FirstSpirit™
Unlock Your Content

FirstSpirit™ Exalead Integration

FirstSpirit Version 5.0

Version	**1.11**
Status	**RELEASED**
Datum	**2013-08-29**
Abteilung	Product Management
Copyright	2013 e-Spirit AG
Dateiname	EXAL4xEN_FirstSpirit_EnterpriseSearch

e-Spirit AG
Barcelonaweg 14
44269 Dortmund | Germany

T +49 231 . 477 77-0
F +49 231 . 477 77-499

info@e-spirit.com
www.e-spirit.com

e-Spirit

Inhaltsverzeichnis

1 Introduction

This document describes the different integration options between the FirstSpirit content management system and the enterprise search engine technology EXALEAD ONE:ENTERPRISE (hereafter referred to as "Exalead" for short).

Both products can generally be used independent of each other. However, in combination, both modern technologies can be connected to each other to produce a high level of functional utility for the website user, who would like to find specific information within FirstSpirit content in the easiest way possible.

The following five aspects are described:

- Chapter 2: Architecture of the integration solution between FirstSpirit and Exalead
- Chapter 3: Description of the FirstSpirit runtime component for displaying Exalead search results (on the web server side)
- Chapter 4: Explanation of the possibilities of assigning permissions to documents and thus of providing a personalised search
- Chapter 5: Use of the Push-API for transferring individual documents into the index
- Chapter 6: Use of the FirstSpirit module "Exalead Content-Update“ for transferring documents into the index within the scope of a deployment

This documentation refers to Version 1.7.1 of the FirstSpirit Exalead Live and FirstSpirit Exalead Content-Update modules!

2 Architecture and Components

The following components are relevant in the specified integration scenario:

- FirstSpirit Server (Version >= 4.2.438), incl. the modules:
    - FirstSpirit Exalead Live
    - FirstSpirit Personalisation (optional)
    - FirstSpirit Exalead Content-Update (optional)
- Exalead one:enterprise Server (Version 4.6.0.181 - 4.6.0.297)
- Java Application Server (Servlet Spec. >= 2.4, recommended: Apache Tomcat)
- Java Runtime Environment 1.5 (Version 1.6 is required for use of the PushAPI)

Actual installation of FirstSpirit, Exalead and the application server is not described here. Please refer to the respective installation documents instead.

Several of the listed modules are optional and are therefore not necessary or useful in all application cases. Therefore, different development stages of integration are examined in the following chapters. The options and variants become increasingly complex from step to step; however, functionally they are based on each other.

2.1 Basic architecture

For example, the basic architecture explained here is suitable for internet sites which primarily want to make public, unpersonalised information browsable.

The interaction of the individual components and the data flow between each other is schematically represented in Figure 2-1:

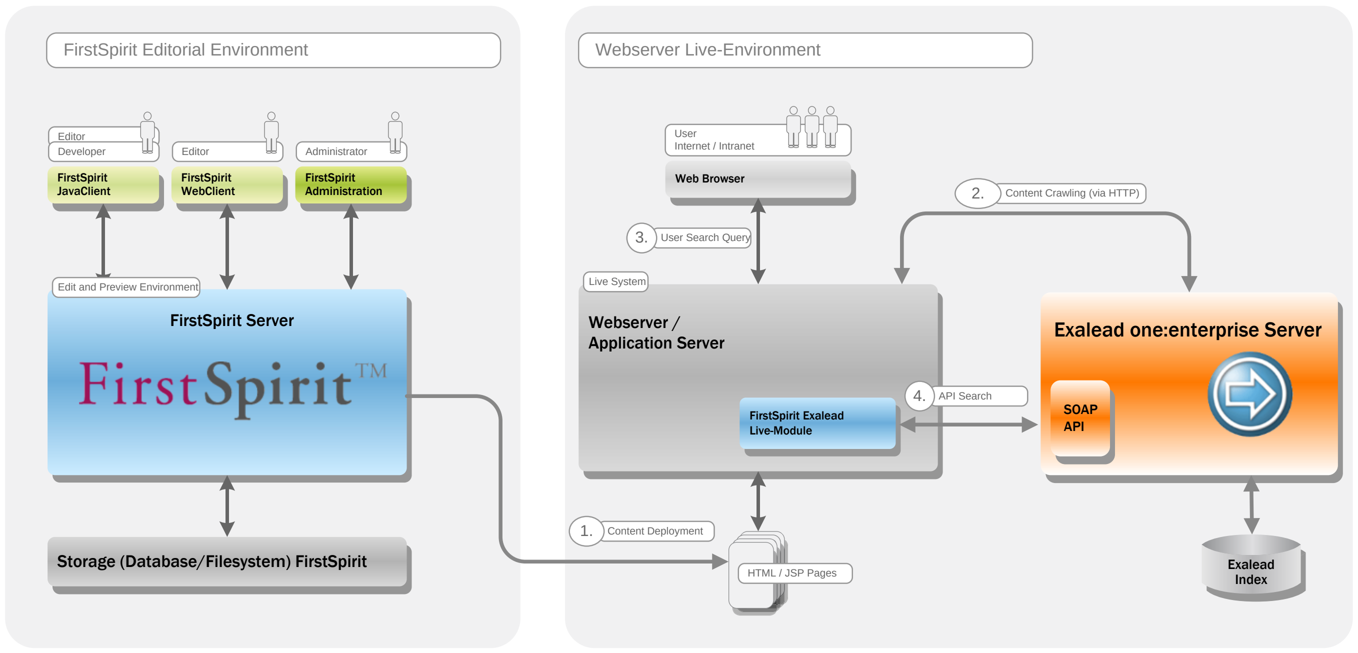

FirstSpirit Editorial Environment
Editor
Developer
Editor
Administrator
FirstSpirit JavaClient
FirstSpirit WebClient
FirstSpirit Administration
Edit and Preview Environment
FirstSpirit Server
FirstSpirit™
Storage (Database/Filesystem) FirstSpirit
Webserver Live-Environment
User Internet / Intranet
Web Browser
2. Content Crawling (via HTTP)
3. User Search Query
Live System
Webserver / Application Server
4. API Search
FirstSpirit Exalead Live-Module
Exalead one:enterprise Server
SOAP API
1. Content Deployment
HTML / JSP Pages
Exalead Index

Figure 2-1: Basic architecture

Workflow description:

- FirstSpirit is used to enter and maintain the internet site content and to transform it into suitable HTML or JSP pages.
- The generated files are then transferred to the application server as part of a deployment process [Step 1].
- As soon as all pages have been transferred to the web server, the Exalead search engine can read in the files as part of a "Crawling" process, analyse them and include them in the index. This process takes place cyclically, e.g. after each deployment of new or amended files [Step 2]. The search index is now filled and can answer search queries.
- The visitor to the internet site can now use a search form to browse through the indexed content [Step 3]. The form passes the form query to the "FirstSpirit Exalead Live Module", which in turn passes the query to the Exalead server. Technically, the "FirstSpirit Exalead Live Module" contains a servlet, which uses the SOAP-API (Simple-Object-Access-Protocol) of the Exalead server.
- After the search query has been processed and answered by the Exalead server, the "FirstSpirit Exalead Live Module" clearly presents the search results on an appropriate page. Technically, the parameters for the output design are assigned via a JSP tag

library.

This basic architecture enables application of the complete Exalead search options to FirstSpirit content, such as

- ✓ Indexing more than 300 formats, incl. Word, Excel, PowerPoint, PDF, RTF, OpenOffice, HTML, XML, Images, audio, video
- ✓ Multi-language support (Unicode)
- ✓ Natural language processing (spelling suggestions, phonetic matching)
- ✓ Automatic classification of search queries and automatic keyword generation
- ✓ Preview thumbnails of the hits
- ✓ Guided navigation through "drilldown" within the results lists

Limitations of the basic architecture:

- no personalised delivery of HTML pages on the basis of permissions
- no personalised delivery of binary documents on the basis of permissions
- no ad-hoc updating of the index in the event of new content

2.2 Architecture with page personalisation

The basic approach of the previous chapter can be enhanced to include the aspect of personalisation for web scenarios in which protected page content is also relevant (e.g. for intranets).

In the following it is assumed that the "FirstSpirit Personalisation" module is used for personalisation of the content on pages of the application server.

Architecturally, the layout is therefore as in Figure 2-2:

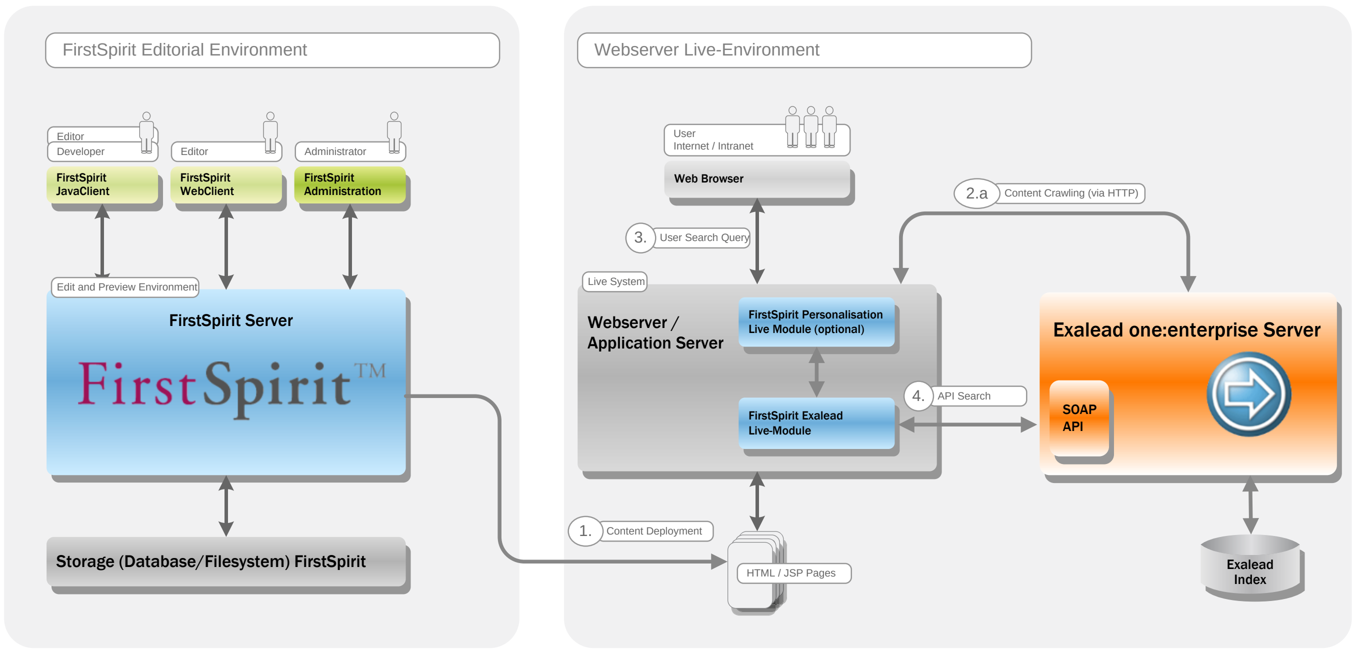

FirstSpirit Editorial Environment
Editor
Developer
Editor
Administrator
FirstSpirit JavaClient
FirstSpirit WebClient
FirstSpirit Administration
Edit and Preview Environment
FirstSpirit Server
FirstSpirit™
Storage (Database/Filesystem) FirstSpirit
Webserver Live-Environment
User Internet / Intranet
Web Browser
2.a Content Crawling (via HTTP)
3. User Search Query
Live System
Webserver / Application Server
FirstSpirit Personalisation Live Module (optional)
FirstSpirit Exalead Live-Module
4. API Search
Exalead one:enterprise Server
SOAP API
1. Content Deployment
HTML / JSP Pages
Exalead Index

Figure 2-2: Personalised search

To a large extent, the layout corresponds to that of the basic architecture; however, coupling of the "FirstSpirit Exalead Live module" and the "FirstSpirit Personalisation module" is added.

