request – Dispatcher – Einstellungen

- grundlegende Verarbeitung von Requests
- technische Verarbeitung von Requests und HTTP – Caching

```
<requestDispatcher handleSelect="true" >  
  <!--Make sure your system has some authentication before enabling remote streaming! -->  
  <requestParsers enableRemoteStreaming="false" multipartUploadLimitInKB="2048" formdataUploadLimitInKB="2048" />  
  |  
  <httpCaching never304="true">  
    </httpCaching>  
  </requestDispatcher>
```

# Konfiguration der Admin – Oberfläche

- <admin> - Element
- Default – Einstellungen für die Admin – Oberfläche ändern
- Admin – Handler aktivieren und deaktivieren

```
<!--  
Admin Handlers - This will register all the standard admin RequestHandlers. Adding  
this single handler is equivalent to registering:  
  
<requestHandler name="/admin/luke" class="org.apache.solr.handler.admin.LukeRequestHandler" />  
<requestHandler name="/admin/system" class="org.apache.solr.handler.admin.SystemInfoHandler" />  
<requestHandler name="/admin/plugins" class="org.apache.solr.handler.admin.PluginInfoHandler" />  
<requestHandler name="/admin/threads" class="org.apache.solr.handler.admin.ThreadDumpHandler" />  
<requestHandler name="/admin/properties" class="org.apache.solr.handler.admin.PropertiesRequestHandler" />  
<requestHandler name="/admin/file" class="org.apache.solr.handler.admin.ShowFileRequestHandler" />  
  
If you wish to hide files under ${solr.home}/conf, explicitly register the ShowFileRequestHandler using:  
<requestHandler name="/admin/file" class="org.apache.solr.handler.admin.ShowFileRequestHandler" >  
  <lst name="invariants">  
    <str name="hidden">synonyms.txt</str>  
    <str name="hidden">anotherfile.txt</str>  
  </lst>  
</requestHandler>  
-->  
<requestHandler name="/admin/" class="org.apache.solr.handler.admin.AdminHandlers" />
```

# ResponseWriter

- zuständig für die Rückgabe von Ergebnissen
- Formatierung der Response von Solr
- zahlreiche ResponseWriter implizit vorhanden (xml, json, csv, usw.)

[http://localhost:8983/solr/select?q=first\\_author:"someQuery"&wt=php&indent=on](http://localhost:8983/solr/select?q=first_author:)

→ QueryResponseWriter für VelocityTemplates auch vorhanden

# RequestHandler

*Request handlers are responsible for accepting HTTP requests, performing searches, then returning the results.*

jeder RequestHandler wird in solrconfig.xml definiert und seine Einstellungen festgelegt:

- Namen, über den er angesprochen werden kann
- Klasse, die seine Funktion definiert (z.B. Suche oder Update)

```
<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>
```

# Suche und Searchfeatures - SearchHandler

```
<requestHandler name="/search" class="org.apache.solr.handler.component.SearchHandler">
  <lst name="defaults">
    <str name="echoParams">explicit</str>
  </lst>

  <arr name="components">
    <str>query</str>
    <str>facet</str>
    <str>mlt</str>
    <str>highlight</str>
    <str>debug</str>
  </arr>
</requestHandler>
```

# Suchkomponenten in Solr

- müssen in solrconfig.xml registriert werden um dann im entsprechenden RequestHandler referenziert zu werden
- Standardkomponenten sind vorregistriert

```
<searchComponent name="elevator"  
class="solr.QueryElevationComponent">  
  <str name="queryFieldType">string</str>  
  <str name="config-file">elevate.xml</str>  
</searchComponent>  
  
<requestHandler name="/elevate" class="solr.SearchHandler"  
startup="lazy">  
  <lst name="defaults">  
    <str name="echoParams">explicit</str>  
  </lst>  
  <arr name="last-components">  
    <str>elevator</str>  
  </arr>  
</requestHandler>
```

By default, the following components are available:

```
<searchComponent name="query"      class="solr.QueryComponent" />  
<searchComponent name="facet"      class="solr.FacetComponent" />  
<searchComponent name="mlt"        class="solr.MoreLikeThisComponent" />  
<searchComponent name="highlight" class="solr.HighlightComponent" />  
<searchComponent name="stats"      class="solr.StatsComponent" />  
<searchComponent name="debug"      class="solr.DebugComponent" />
```

# SearchComponents

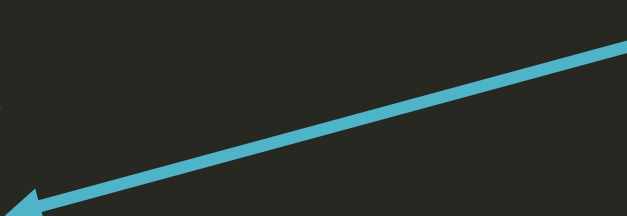
Name	Funktion
highlight	optische Hervorhebungen in der Trefferliste realisieren
facet	Gruppierung eines Ergebnis anhand eines Feldwerts
query	Komponente, die für die Suche zuständig ist
mlt	More Like This
stats	eine Statistikkomponente, die, basierend auf numerischen Feldwerten, statistische Werte errechnen kann.
debug	Debug – Komponente
elevator	kann Dokumente anhand vordefinierter Query in Ergebnisliste nach oben befördern
spellchecker	Korrekturvorschläge bei möglicher falscher Eingabe
terms	Zugriff auf Termfrequenz und Dokumentfrequenz
tvComponent	TermVectorComponent, Informationen zu indexierten Dokumenten
clustering	Gruppierung anhand ähnlicher Eigenschaften

