Website Search with Apache Solr

STC Summit 2019 - Denver

Scott Prentice, Leximation, Inc. - @saprentice

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 DITA development, custom Help systems, creative/functional web applications, and EPUB solutions.

Goals/disclaimers

- Introduce Apache Solr.
- It’s not as hard as you may think.
- Hoping you’ll try it on your own!
- May generate more questions than it answers.
- Warning: This is on the “tech” side of techcomm!
- This is a live demo ...

Why website search?

- Keep visitors on your site
- Get customers to the right information
- Gain insight into what people need
- Potential source for new product ideas

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™ 8.0.0

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)

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
- Extract
- Install (Linux = scripted; Mac/Windows = manual)
- Start
- Test

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 (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)

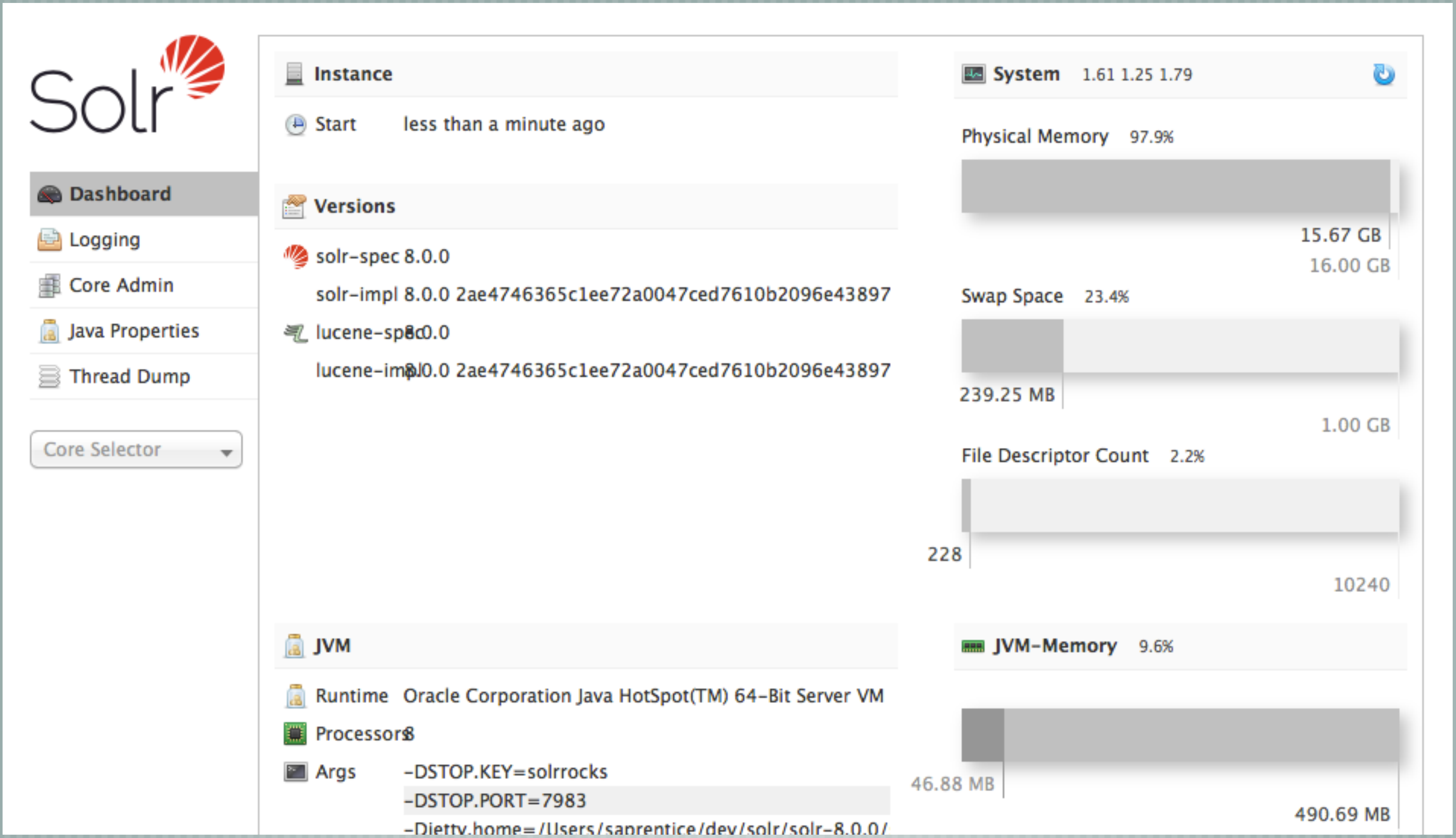
```
SOLR_PID_DIR="SOLR-DATA"
SOLR_HOME="SOLR-DATA/data"
LOG4J_PROPS="SOLR-DATA/log4j.properties"
SOLR_LOGS_DIR="SOLR-DATA/logs"
SOLR_PORT="8983"
```