Technically, the modules interact so that all search queries passed to the SOAP-API of Exalead are additionally supplemented with the relevant personalisation information of the currently logged in visitor. Specifically, these are:

- Login name of the user
- List of the user's groups

This information is transparently copied from the personalisation module by the "FirstSpirit Exalead Live Module".

The index continues to be developed completely via a crawling approach.

Suitable metatags, which define the personalisation information per page, must be deposited within the HTML pages to enable correct filtering of the HTML pages. Details of this are given in Chapter 4.2. At this point it needs only be noted that the permissions information is injected into the HTML by suitable FirstSpirit templates.

This results in the following options as an enhancement of the basic architecture:

✓ Very easy to implement support for page personalisation by means of simple FirstSpirit template constructs

- ✓ Definition of permission on HTML pages directly by the FirstSpirit editor
- ✓ Personalised, user-specific filtered search results by coupling the user stores of FirstSpirit and Exalead

Limitations of this architecture:

- no personalised delivery of binary documents on the basis of permissions (personalisation information cannot be injected there)
- no ad-hoc updating of the index in the event of new content (generally not possible with the crawling approach)

2.3 Architecture with document personalisation (Push-API)

In order to realise the two aspects:

- Passing of personalisation information to binary documents
- ad-hoc updating of the index,

the crawling approach used to date must be replaced by a second strategy: the Exalead Push-API.

When the Push-API is used, active documents from a source (FirstSpirit) are passed to the search engine (by means of the "Push command"). Apart from the pure document, meta attributes such as personalisation information can also be passed at the same time.

The layout is shown in the diagram in Figure 2-3:

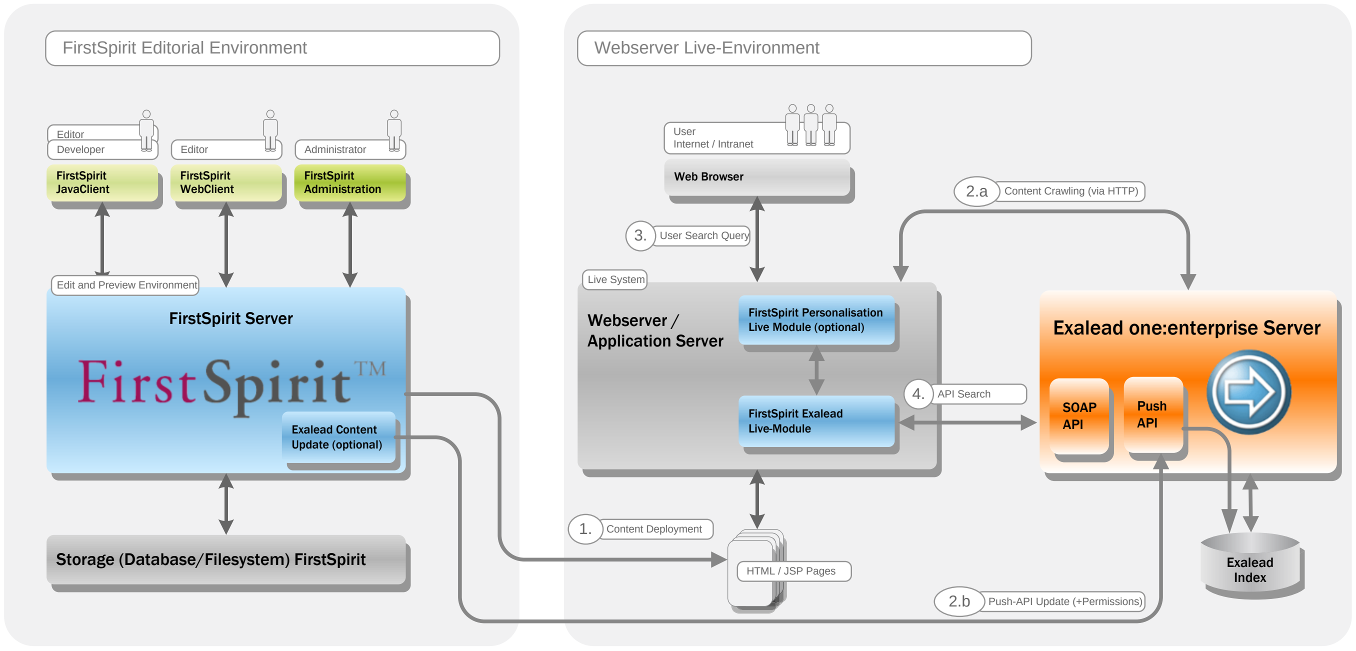

FirstSpirit Editorial Environment
Editor
Developer
Editor
Administrator
FirstSpirit JavaClient
FirstSpirit WebClient
FirstSpirit Administration
Edit and Preview Environment
FirstSpirit Server
FirstSpirit™
Exalead Content Update (optional)
Storage (Database/Filesystem) FirstSpirit
Webserver Live-Environment
User Internet / Intranet
Web Browser
3. User Search Query
2.a Content Crawling (via HTTP)
Live System
Webserver / Application Server
FirstSpirit Personalisation Live Module (optional)
FirstSpirit Exalead Live-Module
4. API Search
Exalead one:enterprise Server
SOAP API
Push API
1. Content Deployment
HTML / JSP Pages
2.b Push-API Update (+Permissions)
Exalead Index

Figure 2-3: Push-API architecture

The "Exalead Content Update" has been added to the FirstSpirit server side. This is capable of directly passing binary documents and HTML pages to the Exalead server by means of Push-API.

To do this, the locally generated files (pages and documents) are read into the FirstSpirit server and are passed to the Exalead Push-API [Step 2.b]. Details of using the Push-API from FirstSpirit are given in Chapter 6.

2.3.1 Crawling vs. Push-API

The two versions of indexing, "Crawling" [Step 2.a] and Push-API [Step 2.b] do not necessarily exclude each other, but instead can also be used in combination. The cases in which this makes sense are examined in the following.

At first glance, sole use of the Push-API only brings advantages:

a) Immediate index updating within the scope of a FirstSpirit deployment

b) Handover of metadata for the personalisation of pages and documents

However, there is one condition under which a combination of crawling (for HTML pages only) and Push-API (for documents only) is necessary.

Whenever the HTML pages contain "indexing relevant" runtime logic, e.g. in the form of JSP, PHP or .NET code, which can only be run within the target application server, it is not possible to do without crawling via the application server.

Example:

A JSP page generated by FirstSpirit not only contains editorial FirstSpirit content, but also an inclusion logic, which integrates and displays message contributions from an external source during the runtime. The content of the messages should be taken into account in the indexing of the pages, so that these indexing terms also lead to a hit during the search.

In such situations, combined use of crawling and Push-API is necessary.

The integration architecture is in any case sufficiently flexible to allow such more complex scenarios to be adequately reproduced too.

3 FirstSpirit Exalead Live Module

3.1 General information

This chapter describes installation and configuration of the "FirstSpirit Exalead Live" module. It is assumed that both FirstSpirit and the Exalead server and a suitable Java Application server have already been set up and configured.

3.1.1 Character coding

The FirstSpirit Exalead Live Module uses UTF-8 throughout as character coding. To ensure uniform use of this character coding, it is necessary to provide several specific details concerning it.

Each HTML page should be assigned a corresponding Meta tag:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

In JSP pages, the page encoding must also be given:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
```

When using forms, ensure that the character coding to be used here is also given:

```
<form ... method="post" accept-charset="UTF-8">
```

Finally, it is also necessary to configure the Apache Tomcat server so that the right character coding is also used when hyperlinks are called. To do this, the `URIEncoding="UTF-8"` parameter must be added to the following section in the `server.xml` of the Apache Tomcat (in the `conf` directory):

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector port="8080" ... URIEncoding="UTF-8"/>
```

3.1.2 Special characters in url path

Despite the usage of UTF-8, special characters within the url path need a special treatment. This applies to both the redirectUrl and errorUrl of the SearchServlet (chapter 3.7.1 SearchServlet) and the redirectUrl and errorUrl of the ContainerTag (chapter 3.10.1 <search:container>).

In case of the SearchServlet the urls have to be passed to the form UTF-8 encoded.

Example: The result page for the search can be accessed under http://www.mydomain.com/search/presseerklärungen/search.jsp

The resultUrl within the form has to look like this:

```
<input type="hidden" name="redirectUrl"
value="/search/presseerkl%C3%A4rungen/search.jsp"/>
```

In case of the ContainerTag it's necessary to UTF-8 encode the url twice:

```
<search:container hasResults="true"
redirectUrl="/search/presseerkl%25C3%25A4rungen/search.jsp" …
```

3.2 Quick install

The easiest and fastest way to install the "FirstSpirit Exalead Live" module is to copy the "search.war" archive into the Webapps directory of the server where it is automatically extracted if the server is running, or following a server reboot at the latest, and is created as a web application called "search".

Now the server name `myserver` in the `webapps/search/WEB-INF/web.xml` file must be replaced by the name of the Exalead server (if necessary adjust the port too).

The search can then be reached by calling `/search/index.jsp`:

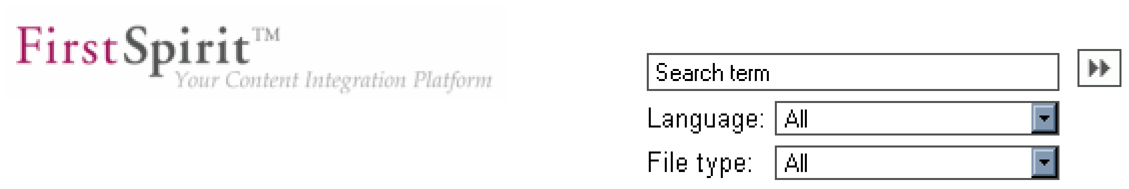


Figure 3-1: Search screen form

3.3 Integration in existing web application

The following files must be copied from the web application created in Chapter 3.1.2 into the existing web application to add the FirstSpirit Exalead Live module to an existing web application:

```
/WEB-INF
      - exalead.tld
      - web.xml (take over only entries which are not available yet)
      - firstpersonalisation.tld (if not available yet)
      - firstpersonalisation.xml (if not available yet)

/WEB-INF/lib

      - complete content


/WEB-INF/classes
      - log4j.properties (if Log4J is not configured elsewhere)
```

In case of concurrent usage of FirstSpirit™ Personalisation it's important to not overwrite the corresponding personalisation.jar in the lib-directory with the personalisation.jar contained in the Exalead Integration module.

