



## BLOG

Welcome to our **BLOG**.

Get the latest scoop or just **follow** and **discuss** our latest **studies**.

Mar  
14

## Precise data extraction with Apache Nutch

By Emir DIZDAREVIĆ // [Lucene Search](#) // [No Comments](#)

Nutch's HtmlParser parses the whole page and returns parsed text, outlinks and additional meta data. Some parts of this are really useful like the outlinks but that's basically it. The problem is that the parsed text is too general for the purpose of precise data extraction. Fortunately the HtmlParser provides us a mechanism (extension point) to attach an HtmlParserFilter to it.

We developed a plugin, which consists of HtmlParserFilter and IndexingFilter extensions, which provides a mechanism to fetch and index the desired data from a web page through use of XPath 1.0. The name of the plugin is filter-xpath plugin.

Using this plugin we are now able to extract the desired data from web site with known structure. Unfortunately the plugin is an extension of the HtmlParserFilter extension point which is hardly coupled to the HtmlParser, hence plugin won't work without the HtmlParser. The HtmlParser generates its own metadata (host, site, url, content, title, cache and timestamp) which will be indexed too. One way to control this is by not including IndexFilter plugins which depend on the metadata data to generate the indexing data (NutchDocument). The other way is to change the SOLR index mappings in the solrindex-mapping.xml file (maps NutchDocument fields to SolrInputDocument field). That way we will index only the fields we want.

The next problem arises when it comes to indexing. We want that Nutch fetches every page on the site but we don't want to index them all. If we use the UrlRegexFilter to control this we will lose the indirect links which we also want to index and add to our URL DB. To address this problem we developed another plugin which is an extension of the IndexingFilter extension point which is called index-omit plugin. Using this plugin we are able to omit indexing on the pages we don't need.

### Filter-xpath plugin configuration

Like every plugin it needs to be placed inside the NUTCH\_HOME/plugin folder. You can download our plugin from here: <https://github.com/ATLANTBH/nutch-plugins>.

Our plugin supports three ways to identify a page: via url, via content or a combination of both:

- Via url: A simple regex/url matching mechanism. Only if the url matches the regex the plugin will extract the data.
- Via content: A bit more complex xpath/regex matching mechanism. The content is extracted through a xpath we specify and then matched against a regex we also specify. If it's a match the data is extracted. It's also possible to fine tune the behavior of the xpath data extraction (ie. trim, concat etc.).
- Combination: Self explanatory.

**Notice: Everything else is described in the configuration file in more detail.**

The location of the main configuration file of our filter needs to be added to the nutch-site.xml:

```
<property>
  <name>filter.xpath.file</name>
  <value>xpathfilter-conf.xml</value>
  <description>
    Configuration file of the xml/html parser/indexing filter.
    Definition of the xpath - field mappings.
  </description>
</property>
```

**Notice: all paths are relative to the Apache Nutch conf folder.**

The format of the main configuration file xpathfilter-conf.xml is as following:

```
<?xml version="1.0" encoding="UTF-8"?>
```

### OUR SERVICES

---

 AUTOMATED QUALITY ASSURANCE

---

 BIG DATA (HADOOP)

---

 LUCENE SEARCH

---

 MOBILE DEVELOPMENT

### BLOG CATEGORIES

---

 Automated QA/ Testing  
(5)

---

 BI and Analytics  
(2)

---

 Big Data  
(8)

---

 Education  
(1)

---

 Hadoop as service  
(3)

---

 Lucene Search  
(6)

---

 Mobile Development  
(1)

---

 Regression and Monitoring  
(1)

---

 Startups  
(2)

---

 Testing Extensions  
(2)

---

 Testing Tools  
(4)

### RECENT POSTS

---

 Book Review: "Web Crawling and Data Mining with Apache Nutch"

