

TikaExtractOnlyExampleOutput

Example of running the [ExtractingRequestHandler](#) with extract only:

```
# WARNING: This handler doesn't take quite these same parameters in the Solr 1.4 release.
# Instead try something more like
# curl http://localhost:8983/solr/update/extract?extractOnly=true
#      --data-binary @tutorial.html  -H 'Content-type:text/html'
curl http://localhost:8983/solr/update/extract?ext.idx.attr=true\&ext.extract.only=true\&ext.def.fl=text  --
data-binary @tutorial.html  -H 'Content-type:text/html'
```

Results in:

```
<response>
<lst name="responseHeader"><int name="status">0</int><int name="QTime">58</int></lst><str>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Solr tutorial (version 1.3.0.2008.09.16.15.17.27)</title>
  </head>
  <body>
    <a href="http://www.apache.org/">apache</a> & <a
      href="http://lucene.apache.org/">lucene</a>
```

```
<a
      href="http://lucene.apache.org/">
```

```
<a
      href="http://lucene.apache.org/solr/">
```

```
<ul>
<li>
<a
      href="index.html">Main</a>
</li>
<li>
<a
      href="http://wiki.apache.org/solr">Wiki</a>
</li>
</ul>
```

About

[index.html](#) > Welcome

[who.html](#) > Who We Are

Documentation

[features.html](#) > Features

Tutorial

<http://wiki.apache.org/solr/> > Docs (Wiki)

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

[api/index.html](#) > All Javadoc

[api/solrj/index.html](#) > SolrJ Javadoc

Resources

<http://www.apache.org/dyn/closer.cgi/lucene/solr/> > Download

[mailing_lists.html](#) > Mailing Lists

[issue_tracking.html](#) > Issue Tracking

[version_control.html](#) > Version Control

Related Projects

[](#)

<http://lucene.apache.org/java/>>Lucene Java>

<a

<http://lucene.apache.org/nutch/>>Nutch>

This document is for Apache Solr version 1.3.0.2008.09.16.15.17.27. If you are using a different version of Solr, please consult the documentation that was distributed with the version you are using.

<a

<tutorial.pdf>>
PDF>

<h1>Solr tutorial</h1>

This document is for Apache Solr version 1.3.0.2008.09.16.15.17.27. If you are using a different version of Solr, please consult the documentation that was distributed with the version you are using.

<a

<#Overview>>Overview>

<a

<#Requirements>>Requirements>

<a

<#Getting+Started>>Getting Started>

<a

<#Indexing+Data>>Indexing Data>

<a

<#Updating+Data>>Updating Data>

<a

<#Deleting+Data>>Deleting Data>

<a

<#Querying+Data>>Querying Data>

<a

<#Sorting>>Sorting>

<a

<#Text+Analysis>>Text Analysis>

```

<ul>
<li>
<a
href="#Analysis+Debugging">Analysis Debugging</a>
</li>
</ul>
</li>
</ul>
</li>
<li>
<a
href="#Conclusion">Conclusion</a>
</li>
</ul>

<a href="/">
  <a
    href="/">
</h2>Overview</h2>

<p>
This document covers the basics of running Solr using an example
schema, and some sample data.
</p>

<a href="/">
  <a
    href="/">
</h2>Requirements</h2>

<p>
To follow along with this tutorial, you will need...
</p>
<ol>
<li>Java 1.5 or greater. Some places you can get it are from
  <a
    href="http://java.sun.com/j2se/downloads.html">Sun</a>,
  <a
    href="http://www-106.ibm.com/developerworks/java/jdk/">IBM</a>, or
  <a
    href="http://www.bea.com/jrockit/">BEA</a>.

  Running java -version at the command line should indicate a version
  number starting with 1.5.
</li>
<li>A <a
  href="http://www.apache.org/dyn/closer.cgi/lucene/solr/">Solr release</a>.
</li>
<li>FireFox or Mozilla is the preferred browser to view the admin pages, as
  the current stylesheet doesn't look good on Internet Explorer.
</li>
</ol>

<a href="/">
  <a
    href="/">
</h2>Getting Started</h2>

<p>
Please run the browser showing this tutorial and the Solr server on the same machine so tutorial links will
correctly point to your Solr server.