The necessary entries in the `web.xml` and the corresponding servlets are described in greater detail in the following. In addition, the changes to the entries in the `web.xml` necessary in relation to integration in an existing web application are also explained.

The tag library with which the search interface and results can be designed to your own wishes is described in Chapter 3.8 page 24.

3.4 web.xml servlets

The Exalead web application provides three servlets:

- *SearchServlet*
  This servlet forwards the search queries to the Exalead server (absolutely necessary).

- *SearchDownload*
  This servlet is responsible for calling results which cannot be reached directly from a user's browser (e.g. search hits in the file system) in the Exalead server and for delivering them to

the browser via the web application.

- *ThumbnailServlet*
  This servlet outputs the preview images of the search hits, provided they exist.

For detailed information on the servlets, please refer to Chapter 3.7 on page 20.

3.5 web.xml context parameter

The context parameter `getDocumentServlet` is only required in conjunction with the `SearchDownload-Servlet` and contains the path to this servlet within the web application (name of the web application in the example: „`/search`“).

3.6 Exemplary web.xml

The necessary parameters in the web.xml file are summarised again here with meaningful values. A web.xml file to which the Exalead integration elements have been added could look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
...
      <context-param>
            <param-name>getDocumentServlet</param-name>
            <param-value>/search/do.searchDownload</param-value>
      </context-param>

      <servlet>
            <servlet-name>SearchServlet</servlet-name>
            <servlet-class>
                  de.espirit.ps.exalead.servlets.SearchServlet
            </servlet-class>
            <init-param>
                  <param-name>serverAddress</param-name>
                  <param-value>
            http://myserver:10000/soap?wsdl=com.exalead.search.wsdl
                  </param-value>
            </init-param>
            <init-param>
                  <param-name>namespace</param-name>
            <param-value>exa:com.exalead.search</param-value>
            </init-param>
            <init-param>
                  <param-name>userPrefix</param-name>
                  <param-value>user:</param-value>
            </init-param>
            <init-param>
                  <param-name>groupsPrefix</param-name>
                  <param-value>groups:</param-value>
            </init-param>
      </servlet>

      <servlet>
            <servlet-name>SearchDownload</servlet-name>
            <servlet-class>
                  de.espirit.ps.exalead.servlets.SearchDownload
            </servlet-class>
            <init-param>
                  <param-name>serverAddress</param-name>
                  <param-value>
                        http://myserver:10000/search/
                  </param-value>
            </init-param>
      </servlet>
      <servlet>
            <servlet-name>ThumbnailServlet</servlet-name>
            <servlet-class>
                  de.espirit.ps.exalead.servlets.ThumbnailServlet
            </servlet-class>
      </servlet>

      <servlet-mapping>
            <servlet-name>SearchServlet</servlet-name>
            <url-pattern>*.search</url-pattern>
      </servlet-mapping>

      <servlet-mapping>
            <servlet-name>SearchDownload</servlet-name>
            <url-pattern>*.searchDownload</url-pattern>
      </servlet-mapping>

      <servlet-mapping>
            <servlet-name>ThumbnailServlet</servlet-name>
            <url-pattern>*.searchThumbnail</url-pattern>
      </servlet-mapping>
...
</web-app>
```

3.7 Description of the servlets

As already mentioned in the previous chapter, the FirstSpirit Exalead Live module is supplied with three servlets, whose configuration is described in the following.

3.7.1 SearchServlet

The `SearchServlet` receives all search enquiries, suitably passes them on to the Exalead server, then receives the search results from it and then makes these results available to the JSP tags responsible for the display.

Configuration example:

```
<servlet>
      <servlet-name>SearchServlet</servlet-name>
      <servlet-class>
            de.espirit.ps.exalead.servlets.SearchServlet
      </servlet-class>
      <init-param>
            <param-name>serverAddress</param-name>
            <param-value>
      http://myserver:10000/soap?wsdl=com.exalead.search.wsdl
            </param-value>
      </init-param>
      <init-param>
```

```
            <param-name>namespace</param-name>
            <param-value>exa:com.exalead.search</param-value>
</init-param>
      <init-param>
      <param-name>userPrefix</param-name>
      <param-value>user:</param-value>
</init-param>
<init-param>
      <param-name>groupsPrefix</param-name>
      <param-value>groups:</param-value>
</init-param>
</servlet>

<servlet-mapping>
      <servlet-name>SearchServlet</servlet-name>
      <url-pattern>*.search</url-pattern>
</servlet-mapping>
```

The servlet must know the call of the Exalead server's SOAP interface in order to be able to forward the search queries. It is informed of these with the `serverAddress` parameter. To this end, a suitable "command“ of the type "SOAP" must have been created in the Exalead administration.

The `namespace` parameter depends on the Exalead version used and for versions up to and including 4.6.0.181 must be `http://endpoint` and for later versions is must be `exa:com.exalead.search`.

The `userPrefix` and `groupsPrefix` parameters give the prefixes of the user and group permissions for documents in the search index and therefore only play a role if personalised search results are to be displayed. If permissions for documents are assigned manually (see also Chapter 4.2, 5.2.2.13 and 5.2.2.14), the prefixes used there must correspond to the values of these parameters. If these parameters are not configured in the `SearchServlet`, the default values `exalead:user:` and `exalead:group` are used.

Within the search page, the servlet is addressed using a form. An exemplary call of the `SearchServlets` looks like this:

```
<form action="do.search">
      <input type="text" name="q" value=""/>
      <input type="hidden" name="b" value="0"/>
      <input type="hidden" name="l" value="de"/>
      <input type="hidden" name="hf" value="10"/>
      <input type=”hidden” name=”r” value=”Top/source/intranet”/>
      <input type=”hidden” name=”r” value=”Top/mime/text#html”/>
      <input type="hidden" name="redirectUrl" value="search.jsp"/>
      <input type="hidden" name="errorUrl" value="error.jsp"/>
      <input type="submit" value="Search"/>
```

```
</form>
```

The parameters have the following meaning:

Parameter	Expected value
q	Input field for the search word (term)
b	Index of the first hit to be displayed in the results list
l	Language to be used to interpret the search term
hf	Number of hits to be displayed per results page
r	Field for search refinements (optional), can be used multiple times
redirectUrl	Results page (only local URLs allowed)
errorUrl	Errors page (only local URLs allowed)

In case the search wasn't successful e.g. due to an unreachable search server, the `SearchServlet` will write the correspondent exception to the session variable `exception.` This variable can then be readout in the search results page so that an appropriate message can be displayed.

3.7.2 SearchServlet – search in results

`The SearchServlet` can also be used to refine the search within search results again by adding another search word. To do this, another search form is integrated into the search results page, which is configured for the search in the search results.

An exemplary call of the `SearchServlets` for the search in search results looks like this:

```
<form action="do.search">
      <input type="hidden" name="q" value="<%=
session.getAttribute("searchString") %>"/>
      <input type="hidden" name="C" value="<%= pageContext.getAttribute("C")
%>"/>
<input type="hidden" name="b" value="0"/>
<input type="hidden" name="l" value="de"/>
<input type="hidden" name="hf" value="10"/>
<input type="hidden" name="redirectUrl" value="search.jsp"/>
```

```
<input type="hidden" name="errorUrl" value="error.jsp"/>
<input type="text" name="noq" value=""/>
<input type="submit" value="Search"/>
</form>
```

3.7.3 SearchDownload

The `SearchDownload` servlet is used to deliver search hits in the search results lists via the web application, as this cannot be directly accessed from the browser because, for example, they are located in a network file system. To do this, this servlet queries the Exalead server for the file and then forwards it to the browser.

Configuration example:

```
      <context-param>
            <param-name>getDocumentServlet</param-name>
            <param-value>/search/do.searchDownload</param-value>
      </context-param>

      <servlet>
            <servlet-name>SearchDownload</servlet-name>
            <servlet-class>
                  de.espirit.ps.exalead.servlets.SearchDownload
            </servlet-class>
            <init-param>
                  <param-name>serverAddress</param-name>
                  <param-value>
                        http://myserver:10000/search/
                  </param-value>
            </init-param>
      </servlet>

      <servlet-mapping>
            <servlet-name>SearchDownload</servlet-name>
            <url-pattern>*.searchDownload</url-pattern>
      </servlet-mapping>
```

To query the Exalead server, the servlet must know the address of the standard Exalead search, which it is informed of via the `serverAddress` parameter. For automatic generation of links to the `SearchDownload` servlet, the `getDocumentServlet` context parameter must be used to inform the application of the address under which the `SearchDownload` servlet can be addressed.

A description of how the output of automatically generated download links can be controlled within the search results page is given in Chapter 3.12.3 on page 41.

3.7.4 ThumbnailServlet

The `ThumbnailServlet` delivers the thumbnails for the search result hits, provided they exist. If a thumbnail does not exist for the hit, a replacement image can also be given in interaction with the corresponding JSP tags.

