

# Website Search with Apache Solr (tutorial)



tcworld conference 2018 - Stuttgart, Germany

Scott Prentice, Leximation, Inc.

# Introduction

— [ Scott Prentice, President of Leximation, Inc.

— [ Specializing in FrameMaker plugin development as well as structured FrameMaker conversions, consulting, and development. FrameMaker user/developer since 1991.

— [ Developed DITA-FMx, a FrameMaker plugin for efficient DITA authoring and publishing.

— [ Consulting for custom Help systems, creative/functional web applications, and EPUB solutions.

# Why website search?

- [ Keeps visitors on your site once they get there

- [ Gets customers to the right information sooner

- [ Gives you insight into what people need

- [ Potential source for new product ideas

# Layers of search

- [ Global — Cross-site, cross-domain, and world-wide search engines. Google, Bing, Yahoo, etc.
- [ Vertical — Domain-specific, cross-site, and world-wide search engines. Yelp, Truila, and others.
- [ Site — Full site-wide search
- [ Document — Search on a page or within a group of related pages (Find?)

# Website search options

- [ Remote search service — Service provided by third party, accessed through web form or API.
- [ Static JavaScript — Pre-compiled static “index” accessed via JavaScript to display matching results.
- [ Custom search application — Server-side application (PHP, Perl, Java, etc.), reading from collection

# Apache Solr

APACHE SOLR™ 7.5.0

Solr is the popular, blazing-fast, open source enterprise search platform built on Apache Lucene™.



# Apache Solr

- [ Open source enterprise search platform
- [ Java application runs on Linux, Mac, Windows
- [ Wrapper around Lucene indexing/search technology
- [ Hit highlighting, faceted search, real-time indexing, rich document support, Unicode compliant, really fast
- [ REST API plus native client APIs

# Solr setup options

- [ Solr “standalone”

- Single collection, no failover, or redundancy

- [ Solr “cloud” (SolrCloud)

- Collection spread across multiple servers (shards)
- Supports failover and redundancy via Zookeeper (distributed file system)



# System requirements

- [ Java 8 (Linux, MacOS, Windows)
- [ Likely multiple machines (or VMs); min of 5 for SolrCloud.
- [ Enough memory to support OS and applicaiton needs plus full index(es) in memory; 8-16 GB or much more.

# Search terminology

- [ Crawl — Process of reading content from website or file system. Creates a “feed” for indexing.
- [ Index — Process of reading the “feed” and creating or updating the search database or collection.
- [ Collection — Compiled data generated by the indexing process. Also, “index” or “search index.”
- [ Shard or Core — One or more components that make up a collection.

# Installing Solr (demo)

- [ Download archive

- [ Extract

- [ Install/setup (script for Linux systems only)

- [ Start

- [ Check Solr Admin

# Casing conventions

- [ These slides use the following casing conventions for special directory locations:

- **SOLR** — Directory containing the Solr application files

- **SOLR-DATA** — Directory containing the Solr data files

- [ These directory locations will differ based on your installation and operating system

# Installing Solr (Linux)

- [ Default install script ..

- Creates "solr" user

- Copies application files to `/opt/solr` (SOLR)

- Creates data folders at `/var/solr` (SOLR-DATA)

- Creates service at `/etc/init.d/solr`

- Creates include script at `/etc/default/solr.in.sh`

- [ After running the script, you're ready to roll!

# Installing Solr (Mac/Win)

- [ Manually create application and data folder structure

- [ Extract archive to application folder

- [ Edit default include script (**SOLR/bin/solr.in.sh** or **.cmd**)

```
SOLR_PID_DIR="SOLR-DATA"  
SOLR_HOME="SOLR-DATA/data"  
LOG4J_PROPS="SOLR-DATA/log4j.properties"  
SOLR_LOGS_DIR="SOLR-DATA/logs"  
SOLR_PORT="8983"
```

- [ Copy **solr.xml** and **zoo.cfg** from **SOLR/server/solr** to **SOLR-DATA/data**