---

 Code of Life – Computer Science Education Week 2013

---

 Startup Weekend Sarajevo 2013 Review

---

 Developing Cross-Platform Mobile Applications Using Phonegap

---

 Alternating Nutch flow to store fetched data to CSV

---

 Precise data extraction with Apache Nutch

---

 Apache Nutch Overview

---

 Bad vs Good Search Experience

---

 Regression Suite

---

 Apache Solr – Slow queries and frequent

```

<xpathFilterConfiguration>
  <!-- List of properties -->
  <!-- (O)pageUrlFilterRegex: If supplied the xpath-field mapping will be evaluated only on
the urls that match the regex (DEFAULT: ALL URLS WILL BE INCLUDED) -->
  <!-- (O)pageContentFilterXPath: Evaluate the pageContentFilterRegex on the data of this
xpath expression -->
  <!-- (O)concatPageContentFilterXPathData: If the xpath return multiple nodes defines if
they should be concatenated (DEFAULT: FALSE) -->
  <!-- ONLY IF pageContentFilterXPath DEFINED -->
  <!-- (R)pageContentFilterRegex: If supplied the xpath-field mapping will be evaluated on
the pageContentFilterXPath retrieved data that matches the regex (DEFAULT: NO DEFAULT) -->
  <!-- (O)trimPageContentFilterXPathData: Defines if the retrieved content xpath data
should be trimmed (whitespace removal) (DEFAULT: TRUE) -->
  <!-- ONLY IF pageContentFilterXPath AND concatPageContentFilterXPathData DEFINED -->
  <!-- (O)concatPageContentFilterXPathDataDelimiter: The delimiter by which the nodes will
be joined (DEFAULT: EMPTY STRING) -->
  <xpathIndexerProperties pageUrlFilterRegex="^http://([a-z0-9]*\.)*/"
pageContentFilterXPath="" pageContentFilterRegex="" trimPageContentFilterXPathData="true"
concatPageContentFilterXPathData="true" concatPageContentFilterXPathDataDelimiter=" ">
    <!-- List of properties -->
    <!-- (R)name: The name of the field that should be indexed with solr -->
    <!-- (R)xpath: The xpath expression to fetch data from xml/html -->
    <!-- (R)type: The data type the field should be indexed with (BOOLEAN, STRING,
FLOAT, DOUBLE, INTEGER, LONG, DATE) -->
    <!-- (O)trimXPathData: Defines if the retrieved xpath data should be trimmed
(whitespace removal) (DEFAULT: TRUE) -->
    <!-- (O)concat: If the xpath return multiple nodes defines if they should be
concatenated (DEFAULT: FALSE) -->
    <!-- ONLY IF type=DATE -->
    <!-- (O)dateFormat: If the type is DATE the format of the date (DEFAULT:
dd.MM.yyyy) -->
    <!-- ONLY IF concat DEFINED -->
    <!-- (O)concatDelimiter: The delimiter by which the nodes will be joined
(DEFAULT: EMPTY STRING) -->
    <field name="title" xpath="//some_xpath" type="STRING" concat="false"
concatDelimiter=" " trimXPathData="false"/>
    <field name="date" xpath="//some_xpath" type="DATE" dateFormat="dd.MM.yyyy" />
  </xpathIndexerProperties>
</xpathFilterConfiguration>

```

The final step is to tell Nutch to use our filter. This can be done through the plugin.includes property inside nutch-site.xml:

```

<property>
  <name>plugin.includes</name>
  <value>protocol-http|urlfilter-regex|parse-
(xmlhtml|html|tika)|filter-xpath|index-
(basic|anchor)|scoring-opic|urlnormalizer-
(pass|regex|basic)
  </value>
  <description>Regular expression naming plugin directory names to
include. Any plugin not matching this expression is excluded.
In any case you need at least include the nutch-extensionpoints
plugin. By
default Nutch includes crawling just HTML and plain text via
HTTP, and basic indexing and search plugins. In order to use
HTTPS please enable protocol-httpclient, but be aware of
possible intermittent problems with the
underlying commons-httpclient library.
  </description>
</property>

```

### Index-omit plugin configuration

Like every plugin it needs to be placed inside the NUTCH\_HOME/plugin folder. You can download our plugin from here: <https://github.com/ATLANTBH/nutch-plugins>.

Index-omit filter supports two ways of operation (BLACKLIST and WHITELIST) and a target attribute which can be:

- URL: The url will be matched against a regex. If it's a match the document will be indexed.
- META\_FIELD\_PRESENT: If the meta field is present (The meta field is a field in the metadata structure that is passed to the IndexingFilter. Usually it has the same name as the field we want to index) the document will be indexed.