```
<servlet>
            <servlet-name>ThumbnailServlet</servlet-name>
            <servlet-class>
                  de.espirit.ps.exalead.servlets.ThumbnailServlet
            </servlet-class>
      </servlet>

      <servlet-mapping>
            <servlet-name>ThumbnailServlet</servlet-name>
            <url-pattern>*.searchThumbnail</url-pattern>
      </servlet-mapping>
```

3.8 JSP tags

With the FirstSpirit Exalead JSP tags it is possible to display the search results in virtually any layout. For example, a preview image can be displayed for each hit, hits in certain categories can be highlighted, refinements (drilldown) can be given for the search results, and much more.

The following Exalead functions are not supported at the present time:

- Display of related terms

The JSP tags available for this are briefly described in the following and are then dealt with in detail.

Global tags:

- <search:container>
  Provision of all data concerning the search results
  (see Chapter 3.10.1 page 31).
- <search:query>
  Output of the search term:
  (see Chapter 3.10.2 page 33).
- <search:nhits>
  Number of search results
  (see Chapter 3.10.3 page 33).
- <search:time_overall>
  Output of the search query duration

(see Chapter 3.10.4 page 33).

- <search:sort_link> and <search:sort_linkparameter>
  Output of a like for sorting the search results by relevance, date or size
  (see Chapter 3.10.5 page 33).
- <search:suggestions_loop>
  Iteration over suggestions for search terms
  (see Chapter 3.10.7 page 36).
- <search:suggestions_link> and <search:suggestions_linkparameter>
  Output of the link to the search with suggested search term
  (see Chapter 3.10.8 page 36).
- <search:suggestions_choice>
  Output of the suggested search term
  (see Chapter 3.10.10 page 36).

Tags for refining the search results:

- <search:groups>
  Provision of all data concerning the refinement of the search results
  (see Chapter 3.11.1 page 36).
- <search:group>
  Provision of the information of a specific search category
  (see Chapter 3.11.2 page 37).
- <search:groups_categories_loop>
  Iteration over the entries of a specific search category
  (see Chapter 3.11.3 page 37).
- <search:groups_category_title>
  Output of the entry's title
  (see Chapter 3.11.4 page 38).
- <search:groups_category_count>
  Output of the number of hits within this entry
  (see Chapter 3.11.5 page 38).
- <search:groups_category_link> and <search:groups_category_linkparameter>
  Output of the link for limiting the search results to this entry
  (see Chapter 3.11.6 page 38).
- <search:reset_refinement>
  Provision of the information about search result limitations made
  (see Chapter 3.11.8 page 39).
- <search:hasRefinements>
  Checks whether limitations exist for a specific category
  (see Chapter 3.11.9 page 39).

- <search:refinements>
  Provision of the information on the restrictions of a specific category
  (see Chapter 3.11.10 page 40).
- <search:reset_link> and <search:reset_linkparameter>
  Link to reset the limitation of a specific category
  (see Chapter 3.11.11 page 40).

Tags for displaying the search results:

- <search:loop_hits>
  Iterates over the search results
  (see Chapter 3.12.1 page 41).
- <search:hits_id>
  Output of the consecutive number of the hit (starting at 0)
  (see Chapter 3.12.2 page 41).
- <search:hits_url>
  Output of the hit's URL
  (see Chapter 3.12.3 page 41).
- <search:hits_title>
  Output of the hit's heading
  (see Chapter 3.12.4 page 42).
- <search:hits_doctype>
  Output of the hit's document type
  (see Chapter 3.12.5 page 43).
- <search:hits_score>
  Output of the hit's ranking
  (see Chapter 3.12.6 page 43).
- <search:hits_summary>
  Output of the text extract of the hit
  (see Chapter 3.12.7 page 43).
- <search:hits_field_attributes>
  Output of other attributes of a hit
  (see Chapter 3.12.8 page 44).
- <search:hits_filesize>
  Output of the size of a hit
  (see Chapter 3.12.9 page 45).
- <search:hits_date>
  Output of the date of a hit
  (see Chapter 3.12.10 page 45).
- <search:hits_has_thumbnail> / <search:hits_has_no_thumbnail>

Checks whether a preview image of the hit exists
(see Chapter 3.12.11 page 46).

- <search:is_in_category> / <search:is_not_in_category>
  Checks whether the search result belongs to a specific category
  (see Chapter 3.12.13 page 47).
- <search:has_meta_data>
  Checks whether the hit has a specific metadata key/value pair
  (see Chapter 3.12.15 page 48).

Tags for navigation:

- <search:navigation>
  Provision of the information for navigation
  (see Chapter 3.13.1 page 50).
- <search:navigation_page_first_link> and
  <search:navigation_page_first_linkparameter>
  Output of the link to the first results page
  (see Chapter 3.13.2 page 50).
- <search:navigation_page_previous_container>
  Checks whether the link to the previous results page is to be displayed or not
  (see Chapter 3.13.4 page 51).
- <search:navigation_page_previous_link> and
  <search:navigation_page_previous_linkparameter>
  Output of the link to the previous results page
  (see Chapter 3.13.5 page 51).
- <search:navigation_loop_pages>
  Iterates over the page numbers of the navigation to be displayed
  (see Chapter 3.13.7 page 51).
- <search:navigation_is_current_page>
  Checks whether the displayed page number corresponds to the currently displayed results page
  (see Chapter 3.13.8 page 52).
- <search:navigation_is_not_current_page>
  Checks whether the displayed page number does not correspond to the currently displayed results page
  (see Chapter 3.13.9 page 52).
- <search:navigation_page>
  Output the page number
  (see Chapter 3.13.10 page 52).
- <search:navigation_page_link> and

<search:navigation_page_linkparameter>
Output of the link of the results page corresponding to the displayed page number
(see Chapter 3.13.11 page 52).

- <search:navigation_page_next_container>
  Checks whether the link to the next results page should be displayed
  (see Chapter 3.13.13 page 52).
- <search:navigation_page_next_link> and
  <search:navigation_page_next_linkparameter>
  Output of the link to the next results page
  (see Chapter 3.13.14 page 53).
- <search:navigation_page_last_container>
  Checks whether the link to the last results page can be displayed
  (see Chapter 3.13.16 page 53).
- <search:navigation_page_last>
  Output of the page number of the last results page
  (see Chapter 3.13.17 page 53).
- <search:navigation_page_last_link> and
  <search:navigation_page_last_linkparameter>
  Output of the link to the last results page
  (see Chapter 3.13.18 page 54).

A minimum search results page could therefore look like this, for example:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@ taglib prefix="search" uri="/WEB-INF/exalead.tld" %>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">



<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Search results</title>

</head>

<body>

<form action="do.search">

<input type="text" name="q" value="<%= session.getAttribute("searchString") != null ?
```

```
session.getAttribute("searchString") : "" %>"/>

<input type="hidden" name="b" value="0"/>

<input type="hidden" name="l" value="de"/>

<input type="hidden" name="hf" value="10"/>

<input type="hidden" name="redirectUrl" value="search.jsp"/>

<input type="hidden" name="errorUrl" value="search.jsp"/>

<input type="submit" value="Search"/>

</form>

<hr/>

<p>Display search results </p>

<search:container hasResults="true" redirectUrl="search.jsp" errorUrl="search.jsp"
hitsPerPage="10">

<table border="1">

<tr><th>Index</th><th>Title</th><th>URL</th><th>Summary</th></tr>

<search:loop_hits>

            <tr>

            <td><search:hits_id/></td>

            <td><a href="<search:hits_url completeDocUrl="true"/>">

                           <search:hits_title/>

            </a></td>

            <td><search:hits_url/></td>

            <td><search:hits_summary isLongForm="true" endline=" ...<br/>"
highlightStart="<strong>" highlightEnd="</strong>" /></td>

            </tr>

</search:loop_hits>

</table>

<br/>

<table border="1">

            <tr>

            <search:navigation surroundingPagesRadius="5">

            <td><a href="<search:navigation_page_first_link/>">
```

```
                              First page

               </a></td>

               <search:navigation_page_previous_container>

               <td><a href="<search:navigation_page_previous_link/>">

                              Previous page

               </a></td>

               </search:navigation_page_previous_container>

               <search:navigation_loop_pages>

               <td>

               <search:navigation_is_current_page>

                              <strong><search:navigation_page/></strong>

               </search:navigation_is_current_page>

               <search:navigation_is_not_current_page>

               <a href="<search:navigation_page_link/>">

               <search:navigation_page/></a>

               </search:navigation_is_not_current_page>

               </td>

               </search:navigation_loop_pages>

               <search:navigation_page_next_container>

               <td><a href="<search:navigation_page_next_link/>">

                              Next page

               </a></td>

               </search:navigation_page_next_container>

               <search:navigation_page_last_container>

               <td><a href="<search:navigation_page_last_link/>">

                              Last page (<search:navigation_page_last/>)

               </a></td>

               </search:navigation_page_last_container>

               </search:navigation>

               </tr>
```

```
</table>

</search:container>

<search:container hasResults="false" redirectUrl="index.jsp" errorUrl="index.jsp"
hitsPerPage="10">

<strong>Sorry, no matches found...</strong><br/>

</search:container>

</body>

</html>
```

3.9 Tag prefix

The relevant tag library must be given in the JSP pages in order to be able to use FirstSpirit Exalead tags. This document uses the "search" prefix for the FirstSpirit Exalead tags.

Example of integration in JSP pages:

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="/WEB-INF/exalead.tld" prefix="search" %>
```

If the “search" prefix is changed to another value, this prefix must be used for the individual tags, i.e. “<**myPrefix**:ref>” instead of “<search:ref>".

3.10 Global tags

3.10.1 <search:container>

The <search:container> tag prepares the search results so that they can be evaluated and output by the other tags available (refinements, hits, navigation).

Attributes:

Attribute	Meaning	Mandatory parameter
hasResults	This attribute controls whether the content of the tags is to be displayed if search hits exist (hasResults="true") or whether the content is to be displayed if there are no search hits (hasResults="false").	Yes
redirectUrl	This attribute gives the address of the search results page. In general, this is the page on which you are currently located. It is necessary to specify this information so that the automatically generated links (e.g. for the navigation) link to the correct page.	Yes
errorUrl	This attribute gives the address of the page which is set in the event of an error. This information is also required for the automatically generated links.	Yes
hitsPerPage	This attribute is used to give the number of hits to be displayed per page. If such information was already specified in the SearchServlet form, the same value must be entered here.	no

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp" hitsPerPage="20">
Display hits
</search:container>
<search:container hasResults="false" redirectUrl=" search.jsp" errorUrl="
error.jsp" hitsPerPage="20">
      No matches found
</search:container>
```

3.10.2 <search:query />

The <search:query/> tag outputs the current search query.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
Your search for '<search:query />' results in <search:nhits /> hits.
</search:container>
```

3.10.3 <search:nhits />

The <search:nhits/> tag outputs the number of hits found.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
Your search for '<search:query />' results in <search:nhits /> hits.
</search:container>
```

3.10.4 <search:time_overall />

The <search:time_overall /> tag outputs the duration of the search query.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
Your search for '<search:query />' results in <search:nhits /> hits in
<search:time_overall /> milliseconds.
</search:container>
```

3.10.5 <search:sort_link />

The <search:sort_link /> tag outputs automatically generated links for sorting by relevance, date or size. To do this, you give the link the sort criterion via an attribute. Switching between ascending and descending sorting takes place automatically and is adopted from the tag.

Attributes:

Attribute	Meaning	Mandatory parameter
sort	This attribute is used to give the sort criterion for the link to be output. Possible values are: score, date and size	Yes

Simple example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
<a href="<search:sort_link sort="score"/>" >Sorting by relevance</a>
<a href="<search:sort_link sort="date"/>" >Sorting by date</a>
<a href="<search:sort_link sort="size"/>" >Sorting by size</a>
</search:container>
```

Advanced example:

If you would like to hide the link to the current sort order and in addition, also indicate whether a click on a link allows sorting in ascending or descending order, you must get the current values from the session beforehand:

```
<%  final String currentSort = (String)
session.getAttribute(de.espirit.ps.exalead.resources.SearchRequestArguments.SE
SSION_SORT);
    final String currentOrder = (String)
session.getAttribute(de.espirit.ps.exalead.resources.SearchRequestArguments.SE
SSION_ORDER); %>
```