```

```

</p>
<pre>
Begin by unzipping the Solr release and changing your working directory
to be the "example" directory. (Note that the base directory name may vary with the version of Solr
downloaded.)
</pre>
<pre>
chrish@asimov:~solr$ ls
solr-nightly.zip
chrish@asimov:~solr$ unzip -q solr-nightly.zip
chrish@asimov:~solr$ cd solr-nightly/example/
</pre>
<pre>
Solr can run in any Java Servlet Container of your choice, but to simplify
this tutorial, the example index includes a small installation of Jetty. In order to compile JSPs, this
version of Jetty requires that you run "java" from a JDK, not from a JRE.
</pre>
<pre>
To launch Jetty with the Solr WAR, and the example configs, just run the start.jar ...
</pre>
<pre>
chrish@asimov:~/solr/example$ java -jar start.jar
1 [main] INFO org.mortbay.log - Logging to org.slf4j.impl.SimpleLogger@1f436f5 via org.mortbay.log.Slf4jLog
334 [main] INFO org.mortbay.log - Extract jar:file:/home/chrish/solr/example/webapps/solr.war!/ to /tmp
/Jetty__solr/webapp
Feb 24, 2006 5:54:52 PM org.apache.solr.servlet.SolrServlet init
INFO: user.dir=/home/chrish/solr/example
Feb 24, 2006 5:54:52 PM org.apache.solr.core.SolrConfig <clinit>
INFO: Loaded Config solrconfig.xml

...

1656 [main] INFO org.mortbay.log - Started SelectChannelConnector @ 0.0.0.0:8983
</pre>
<pre>
This will start up the Jetty application server on port 8983, and use your terminal to display the logging
information from Solr.
</pre>
<pre>
You can see that the Solr is running by loading <a
href="http://localhost:8983/solr/admin/">http://localhost:8983/solr/admin/</a> in your web
browser. This is the main starting point for Administering Solr.
</pre>

<a href="/">
    <a
        href="/">
</h2>Indexing Data</h2>

<pre>
Your Solr server is up and running, but it doesn't contain any data. You can modify a Solr index by POSTing
XML Documents containing
instructions to add (or update) documents, delete documents, commit pending adds and deletes, and optimize your
index.
</pre>
<pre>
The exampledocs directory contains samples of the types of instructions Solr expects, as well as a java utility
for posting
them from the command line (a post.sh shell script is also available, but for this tutorial we'll use the cross-
platform Java
client).
</pre>
<pre>
To try this, open a new terminal window, enter the exampledocs directory, and run "java -jar post.jar" on some
of the
XML files in that directory, indicating the URL of the Solr server:
</pre>