The location of the main configuration file of our filter needs to be added to the nutch-site.xml:

```

<property>
  <name>filter.index.omit.file</name>
  <value>omit-indexfilter-conf.xml</value>
  <description>
    Configuration file of the omit indexing filter.
    Black/white list of permitted indexing.
  </description>
</property>

```

terms

## CONTACT US

Our sales and support staff are standing by

+387 33 716-550

[Contact Us](#)

Come and visit us at

Zmaja od Bosne bb

TC Robot, ulaz JI/5

71000 Sarajevo

Bosnia and Herzegovina

Keep in touch



[-- Contact Us](#)

```
</description>
</property>
```

The format of the main configuration file omit-indexfilter-conf.xml is as following:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- List of properties -->
<!-- (R)filteringType: Defines the type of filtering. If it's going to be a whitelist or a blacklist filtering
(WHITELIST, BLACKLIST) -->
<omitIndexingFilterConfiguration filteringType="WHITELIST">
  <!-- List of properties -->
  <!-- (R)target: Defines the target against which the regex should be matched (URL,
META_FIELD_PRESENT) -->
  <!-- ONLY IF target=URL -->
  <!-- (R)regex: Defines the regex that should be used for matching -->
  <!-- ONLY IF target=META_FIELD_PRESENT -->
  <!-- (R)name: Defines the name of the meta field that must be contained -->
  <omitIndexingFilterConfigurationEntry target="URL" regex="http://([a-z0-9]*\.)*nutch.apache.org/" />
</omitIndexingFilterConfiguration>
```

The final step is to tell Nutch to use our filter. This can be done with the plugin.includes property inside nutch-site.xml:

```
<property>
  <name>plugin.includes</name>
  <value>protocol-http|urlfilter-regex|parse-(html|tika)||index-
(omit|basic|anchor)|scoring-opic|urlnormalizer-(pass|regex|basic)</value>
  <description>Regular expression naming plugin directory names to
  include. Any plugin not matching this expression is excluded.
  In any case you need at least include the nutch-extensionpoints plugin. By
  default Nutch includes crawling just HTML and plain text via HTTP,
  and basic indexing and search plugins. In order to use HTTPS please enable
  protocol-httpclient, but be aware of possible intermittent problems with the
  underlying commons-httpclient library.
</description>
</property>
```

### First step

The first thing we need to do is to configure the filter-xpath and the index-omit plugin.

#### Xpath-filter configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<xpathFilterConfiguration>
  <!--Filter configured to accept only pages from domain atlantbh.com that contain the word
  BLOG in the XPath-->
  <xpathIndexerProperties pageUrlFilterRegex="http://([a-z0-9]*\.)*atlantbh.com/"
  pageContentFilterXPath="//article/div[@id='headerimg2']/div[@id='hText2']/span/span[1]"
  pageContentFilterRegex="BLOG">
    <field name="articleTitle"
    XPath="//div[@id='colLeftInner']/div[@class='blogPost']/h2/a" type="STRING" />
    <field name="articleAuthor"
    XPath="//div[@id='colLeftInner']/div[@class='blogPost']/div[@class='meta']/span"
    type="STRING" />
    <field name="articleContent" XPath="//div[@id='colLeftInner']/div[@class='blogPost']/p|
    //div[@id='colLeftInner']/div[@class='blogPost']/ol|
    //div[@id='colLeftInner']/div[@class='blogPost']/ul|
    //div[@id='colLeftInner']/div[@class='blogPost']/h2|
    //div[@id='colLeftInner']/div[@class='blogPost']/h3|
    //div[@id='colLeftInner']/div[@class='blogPost']/div[@id='LC32']"
    type="STRING" concat="true" concatDelimiter=" " />
  </xpathIndexerProperties>
</xpathFilterConfiguration>
```