# SearchComponents – Beispiel

QueryElevationComponent definieren und registrieren:

```
<searchComponent name="elevator" class="solr.QueryElevationComponent" >  
  <!-- pick a fieldType to analyze queries -->  
  <str name="queryFieldType">string</str>  
  <str name="config-file">elevate.xml</str>  
</searchComponent>
```

```
<requestHandler name="/select" class="solr.SearchHandler">  
  <!-- default values for query parameters can be specified, these  
       will be overridden by parameters in the request  
  -->  
  <lst name="defaults">  
    <str name="echoParams">explicit</str>  
    <int name="rows">10</int>  
    <str name="df">text</str>  
  </lst>  
  <arr name="components">  
    <str>query</str>  
    <str>facet</str>  
    <str>mlt</str>  
    <str>highlight</str>  
    <str>stats</str>  
    <str>debug</str>  
    <str>elevator</str>  
  </arr>
```





# Search Components

Fass ohne Boden!?



Ein Beispiel: Eigene Komponente bauen  
→ Suggesterkomponente

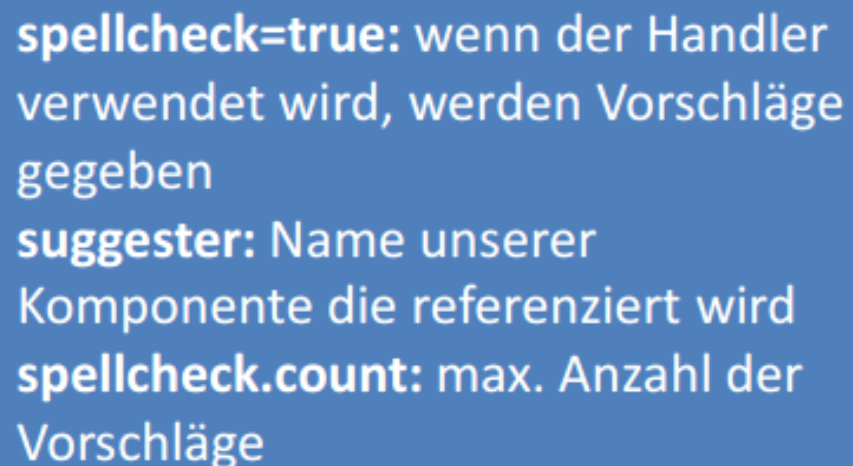
- Hinzufügen einer `<searchComponent>` zur `solrconfig.xml`

```
<searchComponent class="solr.SpellCheckComponent"
name="suggester">
  <lst name="spellchecker">
    <str name="name">suggester</str>
    <str name="classname">
org.apache.solr.spelling.suggest.Suggester</str>
    <str name="lookupImpl">
org.apache.solr.spelling.suggest.tst.TSTLookup</str>
    <str name="field">name</str>
    <str name="threshold">2</str>
  </lst>
</searchComponent>
```

**name:** Name der Komponente  
**Classname:** Klassenname  
**Field:** Feld auf dem die Vorschläge basieren  
**lookupImpl:** nach welchem Algorithmus folgt der Abgleich  
**threshold:** mind. Anzahl an Docs

- Definition eines passenden `<requestHandler>` in `solrconfig.xml`

```
<requestHandler
class="org.apache.solr.handler.component.
SearchHandler" name="/suggester">
  <lst name="defaults">
    <str name="spellcheck">true</str>
    <str name="spellcheck.dictionary">suggester</str>
    <str name="spellcheck.count">10</str>
  </lst>
  <arr name="components">
    <str>suggester</str>
  </arr>
</requestHandler>
```



**spellcheck=true:** wenn der Handler verwendet wird, werden Vorschläge gegeben  
**suggester:** Name unserer Komponente die referenziert wird  
**spellcheck.count:** max. Anzahl der Vorschläge

- Query: <http://localhost:8983/solr/suggester/?q=a>
- liefert folgende Ergebnisse

```
<arr name="suggestions">  
<str>a</str>  
<str>asus</str>  
<str>ati</str>  
<str>ata</str>  
<str>adata</str>  
<str>all</str>  
<str>allinone</str>  
<str>apple</str>  
</arr>
```

# Mehr dazu...

<http://wiki.apache.org/solr/SearchComponent>

<https://cwiki.apache.org/confluence/display/solr/RequestHandlers+and+SearchComponents+in+SolrConfig>

<http://searchhub.org/2013/05/31/solrconfig-xml-search-components-request-handlers-and-spellcheckers/>

Kommende Sitzung! (Facetten, Highlighting, Did You Mean etc.)

# Bücher

Grainger, T., Potter, T. (2014). Solr in Action. Shelter Island, NY: Manning Publications.

Klose, M., Wrigley. (2014). Einführung in Apache Solr. Praxiseinstieg in die innovative Suchtechnologie. Köln: O'Reilly Verlag.

# Übungsaufgabe

- books.xml (GRIPS) in Solr indexieren
- neuen Core erstellen oder bestehenden (collection1) nutzen
- passende schema.xml definieren (oder bestehende anpassen)
- DataImportHandler registrieren
- passende data-config.xml definieren
- Daten importieren
- Testen
  
- → Lösung nächste Woche



# (Über)nächste Woche

UI-Design mit Velocity – Templates

Search Components unter der Lupe