# Starting Solr

Linux (if installed as a service): `sudo service solr start`

Mac: `SOLR/bin/solr start`

Win: `SOLR/bin/solr.cmd start`

This starts Solr in “standalone” mode

Now check Solr Admin! <http://localhost:8983/solr>

# Solr Admin



## Dashboard

Logging

Core Admin

Java Properties

Thread Dump

Core Selector

## Instance

Start about a minute ago

## Versions

**solr-spec** 7.5.0  
**solr-impl** 7.5.0 b5bf70b7e32d7ddd9742cc821d471c5fabd4e3df - jimcz - 2  
**lucene-spec** 7.5.0  
**lucene-impl** 7.5.0 b5bf70b7e32d7ddd9742cc821d471c5fabd4e3df - jimcz - 2

## JVM

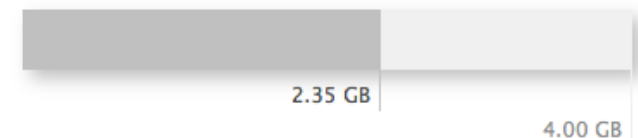
**Runtime** Oracle Corporation Java HotSpot(TM) 64-Bit Server VM 1.8.0\_171 2.  
**Processors** 8  
**Args**  
-DSTOP.KEY=solrrocks  
-DSTOP.PORT=7983  
-Djetty.home=/Users/saprentice/dev/solr/solr-7.5.0/server  
-Djetty.port=8983  
-Dlog4j.configurationFile=file:/Users/saprentice/dev/solr/solr-dat  
-Dsoler.data.home=  
-Dsoler.default.confdir=/Users/saprentice/dev/solr/solr-7.5.0/serv  
-Dsoler.install.dir=/Users/saprentice/dev/solr/solr-7.5.0  
-Dsoler.jetty.https.port=8983