#### Index-omit plugin configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Filter configured to index only pages that have been previously accepted by the XPath-filter
hence contain the articleTitle meta field -->
<omitIndexingFilterConfiguration filteringType="WHITELIST">
  <omitIndexingFilterConfigurationEntry target="META_FIELD_PRESENT" name="articleTitle" />
</omitIndexingFilterConfiguration>
```

### Second step

The next step is to configure nutch-site.xml.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Put site-specific property overrides in this file. -->
<configuration>
```

```

...
<property>
  <name>indexer.score.power</name>
  <value>0</value>
  <description>...</description>
</property>
<property>
  <name>plugin.includes</name>
  <value>protocol-http|urlfilter-regex|parse-(html|tika) |filter-xpath|index-
(omit|basic|anchor) |scoring-opic|urlnormalizer-(pass|regex|basic)</value>
  <description>...</description>
</property>
<property>
  <name>filter.index.omit.file</name>
  <value>omit-indexfilter-conf.xml</value>
  <description>
    Configuration file of the omit indexing filter.
    Black/white list of permitted indexing.
  </description>
</property>
<property>
  <name>filter.xpath.file</name>
  <value>xpathfilter-conf.xml</value>
  <description>
    Configuration file of the xml/html parser/indexing filter.
    Definition of the xpath - field mappings.
  </description>
</property>
...
</configuration>

```

### Third step

The next step is to configure the `RegexURLFilter` to constrain nutch to crawl not outside our domain. To accomplish that the only thing we need to do is a small modification in the `regex-urlfilter.txt` file which is located inside the `conf` folder:

replace this:

```
# accept anything else
+.
```

with this:

```
# accept anything else
#+.
+^http://([a-z0-9]*\.)*atlantbh.com/
```

### Fourth step

Now we need to tell Nutch which `NutchDocument` fields to map into which `SolrInputDocument` fields. This is configured inside the `solrindex-mapping.xml` which is located inside the `conf` folder.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping>
  <fields>

    <!-- These fields are produced by our XPathFilter -->
    <field dest="articleTitle" source="articleTitle"/>
    <field dest="articleAuthor" source="articleAuthor"/>
    <field dest="articleContent" source="articleContent"/>

    <!-- These fields are produced by the HtmlParser -->
    <field dest="content" source="content"/>
    <field dest="site" source="site"/>
    <field dest="title" source="title"/>
    <field dest="host" source="host"/>
    <field dest="segment" source="segment"/>
    <field dest="boost" source="boost"/>
    <field dest="digest" source="digest"/>
    <field dest="tstamp" source="tstamp"/>
    <field dest="id" source="url"/>
    <copyField source="url" dest="url"/>

  </fields>
  <uniqueKey>id</uniqueKey>
</mapping>

```

### Fifth step

Nutch is now configured and ready to go. The only thing left to do is to configure the Apache Solr `schema.xml`. Just place the following `schema.xml` inside the Solr `conf` folder:

```

<?xml version="1.0" encoding="UTF-8" ?>
<schema name="nutch" version="1.4">
  <types>
    <fieldType name="string" class="solr.StrField" sortMissingLast="true" omitNorms="true"/>
    <fieldType name="long" class="solr.TrieLongField" precisionStep="0" omitNorms="true"
      positionIncrementGap="0"/>
    <fieldType name="int" class="solr.TrieIntField" precisionStep="0" omitNorms="true"
      positionIncrementGap="0"/>
    <fieldType name="float" class="solr.TrieFloatField" precisionStep="0" omitNorms="true"
      positionIncrementGap="0"/>
    <fieldType name="date" class="solr.TrieDateField" precisionStep="0" omitNorms="true"
      positionIncrementGap="0"/>
    <fieldType name="text" class="solr.TextField" positionIncrementGap="100">
      <analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.LowerCaseFilterFactory"/>
        <filter class="solr.WordDelimiterFilterFactory"
          splitOnCaseChange="1"
          splitOnNumerics="1"
          generateWordParts="1"
          generateNumberParts="1"
          catenateWords="1"
          catenateNumbers="1"
          catenateAll="1"
          preserveOriginal="1"
        />
        <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
      </analyzer>
    </fieldType>
  </types>
  <fields>
    <field name="id" type="string" stored="true" indexed="true"/>

    <!-- xpath filter fields -->
    <field name="articleTitle" type="text" stored="true" indexed="true" />
    <field name="articleAuthor" type="text" stored="true" indexed="true" />
    <field name="articleContent" type="text" stored="true" indexed="true" />