```

```

<pre>
chris@asimov:~/solr/example/exampledocs$ java -jar post.jar solr.xml monitor.xml
SimplePostTool: version 1.2
SimplePostTool: WARNING: Make sure your XML documents are encoded in UTF-8, other encodings are not currently
supported
SimplePostTool: POSTing files to http://localhost:8983/solr/update..
SimplePostTool: POSTing file solr.xml
SimplePostTool: POSTing file monitor.xml
SimplePostTool: COMMITting Solr index changes..
</pre>
<p>
You have now indexed two documents in Solr, and committed these changes.
You can now search for "solr" using the "Make a Query" interface on the Admin screen, and you should get one
result.
Clicking the "Search" button should take you to the following URL...
</p>
<p>
<a
      href="http://localhost:8983/solr/select/?stylesheet=&amp;q=solr&amp;version=2.1&amp;
start=0&amp;rows=10&amp;indent=on"&gt;http://localhost:8983/solr/select/?stylesheet=&amp;q=solr&amp;
&amp;version=2.1&amp;start=0&amp;rows=10&amp;indent=on</a>
</p>
<p>
You can index all of the sample data, using the following command (assuming your shell supports the *.xml
notation):
</p>
<pre>
chris@asimov:~/solr/example/exampledocs$ java -jar post.jar *.xml
SimplePostTool: version 1.2
SimplePostTool: WARNING: Make sure your XML documents are encoded in UTF-8, other encodings are not currently
supported
SimplePostTool: POSTing files to http://localhost:8983/solr/update..
SimplePostTool: POSTing file hd.xml
SimplePostTool: POSTing file ipod_other.xml
SimplePostTool: POSTing file ipod_video.xml
SimplePostTool: POSTing file mem.xml
SimplePostTool: POSTing file monitor.xml
SimplePostTool: POSTing file monitor2.xml
SimplePostTool: POSTing file mp500.xml
SimplePostTool: POSTing file sd500.xml
SimplePostTool: POSTing file solr.xml
SimplePostTool: POSTing file spellchecker.xml
SimplePostTool: POSTing file utf8-example.xml
SimplePostTool: POSTing file vidcard.xml
SimplePostTool: COMMITting Solr index changes..
</pre>
<p>
...and now you can search for all sorts of things using the default <a
      href="http://lucene.apache.org/java/docs/queryparsersyntax.html"&gt;Lucene QueryParser
syntax</a>...
</p>
<ul>
<li>
<a
      href="http://localhost:8983/solr/select/?version=2.1&amp;indent=on&amp;q=video"&gt;
video</a>
</li>
<li>
<a
      href="http://localhost:8983/solr/select/?version=2.1&amp;indent=on&amp;q=name:video"
&gt;name:video</a>
</li>
<li>
<a
      href="http://localhost:8983/solr/select/?version=2.1&amp;indent=on&amp;q=%2Bvideo+%2Bprice%
3A[*+TO+400]"&gt;+video +price:[* TO 400]</a>
</li>

```

```

</li>

</ul>

<a href="/">
    <a
        href="/">
</h2>Updating Data</h2>

<p>
You may have noticed that even though the file solr.xml has now
been POSTed to the server twice, you still only get 1 result when searching for
"solr". This is because the example schema.xml specifies a "uniqueKey" field
called "id". Whenever you POST instructions to Solr to add a
document with the same value for the uniqueKey as an existing document, it
automatically replaces it for you. You can see that that has happened by
looking at the values for numDocs and maxDoc in the
"CORE" section of the statistics page... </p>
<p>

<a
    href="http://localhost:8983/solr/admin/stats.jsp">http://localhost:8983/solr/admin/stats.jsp<
/a>

</p>
<p>
numDoc should be 16 (because some of our 11 example XML files contain more than one &lt;doc>), but
maxDoc may be larger as
the maxDoc count includes logically deleted documents that have not yet been removed from the index. You can re-
post the sample XML
files over and over again as much as you want and numDocs will never increase, because the new documents will
constantly be replacing the old.
</p>
<p>
Go ahead and edit the existing XML files to change some of the data, and re-run the java -jar post.jar command,
you'll see your changes reflected in subsequent searches.
</p>
<a href="/">
    <a
        href="/">
</h3>Deleting Data</h3>
<p>You can delete data by POSTing a delete command to the update URL and specifying the value
of the document's unique key field, or a query that matches multiple documents (be careful with that
one!). Since these commands
are smaller, we will specify them right on the command line rather than reference an XML file.
</p>
<p>Execute the following command to delete a document</p>
<pre>java -Ddata=args -Dcommit=no -jar post.jar "&lt;delete>&lt;id>SP2514N&lt;
/id>&lt;/delete>"</pre>
<p>Now if you go to the <a
    href="http://localhost:8983/solr/admin/stats.jsp">statistics</a> page and scroll down
to the UPDATE_HANDLERS section and verify that "deletesPending : 1"</p>
<p>If you search for <a
    href="http://localhost:8983/solr/select?q=id:SP2514N">id:SP2514N</a> it will still be
found,
because index changes are not visible until changes are flushed to disk, and a new searcher is opened.
To cause
this to happen, send a commit command to Solr (post.jar does this for you by default):</p>
<pre>java -jar post.jar</pre>
<p>Now re-execute the previous search and verify that no matching documents are found. Also revisit the
statistics page and observe the changes in both the UPDATE_HANDLERS section and the CORE section.</p>
<p>Here is an example of using delete-by-query to delete anything with
<a
    href="http://localhost:8983/solr/select?q=name:DDR&fl=name">DDR</a> in the
name:</p>
<pre>java -Ddata=args -jar post.jar "&lt;delete>&lt;query>name:DDR&lt;