## System 1.00 1.23 1.99

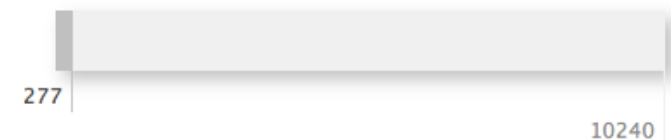
Physical Memory 95.1%



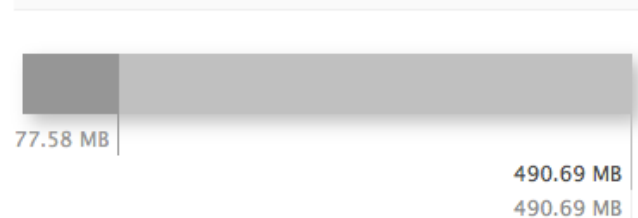
Swap Space 58.7%



File Descriptor Count 2.7%



## JVM-Memory 15.8%





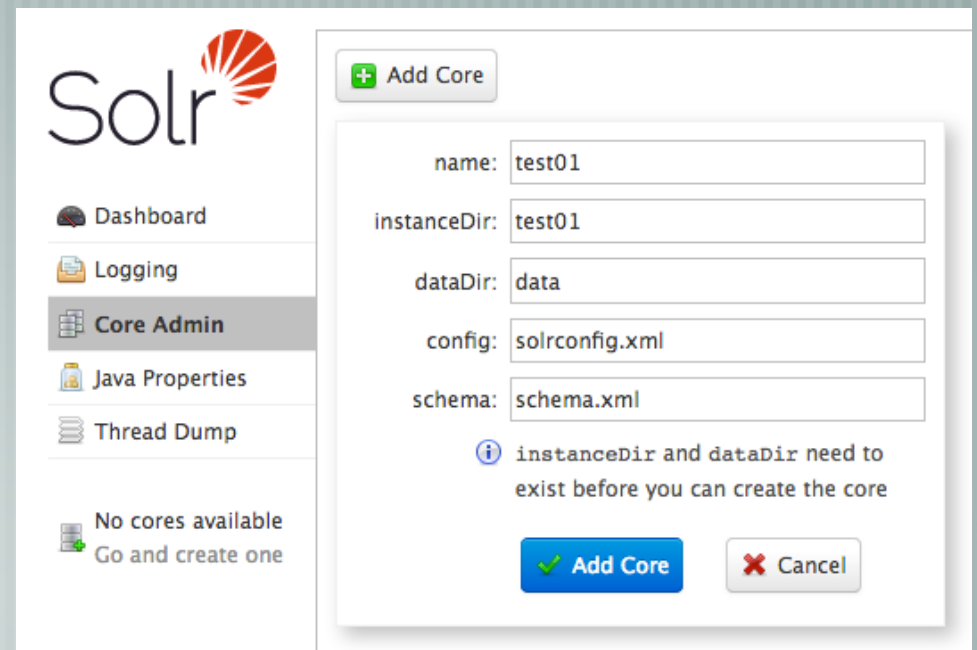
# Create empty collection

Copy “default” schema config files to data folder

```
$ cd SOLR/server/solr/configsets
```

```
$ cp -r _default SOLR-DATA/data/test01
```

In Solr Admin, use  
Core Admin to create  
new “test01” collection



The screenshot shows the Solr Admin web interface. On the left is a sidebar with navigation links: Dashboard, Logging, Core Admin (highlighted), Java Properties, Thread Dump, and a status section indicating 'No cores available' with a link to 'Go and create one'. The main content area displays the 'Add Core' dialog box. This dialog has a title bar with a green plus icon and the text 'Add Core'. It contains five input fields: 'name' (test01), 'instanceDir' (test01), 'dataDir' (data), 'config' (solrconfig.xml), and 'schema' (schema.xml). Below these fields is an information icon and a message: 'instanceDir and dataDir need to exist before you can create the core'. At the bottom of the dialog are two buttons: a blue 'Add Core' button with a green checkmark and a grey 'Cancel' button with a red X.

Solr

Dashboard  
Logging  
Core Admin  
Java Properties  
Thread Dump  
No cores available  
Go and create one

+ Add Core

name: test01  
instanceDir: test01  
dataDir: data  
config: solrconfig.xml  
schema: schema.xml

i instanceDir and dataDir need to exist before you can create the core

✓ Add Core ✗ Cancel

# Upload sample content

- [ Use “post” tool to upload sample data

```
$ cd SOLR
```


```
$ ./bin/post -c test01 example/exampledocs/*
```

- [ Post tool uses default algorithm to extract data and upload to collection “test01”

# Basic testing

- [ Solr Admin > Core Selector > test01 > Query
- [ “Execute Query” using default **\*:\*** query
- [ Review fields and values resulting from default schema and sample content
- [ This schema “works,” but likely not ideal

# Solr Admin - Execute Query



- Dashboard
- Logging
- Core Admin
- Java Properties
- Thread Dump
- test01
  - Overview
  - Analysis
  - Dataimport
  - Documents
  - Files
  - Ping
  - Plugins / Stats
  - Query
  - Replication
  - Schema
  - Segments info

Request-Handler (qt)

/select

— common —

q

\*\*\*

fq

sort

start, rows

0 10

fl

df

Raw Query Parameters

key1=val1&key2=val2

wt

-----

☐ indent off

☐ debugQuery

☐ dismax

☐ edismax

http://localhost:8983/solr/test01/select?q=\*\*\*

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 123,
    "params": {
      "q": "***",
      "_": "1541808836981"
    }
  },
  "response": {
    "numFound": 52, "start": 0, "docs": [
      {
        "id": "0553573403",
        "cat": ["book"],
        "name": ["A Game of Thrones"],
        "price": [7.99],
        "inStock": [true],
        "author": ["George R.R. Martin"],
        "series_t": "A Song of Ice and Fire",
        "sequence_i": 1,
        "genre_s": "fantasy",
        "_version_": 1615431737023660032
      },
      {
        "id": "0553579908",
        "cat": ["book"],
        "name": ["A Clash of Kings"],
        "price": [7.99],
        "inStock": [true],
        "author": ["George R.R. Martin"],
        "series_t": "A Song of Ice and Fire",
        "sequence_i": 2,
        "genre_s": "fantasy",
        "_version_": 1615431738104741888
      }
    ]
  }
}
```

# Solr query primer

— [ `q=<FIELD>:<VALUE>`

— [ `q=*:*` (match all)

— [ `q=cat:electronics`

— [ `q=name:ipod`

— [ `q=price:[10 TO 20]`

— [ `q=manufacturedate_dt:[NOW-13YEARS TO NOW-12YEARS]`

# Solr query primer

- [ List all facets for “cat” field:

**facet=on&facet.field=cat&rows=0&start=0**

- [ Include specific fields: **fl=id,name,manu**

- [ Specify format (default JSON): **wt=xml** or **wt=csv**

# Website integration options

- [ Search form with list of results

- [ Search context with hit highlighting

- [ Faceting for tags or categories

- [ Auto-complete, auto-suggest, spellchecking

- [ Auto-generate related links or “more like this”

- [ Use REST API or native client languages

# Content sources

- [ Content may come from various sources ..
  - Documentation (content, metadata, tags)
  - User comments
  - Product support cases
  - Marketing material
  - External website content



# Schema

- [ Defines the structure and fields in your index
- [ Based on type of integration and type of content
- [ Defines field types with optional index or query analyzers (tokenizers or filters)
- [ Defines static or dynamic fields
- [ Each Solr server can have multiple collections with different schemas

# Simple schema

