UpdateRichDocuments

Updating a Solr Index with Rich Documents such as PDF and MS Office

⚠️ NOTE: This page covers the RichDocumentHandler as created by Eric Pugh and Chris Harris. Solr's Tika integration, which will replace the RichDocumentHandler is described at ExtractingRequestHandler. This page is being preserved here for those users who currently use the RichDocumentHandler. ⚠️

⚠️ NOTE 2 from Eric: As the ExtractingRequestHandler is completed, users of RichDocumentHandler should move to the native Solr solution.

⚠️ DEPRECATED. This code will not be committed to Solr.

Solr has an extensible DocumentHandler architecture that allows you to feed it XML and CSV documents. There is now a patch file available as part of SOLR-284 that adds support for parsing rich binary formats.

This page talks about how to get started using this patch. If you like it, please vote for it on the JIRA issue tracker so we can get it added to the Solr codebase!

Any additions, enhancements, or questions, please email me at epugh at opensourceconnections.com, or just update the JIRA issue!

Requirements

Solr 1.3 is the current development version, and what the code was developed against. The Rich Document request handler needs to be configured in solrconfig.xml This should already be present in the example solrconfig.xml if you follow the Install directions

```
<!-- Rich document update handler, loaded on demand -->
<requestHandler name="/update/rich" class="solr.RichDocumentRequestHandler" startup="lazy">
</requestHandler>
```

How to Install

1) You need a couple patch files and zips of source and test code that are attached to the JIRA issue at https://issues.apache.org/jira/browse/SOLR-284.

2) Download the rich.patch, libs.zip, and test-files.zip files. If there are duplicates of any of these files, download the latest one; you should be able to tell which one is the latest because the other ones should be grayed out.

3) Unzip the libs.zip into SOLR_HOME/lib. These are the jar's required for parsing the rich documents, using PDFBox and POI.

4) Unzip the test-files.zip into SOLR_HOME/src/test/test-files/. These are various test files for running the included unit tests.

5) Apply the rich.patch to your source. Rich.patch has tweaks that add the solr.RichDocumentRequestHandler to your solrconfig.xml files. It also adds the code to support your new update handler. HowToContribute provides more information on how to apply a patch. (See "Working With Patches".)

6) Run `ant test` to verify everything is working!

Methods of uploading Binary records

Binary records may be uploaded to Solr by sending the data to the /solr/update/rich URL. All of the normal methods for uploading content are supported.

Example

These examples assume you have run `ant example` to generate the example app first and have started Solr up using `java -jar start.jar`.

There is a sample Word file at `src/test/test-files/complex.doc` that may be used to add a Word document to the Solr example server.

Example of using HTTP-POST to send the PDF data over the network to the Solr server in browser:

```
curl -F "stream.file=@simple.pdf" -F "stream.type=pdf" -F "id=102" -F "stream.fieldname=name" -F "commit=true"
http://localhost:8983/solr/update/rich
```

Uploading a binary file can be more efficient than sending it over the network via HTTP. Remote streaming must be enabled for this method to work. See the following line in `solrconfig.xml`, change it to `enableRemoteStreaming="true"`, and restart Solr.

```
  <requestParsers enableRemoteStreaming="false" multipartUploadLimitInKB="2048" />
```

The following request will cause Solr to directly read the input file:

```
URL in browser to add document
http://localhost:8983/solr/update/rich?stream.type=doc&stream.file=SOLR_HOME/src/test/test-files/complex.
doc&fieldnames=id&id=101&stream.fieldname=name&commit=true
#NOTE: The full path, or a path relative to the CWD of the running solr server must be used.

URL in browser to get results
http://localhost:8983/solr/select/?q=id:101&indent=on
```

Parameters

stream.type

Specifies what format the document is. It may be pdf, doc, ppt, or xls.

Example: `stream.type=ppt`

stream.fieldname

The name of the field defined in schema.xml to store the contents of the file in.

Example: `stream.fieldname=text`

fieldnames

A list of metadata fields to also index with this document.

example: `fieldnames=author,subject&author=Bill&subject=fiction`

commit

Commit changes after all records in this request have been indexed. The default is `commit=false` to avoid the potential performance impact of frequent commits.

overwrite

If `true` (the default), overwrite documents based on the uniqueKey field declared in the solr schema.

Disadvantages

There is no way to provide document or field index-time boosts, however many indicies do not utilize that feature.