```

```

</pre>
<p>Commit can be a very expensive operation so it's best to make many changes to an index in a batch and then send the commit command at the end. There is also an optimize command that does the same thing as commit, in addition to merging all index segments into a single segment, making it faster to search and causing any deleted documents to be removed. All of the update commands are documented <a href="http://wiki.apache.org/solr/UpdateXmlMessages">here</a>.</p>
<p>To continue with the tutorial, re-add any documents you may have deleted by going to the exampledocs directory and executing</p>
<pre>java -jar post.jar *.xml</pre>

<a href="/">
  <a href="/">
</h2>Querying Data</h2>

<p>
  Searches are done via HTTP GET on the select URL with the query string in the q parameter.
  You can pass a number of optional <a href="http://wiki.apache.org/solr/StandardRequestHandler">request parameters</a> to the request handler to control what information is returned. For example, you can use the "fl" parameter to control what stored fields are returned, and if the relevancy score is returned...
</p>
<ul>
<li>
<a href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;fl=name,id">q=video&amp;fl=name,id</a> (return only name and id fields)
</li>
<li>
<a href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;fl=name,id,score">q=video&amp;fl=name,id,score</a> (return relevancy score as well)
</li>
<li>
<a href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;fl=*,score">q=video&amp;fl=*,score</a> (return all stored fields, as well as relevancy score)
</li>
<li>
<a href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=price desc&amp;fl=name,id">q=video&amp;sort=price desc&amp;fl=name,id</a> (add sort specification: sort by price descending)
</li>
</ul>
<p>
  Solr provides a <a href="http://localhost:8983/solr/admin/form.jsp">query form</a> within the web admin interface that allows setting the various request parameters and is useful when trying out or debugging queries.
</p>
<a href="/">
  <a href="/">
</h3>Sorting</h3>
<p>
  Solr provides a simple method to sort on one or more indexed fields.
  Use the 'sort' parameter to specify "field direction" pairs...
</p>
<ul>
<li>
<a href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=price+desc">q=video&amp;sort=price desc</a>

```



```

</li>

<li>
<a
    href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=price+asc"
    &gt;q=video&amp;sort=price asc</a>
</li>

<li>
<a
    href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=inStock+asc,
    price+desc"&gt;q=video&amp;sort=inStock asc, price desc</a>
</li>

</ul>
<p>
    "score" can also be used as a field name when specifying a sort...
    </p>
<ul>

<li>
<a
    href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=score+desc"
    &gt;q=video&amp;sort=score desc</a>
</li>

<li>
<a
    href="http://localhost:8983/solr/select/?indent=on&amp;q=video&amp;sort=inStock+asc,
    score+desc"&gt;q=video&amp;sort=inStock asc, score desc</a>
</li>

</ul>
<p>
    If no sort is specified, the default is score desc, the same as in the Lucene search APIs.
    </p>

<a href="/"&gt;
    <a
        href="/"&gt;
</h2>Text Analysis</h2>

<p>
    Text fields are typically indexed by breaking the field into words and applying various transformations
    such as
        lowercasing, removing plurals, or stemming to increase relevancy. The same text transformations are
    normally
        applied to any queries in order to match what is indexed.
    </p>
<p>Example queries demonstrating relevancy improving transformations:</p>
<ul>

<li>A search for
    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=power-shot&amp;fl=name"&gt;
    power-shot</a>
        matches PowerShot, and
    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=adata&amp;fl=name"&gt;
    adata</a>
        matches A-DATA due to the use of WordDelimiterFilter and LowerCaseFilter.
    </li>

<li>A search for
    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=name:printers&amp;fl=name"
    &gt;name:printers</a>
        matches Printer, and