```
<schema name="myschema 1.0" version="1.6">  
  <uniqueKey>id</uniqueKey>  
  
  <fieldType name="string" class="solr.StrField"  
sortMissingLast="true" docValues="true"/>  
  
  <field name="id" type="string" required="true"  
indexed="true" stored="true"/>  
  <field name="title" type="string"/>  
  <field name="type" type="string"/>  
  <field name="content" type="string"/>  
</schema>
```

# Schema nodes

- [ <fieldType> – All possible types of fields in this schema
- [ <field> – Static fields in this schema
- [ <dynamicField> – Dynamic fields (wildcard match)
- [ <copyField> – Duplicates the named field to another field

# Schema - <fieldType>

```
<fieldType name="NAME" class="CLASS" [MORE] >  
  [<analyzer type="TYPE">]  
</fieldType>
```

— [ NAME — Unique name for field type

— [ CLASS — Java class for processing (may be user-defined)

— [ MORE — Additional class-specific properties

— [ TYPE — Analyzer type (index, query, or both), may contain tokenizer and multiple filter nodes (classes)

# fieldType analyzers

- [ Modifies the data as it's indexed or queried
- [ Make use of default tokenizers and filters
- [ Can create your own (via Java)
- [ Very powerful feature

# Schema - `<field>`

```
<field name="NAME" type="TYPE" [MORE]/>
```

- [ NAME — Unique name for field to match in feed
- [ TYPE — References the name of the associated fieldType
- [ MORE — Additional type-specific properties

# Schema - `<dynamicField>`

```
<dynamicField name="NAME" type="TYPE"  
  [MORE] />
```

- [ NAME — Unique name for field with leading or trailing asterisk for wildcard match in feed
- [ TYPE — References the name of the associated fieldType
- [ MORE — Additional type-specific properties

# Schema - <copyField>

```
<copyField source="SOURCE" dest="DEST"  
  [maxChars= "NUM"] />
```

— [ SOURCE — Name of field to copy (may include leading/trailing asterisks)

— [ DEST — Name of field to copy to

— [ NUM — Limits the number of characters (optional)



# Updating configuration

- [ Rename **managed-schema** to **schema.xml** and edit

- [ Update **solrconfig.xml**

- [ Update stopwords, synonyms, locale-specific files

- [ Delete unused files

- [ Restart Solr: **SOLR/bin/solr restart**  
(or **sudo service solr restart** if using service on Linux)

# Uploading content to Solr

- [ Can be in many formats (HTML, XML, JSON, PDF, RTF, ..)
- [ Best to use XML or JSON to sync with schema
- [ One or more files with content
- [ Should be flat (nested structures are possible)

# XML feed

```
<add>
  <doc>
    <field name="id">filename-one</field>
    <field name="title">Some Title</field>
    <field name="type">tutorial</field>
    <field name="content">All of the doc content.
Best to remove line breaks and markup. </field>
  </doc>
  <doc>
    <field name="id">filename-two</field>
    <field name="title">Another Title</field>
    <field name="content">More content.</field>
  </doc>
  ...
</add>
```

# JSON feed