- 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
- Check Solr Admin! http://localhost:8983/solr

Solr Admin



Instance

Start less than a minute ago

Versions

solr-spec 8.0.0

solr-impl 8.0.0 2ae4746365c1ee72a0047ced7610b2096e43897

lucene-spec 8.0.0

lucene-impl 8.0.0 2ae4746365c1ee72a0047ced7610b2096e43897

System 1.61 1.25 1.79

Physical Memory 97.9%

15.67 GB

16.00 GB

Swap Space 23.4%

239.25 MB

1.00 GB

File Descriptor Count 2.2%

228

10240

JVM

Runtime Oracle Corporation Java HotSpot(TM) 64-Bit Server VM

Processors 8

Args -DSTOP.KEY=solrrocks

-DSTOP.PORT=7983

-Djetty.home=/Users/saprentice/dev/solr/solr-8.0.0/

JVM-Memory 9.6%

46.88 MB

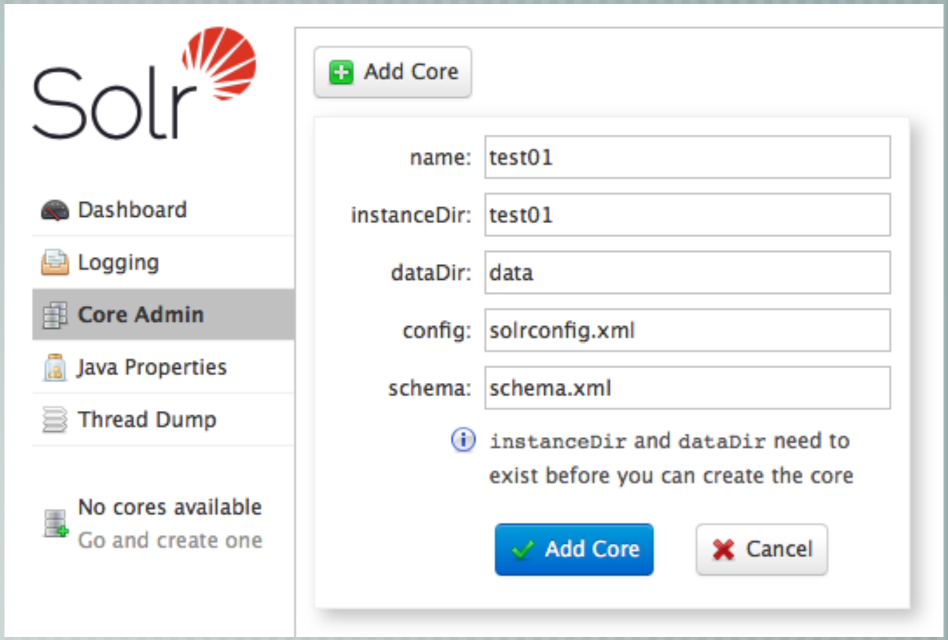
490.69 MB

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



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

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

Upload sample content

- Use “post” tool to upload sample data

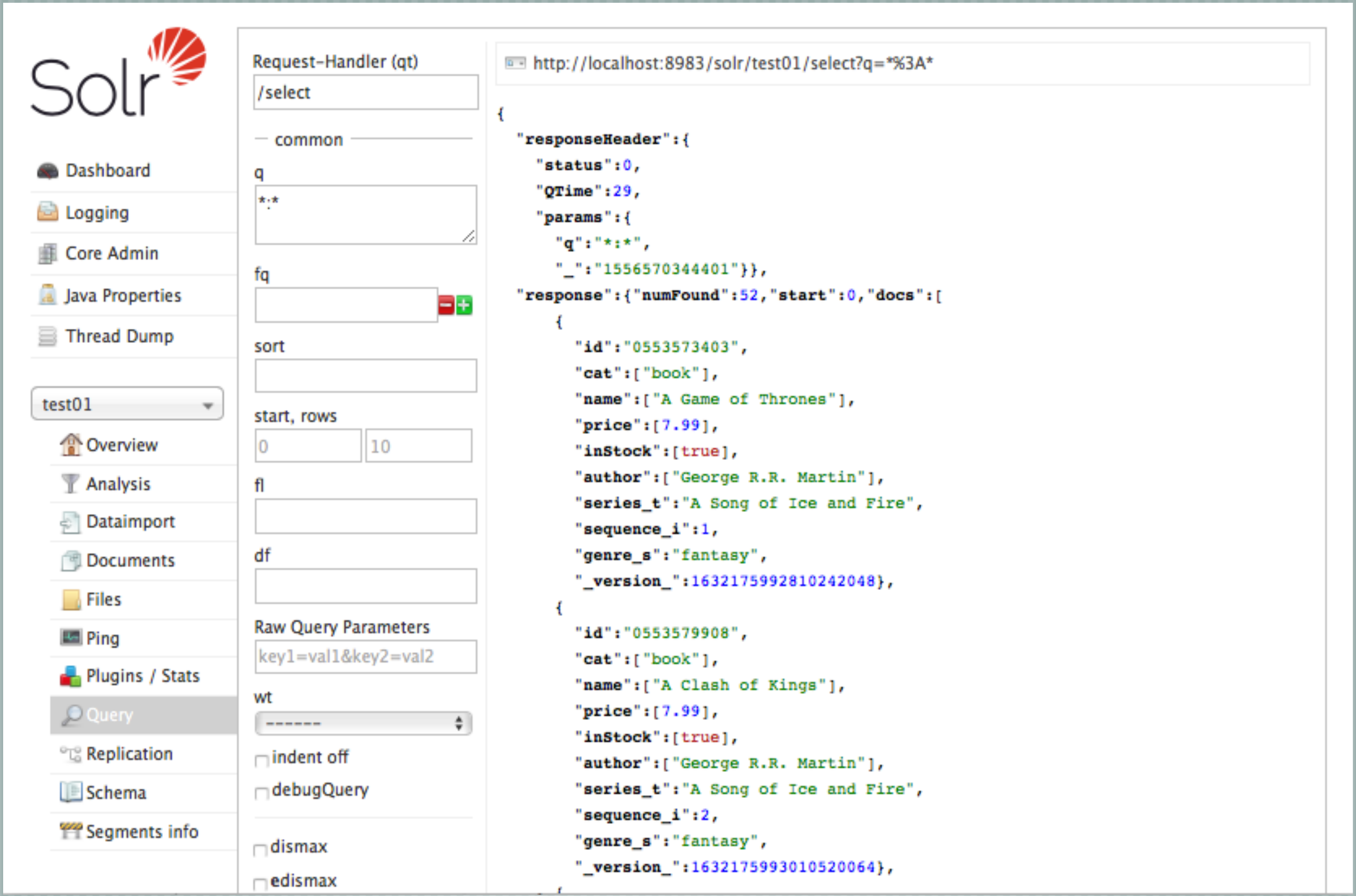
```
$ cd SOLR
$ ./bin/post -c test01 example/exampledocs/*
```

- 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



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=*%3A*

```
{
  "responseHeader":{
    "status":0,
    "QTime":29,
    "params":{
      "q":"*:*",
      "_":"1556570344401"}},
  "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_":1632175992810242048},
      {
        "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_":1632175993010520064},
```

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

Making it real...

- Customize the schema to suit your needs. Consider ..
  - Content sources
  - Website integration
- Generate JSON or XML feed from content
- Upload feed to Solr collection
- Develop search UI (typically JavaScript)

Content sources

- Documentation (content, metadata, tags)
- User comments
- Product support cases
- Marketing material
- Website content (on-site or 3rd party)
- Anything on a file system or website!

Website integration options

- Search form with list of results
- Search context with hit highliting
- 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

Schema

- Defines the structure and fields in your index
- Field names must match names in content feed
- Defines field types with optional index or query analyzers (tokenizers or filters)
- Defines static or dynamic fields
- Each Solr server can have multiple collections, each 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>
