IndexReplace

Index Replace

The **index-replace** plugin is an indexing filter that allows regexp replace manipulation of metadata fields. The use cases would include adjusting the Nutch document field set and structure to conform to a field set used by a target core that was different than the default fieldset used by Nutch. With this plugin you can modify the structure of existing fields and copy modified fields into new fields. It allows these replacements to be done globally for all parsed pages and for modifications to be done only for certain host or URL patterns.

Related plugins include index-static which allows you to add one or more fields with static values. Also the `indexer-solr` plugin has a config file `solrindex-mapping.xml` which allows you to rename and copy fields. The **index-replace** plugin allows you to make modifications to the fields.

Configuration Example

In `conf/nutch-site.xml` add something like:

```
<property>
  <name>index.replace.regexp</name>
  <value>
    id=/file\:/http\:my.site.com/
    url=/file\:/http\:my.site.com/2
  </value>
</property>
```

Also insure that `index-replace` is among the plugins that will be used.

```
<property>
  <name>plugin.includes</name>
  <value>...|index-(basic|anchor|metadata|static|replace)|...</value>
</property>
```

Property format

Name: `index.replace.regexp`

The format of the property is a list of regexp replacements, one line per field being modified. Field names would be one of those from IndexStructure.

The field name precedes the equal sign. The first character after the equal sign signifies the delimiter for the regexp, the replacement value and the optional flags.

Replacement Sequence

The replacements will happen in the order listed. If a field needs multiple replacement operations it may be listed more than once.

RegExp Format

The regexp and the optional flags should correspond to Java's Pattern.compile.

Patterns are compiled when the plugin is initialized for efficiency.

Replacement Format

The replacement value should correspond to Java Matcher

Flags

The flags is an integer sum of the flag values defined in Java constant values (Sec: java.util.regex.Pattern)

Creating New Fields

If you express the fieldname as `fldname1:fldname2=[replacement]`, then the replacer will create a new field (fldname2) from the source field (fldname1). The source field remains unmodified. This is an alternative to `solrindex-mapping.xml` which is only able to copy fields verbatim.

Multi-valued Fields

If a field has multiple values, the replacement will be applied to each value in turn.

Non-string Datatypes

Replacement is possible only on `String` field datatypes. If the field you name in the property is not a `String` datatype, it will be silently ignored.

Host and URL specific replacements

If the replacements should apply only to specific pages, then add a sequence like

```
  hostmatch=hostmatchpattern
  fld1=/regexp/replace/flags
  fld2=/regexp/replace/flags
```

or

```
  urlmatch=urlmatchpattern
  fld1=/regexp/replace/flags
  fld2=/regexp/replace/flags
```

When using Host and URL replacements, all replacements preceding the first `hostmatch=` or `urlmatch=` will apply to all parsed pages. Replacements following a `hostmatch` or `urlmatch` will be applied to pages which match the host or url field (up to the next `hostmatch` or `urlmatch` line). `hostmatch` and `urlmatch` patterns must be unique in this property.

Plugin order

In most cases you will want this plugin to run last among the index filters, just before you run your indexer plugin.

Testing your match patterns

Online Regexp testers can help get the basics of your pattern working.

If your property does not parse correctly, you can discover this by looking in the `hadoop.log` after doing a trial indexing run. Its important to test your patterns because the `index-replace` plugin will mark any entry in the replacement list as *invalid* which does not parse into a proper regexp operation. Invalid replacement operations are simply ignored.

To test in Nutch

- Prepare a test HTML file with the field contents you want to test.
- Place this in a directory accessible to nutch.
- Use the file:/// syntax to list the test file(s) in a test/urls seed list.
- See the nutch faq index my local file system for conf settings you will need. (Note the `urlmatch=` and `hostmatch=` patterns in your configuration may not conform to your test file url; This test approach confirms only how your global matches behave, unless your `urlmatch=` and `hostmatch=` patterns also match the file: URL pattern for your test file)

Run..

```
  bin/nutch inject crawl/crawldb test
  bin/nutch generate crawl/crawldb crawl/segments
  bin/nutch fetch crawl/segments/[segment]
  bin/nutch parse crawl/segments/[segment]
  bin/nutch invertlinks crawl/linkdb -dir crawl/segments
    ...index your document, for example with SOLR...
  bin/nutch solrindex http://localhost:8983/solr crawl/crawldb/ -linkdb crawl/linkdb/ crawl/segement[segment] -
filter -normalize
```

Inspect `hadoop.log` for info about pattern parsing and compilation..

```
  grep replace logs/hadoop.log
```

To inspect your index with the solr admin panel browse to...

```
http://localhost:8983/solr/#/
```

And if you want to adjust your patterns in `nutch-site.xml` and re-test, you only need to repeat the solrindex step above and review the result.