```
[ {  
    id: "filename-one",  
    title: "Some Title",  
    type: "tutorial",  
    content: "All of the content for the document.  
Best to remove line breaks and markup."  
}, {  
    id: "filename-two",  
    title: "Another Title",  
    type: "tutorial",  
    content: "And more content."  
}  
...  
]
```

# Using “real” data (demo)

- [ Create new schema

- [ Generate feed from content

- [ Upload feed to index

- [ Develop search UI (JavaScript)

# Create new schema

- [ Copy “default” schema config files to data folder

```
$ cd SOLR/server/solr/configsets
```

```
$ cp -r _default SOLR-DATA/data/test02
```

- [ Edit schema config files based on your needs (simplify)

- [ Schema fields must match data being indexed

- [ In Solr Admin, create new “test02” collection  
(watch for and correct errors)

# Generate and upload feed

- [ Process your content to generate a JSON feed  
(see [html2json.pl](#) script)

- [ Use curl to upload feed to collection

```
$ curl 'http://localhost:8983/solr/test02/update/json?  
commit=true' -H 'Content-type:application/json' -data-binary  
@test02.json
```

- [ Test queries in Solr Admin

# Updating content

- [ Uploading another feed ..

- duplicate IDs replaces existing records

- new IDs add those records

- [ Delete entire index ..

```
$ curl 'http://localhost:8983/solr/test02/update?  
commit=true' -H 'Content-Type: text/xml' -data-binary  
'<delete><query>*:*</query></delete>'
```



# Develop a search UI

- [ REST API is very flexible and easy to test
- [ Simple JavaScript UI is good place to start
- [ Use jQuery to make the scripting easier
- [ Sample JavaScript provides options for basic search results or hit highlighting (customize as needed!)

# CORS?

- [ Cross-Origin Resource Sharing

- [ Restricts sharing of resources across domains

- [ Will be an issue if requesting Solr results via JavaScript (not with PHP or other server scripting)

- [ Need to edit this file in Solr installation

- [ `SOLR/server/solr-webapp/webapp/WEB-INF/web.xml`

- [ See: <https://opensourceconnections.com/blog/2015/03/26/going-cross-origin-with-solr/>

# Further investigation

- [ Consider implementing:

- Auto-suggest

- Auto-complete

- Spell checking

- [ Try auto-generating related links based on tags

- [ Other creative ideas?

# Web crawlers

- [ Apache Nutch — Integrates directly with Solr (Java)
- [ Heritrix — Internet Archive's open-source, extensible, web-scale, archival-quality web crawler (Java)
- [ GNU Wget — Command line tool for retrieving files using HTTP, HTTPS, FTP and FTPS. (Linux)
- [ See "Top 50 open source web crawlers"

# Taking it to production?

- [ Restrict access to Solr (iptables command on Linux)
- [ Consider using SolrCloud
  - Provides failover and redundancy
  - Zookeeper adds complexity
  - Multiple servers

# Server access issues?

- [ What if I don't have easy access to a server?
  - [linode.com](https://linode.com) — Very affordable (\$5/mo or more) linux servers for development and testing.
  - [websolr.com](https://websolr.com) — Reasonable cost (\$59 or \$549/mo). Fully configured Solr installations. You provide the schema and content.

# Wrap-up

- [ Solr is an incredibly powerful and full featured search platform that can be implemented in stages
- [ Solr does require development resources, but it's not necessarily "rocket science"
- [ Solr gives you control over your customer's website search experience

# Resources

— [ Apache Solr — [lucene.apache.org/solr/](http://lucene.apache.org/solr/)

— [ Apache Solr Reference Guide — [lucene.apache.org/solr/guide/7\\_5/](http://lucene.apache.org/solr/guide/7_5/)

— [ solr-user mailing list — [lucene.apache.org/solr/community.html](http://lucene.apache.org/solr/community.html)

— [ Top 50 open source web crawlers — [bigdata-madesimple.com/top-50-open-source-web-crawlers-for-data-mining/](http://bigdata-madesimple.com/top-50-open-source-web-crawlers-for-data-mining/)

— [ Scott Prentice <scott AT leximation.com> — [www.leximation.com](http://www.leximation.com)



# Feedback



**Your opinion is important!**

Please tell us what you thought of the lecture. We look forward to your feedback via smartphone or tablet.

**Scan the QR code  
or visit the URL:**

<http://ux09.honestly.de>

The feedback tool will be available  
even after the conference!