```

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

Schema modifications

- Rename managed-schema to schema.xml and edit
- Update solrconfig.xml
- Update stopwords, synonyms, locale-specific files
- Delete unused files
- Restart Solr after updates: SOLR/bin/solr restart
  (or sudo service solr restart if using service on Linux)

Generate content feed

- Crawl your content to create XML or JSON feed(s)
- Should be a flat structure
- Could be part of build process or separate script
- [DEMO] The html2json.pl script is one example

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."
  }
  …
]
```

Upload content feed

- Use curl to upload feed to Solr

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

- Test queries in Solr Admin or browser URL

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
- [DEMO] Sample JavaScript provides options for basic search results or hit highlighting

CORS?

- Cross-Origin Resource Sharing
- Restricts sharing of resources across domains
- Will be an issue if requesting Solr results via JavaScript (not with server-side scripting like PHP)
- 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/

Updating configuration

- Rename managed-schema to schema.xml and edit
- Edit other config files
- Restart Solr: SOLR/bin/solr restart

Updating content

- Uploading another feed ..
  - duplicate IDs replaces existing records
  - new IDs add those records
- Delete entire index if needed ..

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

Taking it to production?

- Restrict access to Solr (iptables command on Linux)
- Consider using SolrCloud
  - Provides failover and redundancy
  - Zookeeper adds complexity
  - Multiple servers/VMs (min of 5)
  - RAM for full indexes in memory; 8-16 GB or more

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"

Server access issues?

- No easy access to a server?
  - linode.com – Very affordable ($5/mo or more) linux servers for development and testing.
  - 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/
- Apache Solr Reference Guide – lucene.apache.org/solr/guide/
- solr-user mailing list – lucene.apache.org/solr/community.html
- Top 50 open source web crawlers – bigdata-madesimple.com/top-50-open-source-web-crawlers-for-data-mining/
- Scott Prentice <scott AT leximation.com> – www.leximation.com
- Sample/demo files available, email Scott.