    <!-- core fields -->
    <field name="segment" type="string" stored="true" indexed="false"/>
    <field name="digest" type="string" stored="true" indexed="false"/>
    <field name="boost" type="float" stored="true" indexed="false"/>

    <!-- fields for index-basic plugin -->
    <field name="host" type="url" stored="false" indexed="true"/>
    <field name="site" type="string" stored="false" indexed="true"/>
    <field name="url" type="url" stored="true" indexed="true"
      required="true"/>
    <field name="content" type="text" stored="false" indexed="true"/>
    <field name="title" type="text" stored="true" indexed="true"/>
    <field name="cache" type="string" stored="true" indexed="false"/>
    <field name="tstamp" type="date" stored="true" indexed="false"/>

    <!-- fields for index-anchor plugin -->
    <field name="anchor" type="string" stored="true" indexed="true"
      multiValued="true"/>

    <!-- fields for index-more plugin -->
    <field name="type" type="string" stored="true" indexed="true"
      multiValued="true"/>
    <field name="contentLength" type="long" stored="true"
      indexed="false"/>
    <field name="lastModified" type="date" stored="true"
      indexed="false"/>
    <field name="date" type="date" stored="true" indexed="true"/>

    <!-- fields for languageidentifier plugin -->
    <field name="lang" type="string" stored="true" indexed="true"/>
    <!-- fields for subcollection plugin -->
    <field name="subcollection" type="string" stored="true"
      indexed="true" multiValued="true"/>

    <!-- fields for feed plugin (tag is also used by microformats-reltag)-->
    <field name="author" type="string" stored="true" indexed="true"/>
    <field name="tag" type="string" stored="true" indexed="true" multiValued="true"/>
    <field name="feed" type="string" stored="true" indexed="true"/>
    <field name="publishedDate" type="date" stored="true"
      indexed="true"/>
    <field name="updatedAt" type="date" stored="true"
      indexed="true"/>

    <!-- fields for creativecommons plugin -->
    <field name="cc" type="string" stored="true" indexed="true"
      multiValued="true"/>
  </fields>
</schema>

```

```
</fields>
<uniqueKey>id</uniqueKey>
<defaultSearchField>articleContent</defaultSearchField>
<solrQueryParser defaultOperator="OR"/>
</schema>
```

### Sixth step

Now everything is configured and ready to go. The Apache Solr server must be up and running. To start the crawling we run our script for Apache Nutch Workflow Automation described in the previous article with the following command:


```
gfpw 100 5
```

After the crawling is done we just need to tell Nutch to index the data with the following command:

```
bin/nutch solrindex http://127.0.0.1:8983/solr/ crawlddb
crawlddb/segments/*
```

Now if we open our index we will see that only the BLOG articles have been indexed along with the data we specified through XPath.


---

Like Share 14 people like this. Be the first of your friends.  +6 Suosittele tätä Googlessa


Tweet

4

12 comments [Add a comment](#)



**Julien Nioche** · Bristol, United Kingdom  
> Unfortunately the plugin is an extension of the `HtmlParserFilter` extension point which is hardly > coupled to the `HtmlParser`, hence plugin won't work without the `HtmlParser`.  
  
The `HtmlParserFilters` are called from the `parse-tika` plugin too so you can have your filters running on the XHTML representation of a binary type like pdf.  
[Reply](#) · [Like](#) · [Follow Post](#) · April 3, 2012 at 9:18pm



**Rogerio Pereira Araujo** · Top Commenter · Sócio Proprietário at BMobile Soluções em Informática  
It would be nice have a version compatible with Nutch 2.x.  
[Reply](#) · [Like](#) · [Follow Post](#) · October 23, 2012 at 12:09am

[View 8 more](#)

Facebook social plugin



© Copyright 1996 - 2014, Atlantbh d.o.o., © All Rights Reserved

**Career:**  
phone: +387 33 716-557  
email: [jobs@atlantbh.com](mailto:jobs@atlantbh.com)

**Contact:**  
phone: +387 33 716-550  
fax: +387 33 716-551  
email: [contact@atlantbh.com](mailto:contact@atlantbh.com)

**Follow:**

Like 

662

  
[Atlantbh](#)

[Follow](#)