```

```

    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=features:recharging&amp;
fl=name,features">features:recharging</a>
        matches Rechargeable due to stemming with the EnglishPorterFilter.
    </li>

<li>A search for
    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=%221gigabyte%22&amp;
fl=name">"1 gigabyte"</a>
        matches things with GB, and
    <a
        href="http://localhost:8983/solr/select/?indent=on&amp;q=pixima&amp;fl=name">
pixima</a>
        matches Pixma due to use of a SynonymFilter.
    </li>

</ul>
<p>
    The
    <a
        href="http://wiki.apache.org/solr/SchemaXml">schema</a> defines
        the fields in the index and what type of analysis is applied to them. The current schema your server is
        using
        may be accessed via the [SCHEMA] link on the
    <a
        href="http://localhost:8983/solr/admin/">admin</a> page.
    </p>
<p>A full description of the analysis components, Analyzers, Tokenizers, and TokenFilters
    available for use is
    <a href="http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters">here</a>.
</p>
<a href="">
    <a href="">
</h3>Analysis Debugging</h3>
<p>There is a handy
    <a href="http://localhost:8983/solr/admin/analysis.jsp">analysis</a>
        debugging page where you can see how a text value is broken down into words,
        and shows the resulting tokens after they pass through each filter in the chain.
    </p>
</p>

<a
        href="http://localhost:8983/solr/admin/analysis.jsp?name=name&amp;val=Canon+PowerShot+SD500"
    >This</a>
        shows how "Canon PowerShot SD500" would be indexed as a value in the name field. Each row of
        the table shows the resulting tokens after having passed through the next TokenFilter in the Analyzer for
        the name field.
        Notice how both powershot and power, shot are indexed. Tokens generated at the same position
        are shown in the same column, in this case shot and powershot.
    </p>
<p>Selecting
    <a href="http://localhost:8983/solr/admin/analysis.jsp?name=name&amp;verbose=on&amp;
val=Canon+PowerShot+SD500">verbose output</a>
        will show more details, such as the name of each analyzer component in the chain, token positions, and the
        start and end positions
        of the token in the original text.
    </p>
<p>Selecting
    <a href="http://localhost:8983/solr/admin/analysis.jsp?name=name&amp;highlight=on&amp;
val=Canon+PowerShot+SD500&amp;qval=power-shot">highlight matches</a>
        when both index and query values are provided will take the resulting terms from the query value and
        highlight
        all matches in the index value analysis.
    </p>
</p>
<a
        href="http://localhost:8983/solr/admin/analysis.jsp?name=text&amp;highlight=on&amp;
val=Four+score+and+seven+years+ago+our+fathers+brought+forth+on+this+continent+a+new+nation%
2C+conceived+in+liberty+and+dedicated+to+the+proposition+that+all+men+are+created+equal.+&amp;
qval=liberties+and+equality">Here</a>

```

is an example of stemming and stop-words at work.
</p>

<h2>Conclusion</h2>

<p>

Congratulations! You successfully ran a small Solr instance, added some documents, and made changes to the index. You learned about queries, text analysis, and the Solr admin interface. You're ready to start using Solr on your own project! Continue on with the following steps:

</p>

Subscribe to the Solr mailing lists!

Make a copy of the Solr example directory as a template for your project.

Customize the schema and other config in solr/conf/ to meet your needs.

<p>

Have Fun, and we'll see you on the Solr mailing lists!

</p>

Copyright ©

2007 The Apache Software Foundation.

</body>

</html>

</str>

</response>