These values can then be used to more precisely specify output of the links:

```
<% if ("date".equals(currentSort) || "size".equals(currentSort)) { %>

<!-- Output with linking -->
<a href="<search:sort_link sort="score"/>" >By relevance</a>

<% } else { %>

<!-- Output without linking, because inversion of sort order is not desired
when searching by relevance -->
<b>By relevance</b>
<% }
   if (!"date".equals(currentSort)) { %>

<!-- Output without indication of the sort order, because from the beginning
results are sorted always in descending order -->
<a href="<search:sort_link sort="date"/>" >By date</a>

<% } else { if ("1".equals(currentOrder)) { %>

<!-- If the results have been sorted by date yet another click on the link
will invert the sort order -->
<b>By date</b> <a href="<search:sort_link sort="date"/>" >descending</a>


<% }  else { %>

<!-- If the results have been sorted by date yet another click on the link
will invert the sort order -->
<b>By date</b> <a href="<search:sort_link sort="date"/>" >ascending</a>

<% } }
   if (!"size".equals(currentSort)) { %>

<!-- Output without indication of the sort order, because from the beginning
results are sorted always in descending order -->
<a href="<search:sort_link sort="size"/>" >By size</a>

<% } else { if ("1".equals(currentOrder)) { %>

<!-- If the results have been sorted by size yet another click on the link
will invert the sort order -->
<b>By size</b> <a href="<search:sort_link sort="size"/>" >descending</a>

<% } else { %>

<!-- If the results have been sorted by size yet another click on the link
will invert the sort order -->
<b>By size</b> <a href="<search:sort_link sort="size"/>" >ascending</a>

<% } } %>
```

3.10.6 <search:sort_linkparameter />

The functionality corresponds to that of the tag <search:sort_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

Example:

```
<a href="do.search?<search:sort_linkparameter sort="date"/>" >Sort by date</a>
```

3.10.7 <search:suggestions_loop>

The <search:suggestions_loop> tag iterates over all suggestions for search terms and provides the respective suggestion with the corresponding search link.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
<% if ("true".equals(pageContext.getAttribute("hasSuggestions"))) { %>
      Did you mean
      <search:suggestions_loop>
      <a href="<search:suggestions_link/>">
      <search:suggestions_choice/>
      </a>
      </search:suggestions_loop> ?
<% } %>
</search:container>
```

3.10.8 <search:suggestions_link />

The <search:suggestions_link/> tag shows the link to the search with the suggested search term.

3.10.9 <search:suggestions_linkparameter />

The functionality corresponds to that of the tag <search:suggestions_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.10.10 <search:suggestions_choice />

The <search:suggestions_choice/> tag shows the suggested search term.

3.11 Tags for refining the search results

3.11.1 <search:groups>

The <search:groups> tag defines the area for the display of the available search categories and the search refinements made

3.11.2 <search:group>

The <search:group> tag is called within the <search:groups> tag and contains the available entries of a specific search category, which the tag is notified of by means of an attribute. If this search category does not contain any entries the tag hides its complete contents.

Attributes:

Attribute	Meaning	Mandatory parameter
groupID	Name of the search category whose entries are to be displayed. If no entries are available the content is hidden.	Yes

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
      <search:groups>
            <search:group groupID="Category1">
                  Display of entries of Category1
            </search:group/>
            <search:group groupID="Category2">
                  Display of entries of Category2
            </search:group/>
      </search:groups>
</search:container>
```

3.11.3 <search:groups_categories_loop>

The <search:groups_categories_loop> tag iterates over all entries of a search category and makes the displayable information of these entries available.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
      <search:groups>
            <search:group groupID="Category1">
                  <search:groups_categories_loop>
<a href="<search:groups_category_link/>">
<search:groups_category_title/>
</a>
(<search:groups_category_count/>)
<a href="<search:groups_category_link action="exclude"/>">
```

```
excluding
</a>
                  </search:groups_categories_loop>
            </search:group/>
      </search:groups>
</search:container>
```

3.11.4 <search:groups_category_title />

This tag outputs the title of the entry.

For an example, see Chapter 3.11.3 page 37.

3.11.5 <search:groups_category_count />

This tag outputs the number of search hits for this entry.

For an example, see Chapter 3.11.3 page 37.

3.11.6 <search:groups_category_link />

This tag outputs the link for limiting the search results to this entry or to exclude the search results for this entry.

Attributes:

Attribute	Meaning	Mandatory parameter
action	If action="include" (default value), the link for limiting the search results to this entry is output. If action="exclude" the link for excluding the search results of this entry is output.	No

For an example, see Chapter 3.11.3 page 37.

3.11.7 <search:groups_category_linkparameter />

The functionality corresponds to that of the tag <search:groups_category_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.11.8 <search:reset_refinement>

The <search:reset_refinements> tag defines the area for the display of the search refinements made. The content of the tag is hidden if no refinements have been made.

If refinements have been made and the content of the tag is shown, the pagecontext variable "resetUrl" can be used to get a link for removing all the refinements made. As an alternative, the pagecontext variable "resetUrlParameter" can be used to query only the parameters of the reset link.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
      <search:groups>
            <search:reset_refinement>
<a href="<%= resetUrl %>">Reset all refinements</a>
            </search:reset_refinement>
      </search:groups>
</search:container>
```

3.11.9 <search:hasRefinements>

The <search:hasRefinements> tag is called within the <search:reset_refinement> tag and contains the refinements to a specific search category made, which the tag was notified of by means of an attribute. If no refinements have been made for this search category, the tag hides its complete contents.

Attributes:

Attribute	Meaning	Mandatory parameter
groupID	Name of the search category whose refinements are to be displayed. If no refinements exist, the content is hidden.	Yes

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
      <search:groups>
            <search:reset_refinement>
            <search:hasRefinements groupID="Category1">
                        Display of refinements of Category1
                  </search:hasRefinements />
            <search:hasRefinements groupID="Category2">
                        Display of refinements of Category2
                  </search:hasRefinements />
            </search:reset_refinement>
      </search:groups>
</search:container>
```

3.11.10 <search:refinements>

The <search:refinements> tag iterates over all refinements made to a search category.

Within the tag, the pagecontext variable "isExcluded" can be used to check whether the refinement involves a limitation or whether it is an exclusion rule. The "path" variable returns the names of the refinement.

Example:

```
<search:container hasResults="true" redirectUrl="search.jsp"
errorUrl="error.jsp">
      <search:groups>
            <search:reset_refinement>
            <search:hasRefinements groupID="Category1">
                  <search:refinements>
<%= isExcluded ? "NOT " : "" %><%= path %>
<a href="<search:reset_link/>">delete</a>
                  </search:refinements>
                  </search:hasRefinements />
            </search:reset_refinement>
      </search:groups>
</search:container>
```

3.11.11 <search:reset_link />

The <search:reset_link /> tag returns the link for resetting the refinement.

For an example, see Chapter 3.11.10 page 40.

3.11.12 <search:reset_linkparameter />

The functionality corresponds to that of the tag <search:reset_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.12 Tags for displaying the search results

3.12.1 <search:loop_hits>

The <search:loop_hits> tag iterates over all search hits of the currently displayed results page and makes the displayable information of these search hits available.

3.12.2 <search:hits_id />

This tag gives the current index of the search hit within the hit list. The first search hit has the ID 0.

Example:

```
<search:loop_hits>
      ID of hit: <search:hits_id/>
</search:loop_hits>
```

3.12.3 <search:hits_url />

This tag returns the URL of the search hit.

Attributes:

Attribute	Meaning	Mandatory parameter
completeDocUrl	Set to “false” (default value), the tag returns the URL of the document, as it is saved in the index. However, these links cannot be effortlessly opened in the case of files which cannot be reached by http. completeDocUrl=“true“ is used to extend links to documents which cannot be reached by http to include the path of the DownloadServlet, so that these documents can be delivered.	No
replaceRule	Replacement rule for replacing part of the URL with another value, for example, another server. Syntax: "<alter String>,<neuer String>“	No

Example:

```
<search:loop_hits>
      URL of the document: <a href="<search:hits_url completeDocUrl="true"
replaceRule="myIndexServer,myWebsiteServer"/>">
<search:hits_url replaceRule="myIndexServer,myWebsiteServer"/>
</a>
</search:loop_hits>
```

3.12.4 <search:hits_title />

This tag returns the title of the search hit.

Attributes:

Attribute	Meaning	Mandatory parameter
highlightStart / highlightEnd	These attributes can be used to define the type of highlighting of the search word within the title.	No

Example:

```
<search:loop_hits>
      Title: <search:hits_title highlightStart="<strong>"
highlightEnd="</strong>" />
</search:loop_hits>
```

3.12.5 <search:hit_doctype />

This tag returns the document type of the search hit.

Example:

```
<search:loop_hits>
      Type: <search:hits_doctype />
</search:loop_hits>
```

3.12.6 <search:hits_score />

This tag returns the score (number of points) of the search hit.

Example:

```
<search:loop_hits>
      Type: <search:hits_score />
</search:loop_hits>
```

3.12.7 <search:hits_summary />

This tag returns an extract of the search hit.

Attributes:

Attribute	Meaning	Mandatory parameter
isLongForm	This attribute is used to control whether the first line of the extract only (isLongForm = false) or whether the complete extract (isLongForm = true) is to be shown. The default value is false.	No
endline	This attribute can be used to define a text snippet, which is appended to the end of each line of the extract (e.g. "...<br/>", to end each line with three dots and a line break).	No
highlightStart / highlightEnd	These attributes can be used to define the way in which the search word is highlighted within the text extract.	No

Example:

```
<search:loop_hits>
      <search:hits_summary isLongForm="true" endline=" ...<br/>"
highlightStart="<strong>" highlightEnd="</strong>" />
</search:loop_hits>
```

3.12.8 <search:hits_field_attributes />

This tag can be used to read out additional index fields (e.g. metadata from FirstSpirit or metadata, which was given in the HTML header of a page). The fields which can be read out must be defined beforehand using Exalead. If there are multiple values defined for a field, these values will be returned as a comma separated list.

Attributes:

Attribute	Meaning	Mandatory parameters
Key	Name of the index field to be output.	Yes

Example:

```
<search:loop_hits>
      <search:hits_field_attributes key="myCustomAttribute">
            <%= value %>
      </search:hits_field_attributes>
</search:loop_hits>
```

3.12.9 <search:hits_filesize />

This tag returns the size of the document.

Attributes:

Attribute	Meaning	Mandatory parameter
type	Units in which the size is to be output. Possible values: b, kb, mb, gb. If this attribute is omitted, a suitable unit is automatically selected.	No

Example:

```
<search:loop_hits>
      File size: <search:hits_filesize type="kb" />
</search:loop_hits>
```

3.12.10 <search:hits_date />

This tag returns the date on which the document was created.

Attributes:

Attribute	Meaning	Mandatory parameter
format	The output format of the date, e.g. dd.MM.yy	Yes

Example:

```
<search:loop_hits>
      Date: <search:hits_date format="dd.MM.yy" />
</search:loop_hits>
```

3.12.11 <search:hits_has_thumbnail>

This tag checks whether a preview image exists for a search hit. If this is the case, the content of the tag is shown, otherwise it is hidden.

This check as well as the display of the preview image on the basis of the thumbnailServlet requires a counter which counts the position of the current search hit within the current page.

Attributes:

Attribute	Meaning	Mandatory parameter
index	Position of the search hit in the current page	Yes

Example:

```
<% int index = 0; %>
<search:loop_hits>
      <search:hits_has_thumbnail index="<%= index %>">
            <img src="/exalead/do.searchThumbnail?index=<%= index %>" alt=""
border="0" />
</search:hits_has_thumbnail>
      <search:hits_has_no_thumbnail index="<%= index %>">
      <img src="images/no_thumbnail.gif" alt="" border="0" />
      </search:hits_has_no_thumbnail></a>
      <% index++; %>
</search:loop_hits>
```

3.12.12 <search:hits_has_no_thumbnail>

This tag checks whether a preview image exists for a search hit. If this is **not** the case, the content of the tag is shown, otherwise it is hidden.

This check requires a counter which counts the position of the current search hit within the current page.

Attributes:

Attribute	Meaning	Mandatory parameter
index	Position of the search hit in the current page	Yes

For an example, see Chapter 3.12.11 page 46.

3.12.13 <search:is_in_category>

This tag can be used to check whether a search hit belongs to a specific category. If this is the case, the contents of the tag are shown.

Attributes:

Attribute	Meaning	Mandatory parameter
fullPath	The absolute path to the category.	Yes

Example:

```
<search:loop_hits>
      <search:is_in_category fullPath="Top/FirstSpirit/common">This hit
belongs to the category FirstSpirit/common     </search:is_in_category>
      <search:is_not_in_category fullPath="Top/FirstSpirit/common"> This hit
does not belong to the category FirstSpirit/common
      </search:is_not_in_category>
</search:loop_hits>
```

3.12.14 <search:is_not_in_category>

This tag can be used to check whether a search hit belongs to a specific category. The content of the tag is shown if this is **not** the case.

Attributes:

Attribute	Meaning	Mandatory parameter
fullPath	The absolute path to the category.	Yes

For an example, see Chapter 3.12.13 page 47.

3.12.15 <search:has_meta_data>

This tag can be used to check a metadata of a search hit for a specific value. Possible metadata are: date, mime, ext, size, documenturl, hitindex. If the value matches, the content of the tag is shown.

Attributes:

Attribute	Meaning	Mandatory parameter
key	Name of the meta data to be compared.	Yes
value	Value with which the metadata is to be compared.	Yes
exclude	With exclude="true", the content of the tag is only shown if no match occurred.	No

Example:

```
<search:loop_hits>
      <search:has_meta_data key="mime" value="application/pdf">
            <img src="pdf_file_icon.gif"/>
</search:has_meta_data>
<search:has_meta_data key="mime" value="text/html">
            <img src="html_file_icon.gif"/>
```

```
</search:has_meta_data>
<search:has_meta_data key="mime" value="application/pdf,text/html"
exclude="true">
      <img src="unknown_file_icon.gif"/>
</search:has_meta_data> </search:hits_field_attributes>
</search:loop_hits>
```

3.13 Tags for the navigation

3.13.1 <search:navigation>

The <search:navigation> tag defines the area for displaying the navigation through the search results pages.

Attributes:

Attribute	Meaning	Mandatory parameter
surroundingPagesRadius	Defines the page radius of the navigation around the current page. If no value or a values smaller than 5 is given, a radius of 5 is used.	No

Example:

```
<search:navigation surroundingPagesRadius="5">
      … navigation bar …
</search:navigation>
```

3.13.2 <search:navigation_page_first_link />

This tag generates the link to the first results page.

Example:

```
<search:navigation surroundingPagesRadius="5">
<a href="<search:navigation_page_first_link />">First page</a>
</search:navigation>
```

3.13.3 <search:navigation_page_first_linkparameter />

The functionality corresponds to that of the tag <search:navigation_page_first_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.13.4 <search:navigation_page_previous_container>

This tag checks whether the current search results page is the first page and only outputs the content of the tag if this is not the case.

Example:

```
<search:navigation surroundingPagesRadius="5">
<search:navigation_page_previous_container>
      <a href="<search:navigation_page_previous_link/>">
            previous page
      </a>
</search:navigation_page_previous_container>
</search:navigation>
```

3.13.5 <search:navigation_page_previous_link />

This tag generates the link to the previous page in the results page navigation.

For an example, see Chapter 3.13.4 page 51.

3.13.6 <search:navigation_page_previous_linkparameter />

The functionality corresponds to that of the tag <search:navigation_page_previous_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.13.7 <search:navigation_loop_pages>

This tag iterates over the search results pages of the navigation, depending on the set pages radius.

Example:

```
<search:navigation surroundingPagesRadius="5">
<search:navigation_loop_pages>
      <search:navigation_is_current_page>
            <strong><search:navigation_page/></strong>
      </search:navigation_is_current_page>
      <search:navigation_is_not_current_page>
            <a href="<search:navigation_page_link/>">
                  <search:navigation_page/>
            </a>
      </search:navigation_is_not_current_page>
</search:navigation_loop_pages>
</search:navigation>
```

3.13.8 <search:navigation_is_current_page>

This tag checks whether the page from the iteration is the currently displayed search results page. If this is the case, the contents of the tag are shown.

For an example, see Chapter 3.13.7 page 51.

3.13.9 <search:navigation_is_not_current_page>

This tag checks whether the page from the iteration is **not** the currently displayed search results page. If this is the case, the contents of the tag are shown.

For an example, see Chapter 3.13.7 page 51.

3.13.10 <search:navigation_page />

This tag outputs the page number of the search results page.

For an example, see Chapter 3.13.7 page 51.

3.13.11 <search:navigation_page_link />

This tag generates the link to the search results page.

For an example, see Chapter 3.13.7 page 51.

3.13.12 <search:navigation_page_linkparameter />

The functionality corresponds to that of the tag <search:navigation_page_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.13.13 <search:navigation_page_next_container>

This tag checks whether the current search results page is the last page and only outputs the content of the tag is this is not the case.

Example:

```
<search:navigation surroundingPagesRadius="5">
<search:navigation_page_next_container>
      <a href="<search:navigation_page_next_link/>">
```

```
                  next page
            </a>
</search:navigation_page_next_container>
</search:navigation>
```

3.13.14 <search:navigation_page_next_link />

This tag generates the link to the next page in the results page navigation.

For an example, see Chapter 3.13.13 page 52.

3.13.15 <search:navigation_page_next_linkparameter />

The functionality corresponds to that of the tag <search:navigation_page_next_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.13.16 <search:navigation_page_last_container>

This tag checks whether the page number of the last page is known or not. If there is a very large quantity of hits, it is possible that the number of hits and therefore the number of results pages too, can only be estimated. If this is the case, the contents of this tag are hidden.

Example:

```
<search:navigation surroundingPagesRadius="5">
<search:navigation_page_last_container>
      <a href="<search:navigation_page_last_link/>">
      Last page (<search:navigation_page_last/>)
      </a>
</search:navigation_page_last_container>
</search:navigation>
```

3.13.17 <search:navigation_page_last />

This tag gives the page number of the last search results page.

For an example, see Chapter 3.13.16 page 53.

3.13.18 <search:navigation_page_last_link />

This tag generates the link to the last search results page.

For an example, see Chapter 3.13.16 page 53.

3.13.19 <search:navigation_page_last_linkparameter />

The functionality corresponds to that of the tag <search:navigation_page_last_link/>, the only difference being that this tag generates only the URL parameters for the required servlet call.

3.14 Implicit search

3.14.1 <search:loop_tophits>

The <search:loop_tophits> tag is used to display the top N search results for a specific predetermined search word or for the current search query; however, independent of any search refinements.

Attributes:

Attribute	Meaning	Mandatory parameter
query	Specification of the search word (term). If nothing is specified, the current search word of the normal search is used.	No
filetype	Limitation to search results of a specific file type.	No
numberOfHits	Maximum number of displayed hits. Default value: 5	No

Example:

```
<search:loop_tophits query="internet" filetype="pdf" numberOfHits="3">
      <a href="<search:hits_url completeDocUrl="true"/>">
```

```
            <search:hits_title/>
      </a><br/>
</search:loop_tophits>
```

The same tags can be used within the <search:loop_tophits> tag as within the <search:loop_hits> tag, with the exception of the display of thumbnails.

4 Personalised Search Results

In order to achieve personalised display of the search results, Exalead gives users the opportunity to read out user and group information from documents and to display these documents in the list of the search results to authorised users only.

This chapter deals with the possibilities of assigning permissions to documents. In this context, it also explains use of the Push-API, which allows documents to be individually transferred into the index.

4.1 Personalised search on HTML pages

An easy way to assign permissions to HTML pages is to specify meta information, which can be read out by Exalead and used to determine the user and group guidelines.

4.2 Preparation of the HTML files / templates

The groups or users are entered in the HTML file as a MetaTag in the following form:

```
<meta name="security" content="exalead:group:AnyGroup" />
<meta name="security" content="exalead:user:Jane Doe" />
```

Any name can be chosen for the MetaTag (in the example: security). The `content` attribute is used to assign the permissions for this document. This should always consist of a prefix and the user or group name. **Important:** The prefix for groups (here `exalead:group:`) should differ from the prefix for user names (here `exalead:user:`).

Apart from one exception, any prefixes can be chosen (see also Chapter 3.7.1). If the standard Exalead search interface is used for the search instead of the FirstSpirit Exalead Live Module, the prefixes must be as in the example above.

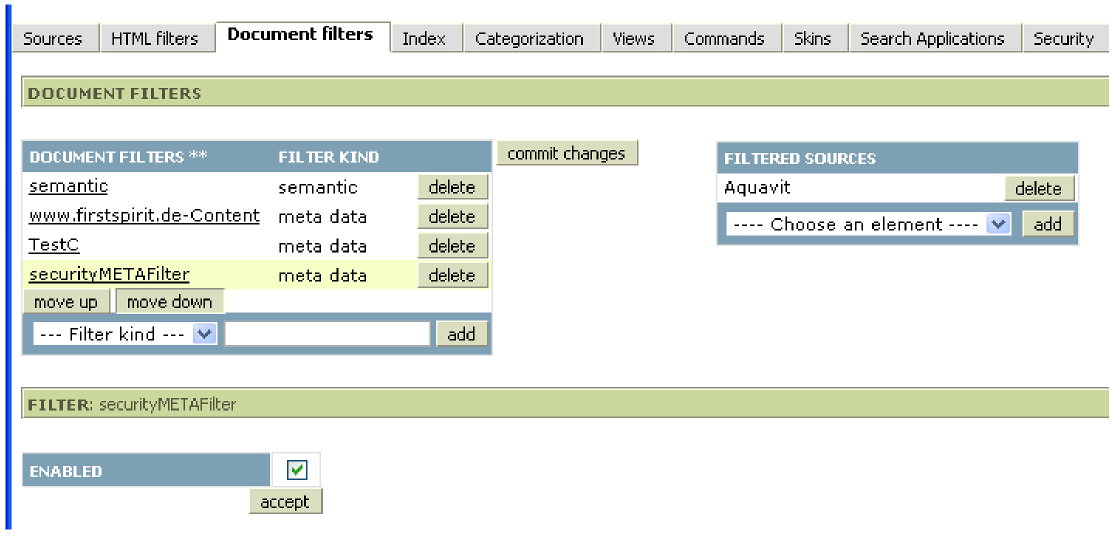
The MetaTag can occur n times in the document, for example, in order to specify several groups and users:

```
<meta name="security" content="exalead:group:Editor" />
<meta name="security" content="exalead:group:Guest" />
<meta name="security" content="exalead:group:Admin" />
<meta name="security" content="exalead:user:John Doe" />
<meta name="security" content="exalead:user:Jane Doe" />
```

4.3 Setting up a document filter

After the HTML files have been adjusted accordingly, a document filter is now set up in Exalead, which reads out the MetaTag created and assigns the content of the tag to the document as security information.

To do this, take the following steps:



DOCUMENT FILTERS **	FILTER KIND	
semantic	semantic	delete
www.firstspirit.de-Content	meta data	delete
TestC	meta data	delete
securityMETAFilter	meta data	delete

FILTERED SOURCES	
Aquavit	delete

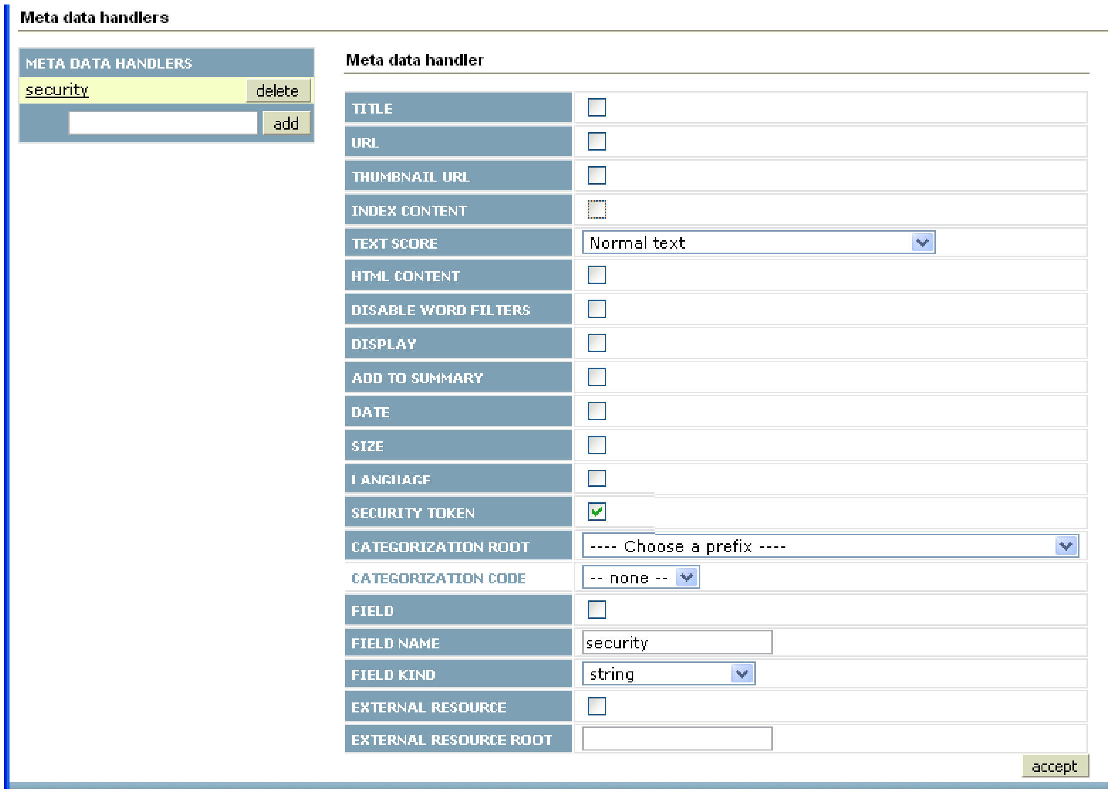
FILTER: securityMETAFilter

Figure 4-1: Document filters

- Create a new filter in the *Document filters* tab (e.g. securityMETAFilter). Assign type *"meta data"* in the "*FILTER KIND*" column and select the "*ENABLED*" checkbox.

- Add the name of the source to which this filter is to apply under "*FILTERED SOURCES*".

All kinds of different sources can be entered here, provided they work with the same security MetaTags, e.g. an HTML source and a Push-API source.

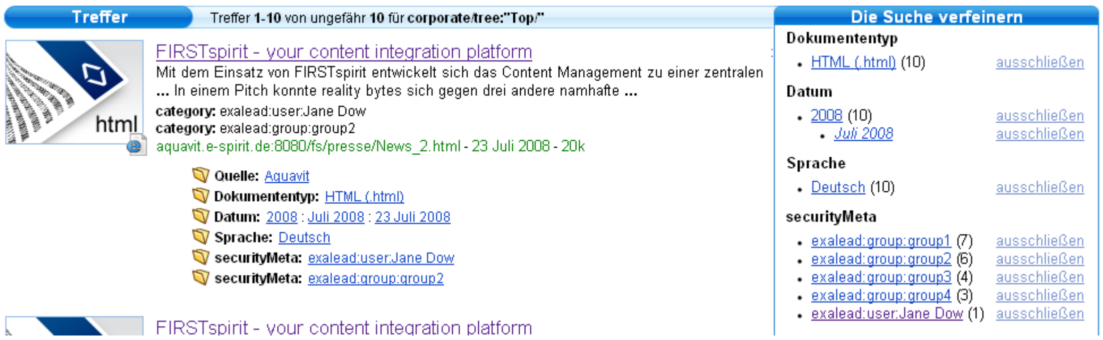


TITLE	☐
URL	☐
THUMBNAIL URL	☐
INDEX CONTENT	☐
TEXT SCORE	Normal text
HTML CONTENT	☐
DISABLE WORD FILTERS	☐
DISPLAY	☐
ADD TO SUMMARY	☐
DATE	☐
SIZE	☐
LANGUAGE	☐
SECURITY TOKEN	✓
CATEGORIZATION ROOT	---- Choose a prefix ----
CATEGORIZATION CODE	-- none --
FIELD	☐
FIELD NAME	security
FIELD KIND	string
EXTERNAL RESOURCE	☐
EXTERNAL RESOURCE ROOT	

Figure 4-2: Meta data handlers

- Now create a new handler under "*META DATA HANDLERS*, which bears the name of the MetaTag which is to be used as the security MetaTag in the HTML document (in the example above: security), and enabled the "*SECURITY TOKEN*" option on the right-hand side.

If required, the security MetaTags of the document found in the search hits of the individual hits (meta data handler → "DISPLAY" option) or in the search overview of the hit pool (Meta data handler → "ADD TO SUMMARY" option) can be displayed, so that searching and grouping search hits by permissions is possible or testing and checking the results of the search engine (see Figure 4-3) is made easier.


Treffer | Treffer **1-10** von ungefähr **10** für **corporate/tree:"Top/"**

FIRSTspirit - your content integration platform
Mit dem Einsatz von FIRSTspirit entwickelt sich das Content Management zu einer zentralen
... In einem Pitch konnte reality bytes sich gegen drei andere namhafte ...
category: exalead:user:Jane Dow
category: exalead:group:group2
aquavit.e-spirit.de:8080/fs/presse/News_2.html - 23 Juli 2008 - 20k

Quelle: Aquavit
Dokumententyp: HTML (.html)
Datum: 2008 : Juli 2008 : 23 Juli 2008
Sprache: Deutsch
securityMeta: exalead:user:Jane Dow
securityMeta: exalead:group:group2

FIRSTspirit - your content integration platform

Die Suche verfeinern

Dokumententyp
- HTML (.html) (10) ausschließen

Datum
- 2008 (10) ausschließen
  - Juli 2008 ausschließen

Sprache
- Deutsch (10) ausschließen

securityMeta
- exalead:group:group1 (7) ausschließen
- exalead:group:group2 (6) ausschließen
- exalead:group:group3 (4) ausschließen
- exalead:group:group4 (3) ausschließen
- exalead:user:Jane Dow (1) ausschließen

Figure 4-3: Search View (permissions)

- Finally, the changes made here are saved using the "accept" button, then "Commit Changes".

After completing these steps, the document filter created is now applied in future indexing of the documents; this filter reads out the users and groups from the designated security MetaTag and assigns them to the document as user and group guidelines.

4.4 Personalised search

As a default, sources are created in Exalead with access type "public". As a result, no permissions are compared during the search and therefore all documents are always displayed to all users.

To take into account the permissions read out in Chapter 4.3, it is necessary to set the access type of the source to be protected to `custom`. To do this, click `default` to set the value for "*Security*" in the "*Sources*" area to the new value `custom`.

No further configuration is necessary in relation to use of the "FirstSpirit Personalisation" module. After logging into the web applications, with each search query the user and group information of the logged in user is automatically evaluated and therefore only the search hits the user may see on the basis of their permissions are displayed.

For information on special treatment of permissions for file system sources, please refer to Chapter 5.4 page 68.

5 Using the Push-API

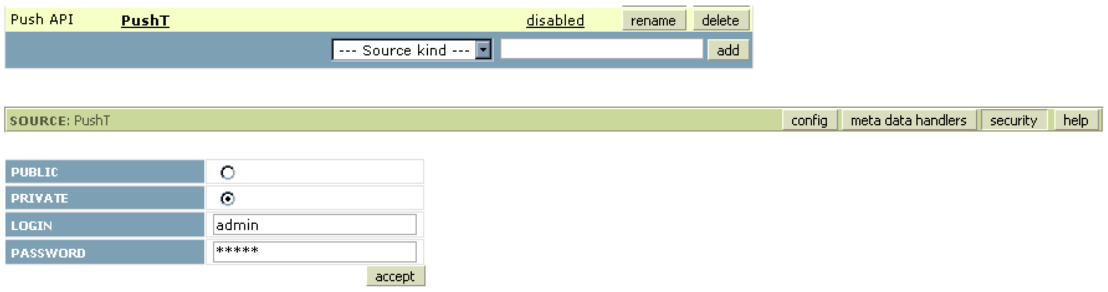
The Push-API provides an interface which can be used to directly write ("push") documents in the index of the search engine. As a result, changes to documents can be made immediately known to the index, without having to wait for the next crawling occasion.

A Java helper class for performing the operations necessary for this is located in the `exalead.jar` file, which is part of the FirstSpirit Exalead Live module.

In the following, an example is used to explain the steps necessary to configure the Push-API and the available helper class methods.

The Push-API connector automatically recognises, on the basis of the META handler settings (see Chapter 4.3 page 57), whether a passed document is secured or not. Security mode under sources can therefore not be changed.

5.1 Setting up the Push-API source



PUBLIC	○
PRIVATE	●
LOGIN	admin
PASSWORD	*****

Figure 5-1: Push-API security

To set up the Push-API connector, simply select a suitable name in the "Sources" tab in the "Sources" section and select the "Push API" as the "Source kind".

In the "Sources" tab, under "security", it is possible to specify whether the Push-API Connector is public ("PUBLIC" option) or whether a LOGIN / PASSWORD must be used in the connection string to pass data to it ("PRIVATE" option).

In general, the source should be secured by means of a login and password.

5.2 The Java Push-API helper class

5.2.1 Initialisation

The helper class must now be initialised first in order to use the Push-API via Java:

```
de.espirit.ps.exalead.pushapi.PushAPIHelper pHelper = new
de.espirit.ps.exalead.pushapi.PushAPIHelper("myserver", "10011", "PushT",
"admin", "admin");
```

The first parameter of the constructor stands for the server name and the second parameter for the port (`baseport + 11` is the default port of the push service). The third parameter gives the name of the created Push-API source. The third and fourth parameters give the assigned login data (are not taken into account in the `public` security setting).

5.2.2 Public methods

5.2.2.1 sendDocument(PushAPIDocumentValues docValues)

This method passes any individual document (local file system or HTTP resource) into the index. To do this, a PushAPIDocumentValues object is generated first, which must be filled with appropriate values.

Example: Push a file from the local file system

```
PushAPIDocumentValues pav = new PushAPIDocumentValues();


// Mandatory input: URI = Storage location of the file
pav.setUri("E:/PushTest/MyDocument.pdf");
// Mandatory input: PublicURL = Publicly available path
pav.setPublicURL("http://www.myserver.de/media/pdf/MyDocument.pdf");


// Optional input:
pav.setAuthor("Me");
pav.setLanguage("de"); // indication in lower cases
pav.setTitle("My document");


// Pushing the document
pHelper.sendDocument(pav);
```

Example: Push an HTTP resource

```
PushAPIDocumentValues pav = new PushAPIDocumentValues();


// Mandatory input: URI = Storage location of the page
pav.setUri("http://www.myserver.de/content/de/index.html");
// Mandatory input: PublicURL = Publicly available path
pav.setPublicURL("http://www.myserver.de/content/de/index.html");


// Optional input:
pav.setAuthor("Me");
pav.setLanguage("de"); // indication in lower cases
pav.setTitle("My document");


// Pushing the document
pHelper.sendDocument(pav);
```

In addition, it is also possible to directly specify the content of a document. To do this, an additional byte array can be passed to the PushAPIDocumentValues object. On pushing, the contents of the document are no longer read from the location specified as the URI and the contents of the byte array are passed instead.

Example: Push a byte array

```
PushAPIDocumentValues pav = new PushAPIDocumentValues();


// Mandatory input: content as byte array
pav.setContent(myByteArray);
// Mandatory input: URI (is required internally in spite of indication of
the byte array)
pav.setUri("http://www.myserver.de/myByteArray.html ");
// Mandatory input: PublicURL
pav.setPublicURL("http://www.myserver.de/myByteArray.html");
// Mandatory input: Date
pav.setDate(new Date());
// Mandatory input: Filename = name of the document
pav.setFilename("myByteArray.html");
// Mandatory input: Stamp = Unique indication for identifying changes made
at the document, e.g. character string built from creation date and length
of the document
pav.setStamp((new Date()).toString + myByteArray.length);
```

```
// Optional input:
pav.setAuthor("Me");
pav.setLanguage("de"); // indication in lower cases
pav.setTitle("My Byte Array Document");


// Pushing the document
pHelper.sendDocument(pav);
```

If a language abbreviation is not set using the setLanguage() method, Exalead performs an internal content check to try to establish the language of the document. If you want to prevent this for documents to which a language has not been assigned, the language can be set to unknown using setLanguage("xx"). Exalead will then no longer check the contents of the document for an ascertainable language.

5.2.2.2 sendDirectory(PushAPIDocumentValues docValues, boolean recursive)

This method passes the complete contents of a directory into the index. Permissions are not set. It is possible to specify whether this is to apply recursively to all subdirectories or for the specified directory only.

Example: Push a directory

```
PushAPIDocumentValues pav = new PushAPIDocumentValues();


// Mandatory input: URI = the directory which is to be pushed
pav.setUri("E:/PushTest/dir");
// Mandatory input: PublicURL = publicly available path
pav.setPublicURL("http://www.myserver.de/content/" +
PushAPIHelper.PUBLIC_URL_PLACEHOLDER);

// Recursive push of the directory
pHelper.sendDirectory(pav, true);
```

As, on pushing a complete directory, it is not possible to specify the PublicURL for each individual document, in this case it is also possible to use a wildcard, which is then replaced by the file name of the respective document when the individual documents are pushed. The file name is always the complete path, which begins after the given URI.

Example:

The following file structure exists:

E:\PushTest\dir\a.html
E:\PushTest\dir\b.html
E:\PushTest\dir\subdir1\c.html
E:\PushTest\dir\subdir1\d.html
E:\PushTest\dir\subdir1\subdir2\e.html
E:\PushTest\dir\subdir1\subdir2\f.html

The PublicURLs of these documents are then as follows:

http://www.myserver.de/content/a.html
http://www.myserver.de/content/b.html
http://www.myserver.de/content/subdir1/c.html
http://www.myserver.de/content/subdir1/d.html
http://www.myserver.de/content/subdir1/subdir2/e.html
http://www.myserver.de/content/subdir1/subdir2/f.html

5.2.2.3 resetAllDocuments()

This method is used to remove all pushed documents from the index.

5.2.2.4 long getTransferRate()

This method can be used to query the transfer rate in kb/s of all documents pushed since initialisation of the helper class.

5.2.2.5 Date getGlobalStartDate()

Date on which the helper class is initialised

5.2.2.6 long getGlobalByteCounter()

Bytes transferred since initialisation of the helper class

5.2.2.7 long getErrorCounter()

Number of errors which have occurred since initialisation of the helper class

5.2.2.8 long getDocCounter()

Number of documents pushed since initialisation of the helper class

5.2.2.9 resetCounters()

Resets all counters and resets GlobalStartDate to the current date.

5.2.2.10 setMaxFileSize(int maxFileSize)

Maximum size of the documents to be pushed in bytes. Documents which exceed this value are not written in the index.

5.2.2.11 int getMaxFileSize()

Query the set maximum size

5.2.2.12 setCheckpoint()

Calling this method causes Exalead to save the changes to the index and to close the transaction. However, it is not the same as a commit, as it does not cause the changes to be immediately reflected in the search results too.

5.2.2.13 setUserPrefix(String userPrefix)

This method is used to inform the Push-API helper class which prefix it should use on setting the user permissions. The same value as the one used in the configuration of the SearchServlet should be used (see Chapter 3.7.1).

5.2.2.14 setGroupsPrefix(String groupsPrefix)

This method is used to inform the Push-API helper class which prefix it should use on setting the group permissions. The same value as the one used in the configuration of the SearchServlet should be used (see Chapter 3.7.1).

5.2.2.15 setNoIndexStart(String noIndexStart)

This method overwrites the default value for marking the beginning of an area which is not to be indexed within an HTML document. The default value is:

```
<!-- NOINDEX -->
```

5.2.2.16 setIndexStart(String indexStart)

This method overwrites the default value for marking the end of an area which is not to be indexed within an HTML document (= start of the area which is to be indexed again). The default value is:

```
<!-- INDEX -->
```

5.2.2.17 setTextTypeExtensions(String extensions)

This method is used to indicate a list of filename extensions, separated by comma, of documents saved in the text format. Only for those documents areas can be defined which are to be excluded from the indexing. The default value is:

```
html,jsp
```

5.3 Assigning permissions via the Push-API

If you would like to assign documents permissions on pushing them via the Push-API, this is done by setting the permissions in the corresponding PushAPIDocumentValues object:

```
List<String> users = new ArrayList<String>();
users.add("user1");
users.add("user2");
pav.setUsers(users);

List<String> groups = new ArrayList<String>();
groups.add("group3");
groups.add("group4");
pav.setGroups(groups);
```

In this case, unlike setting permissions via MetaTags (see Chapter 4.2), a prefix is not specified, as this is automatically added by the Push-API helper class (see Chapter 5.2.2.13 and 5.2.2.14).

5.4 Using SIDs

When using a file system source, on indexing the files Exalead automatically reads out the corresponding permissions and assigns them to the appropriate file. These automatically read out permissions are now available in SID format instead of the already mentioned `exalead:user:-` or `exalead:group:` format. In order for a user logged in via FirstSpirit Personalisation to be able to see documents protected by SID, their SID and the SIDs of their groups must be determined first. This task is performed by the `getSIDs` tag, which has to be called at the beginning of a page (after the `authorize` tag of FirstSpirit Personalisation).

Example:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib prefix="search" uri="/WEB-INF/exalead.tld" %>
<%@ taglib prefix="perso" uri="/WEB-INF/firstpersonalisation.tld" %>

<perso:authorize/>
<search:getSIDs/>

<html>
<head>

…
```

As the `getSIDs` tag sets up a link with the Active Directory and then has to perform a search for the current user, it needs several configuration parameters, which have to be specified in the web.xml as context parameters:

```
<!-- AD configuration for reading the SIDs -->
<context-param>
          <param-name>ldapURL</param-name>
          <param-value>ldap://myadserver:389</param-value>
</context-param>

<context-param>
          <param-name>adminName</param-name>
          <param-value>cn=admin,cn=users,dc=mycompany,dc=de</param-value>
</context-param>
```

```
<context-param>
         <param-name>adminPassword</param-name>
         <param-value>mypassword</param-value>
</context-param>

<context-param>
         <param-name>searchBase</param-name>
         <param-value>cn=users,dc=mycompany,dc=de</param-value>
</context-param>

<context-param>
         <param-name>searchFilter</param-name>
         <param-value>(sAMAccountName=$USER_LOGIN$)</param-value>
</context-param>
```

The ldapURL, adminName and adminPassword parameters are used to set up the link with the Active Directory. The searchBase parameter specifies the start node in the Active Directory, from which the search for a user is to be made. The searchFilter parameter specifies a filter which is to be used in the search for the user. The `$USER_LOGIN$` character string is replaced with the name of the currently logged in user.

5.5 Excluding areas from indexing

Special areas within documents which are added to the index via the Push-API can be excluded from being indexed by using special NoIndex tags. This may be suggestive, for example, for the navigation of an HTML page, which should not be indexed.

The beginning of an area which is not to be indexed must be indicated by the following tag:

```
<!-- NOINDEX -->
```

The end of such an area is indicated by the following tag:

```
<!-- INDEX -->
```

Note: These tags are the default setting and can be overwritten by using the according methods of the Push API helper class (see Chapter 5.2.2.15 and Chapter 5.2.2.16).

Note: Such areas can only be identified in documents which are stored in the text and not in the binary format. For this reason it is necessary to announce such file types which are existent in the text format to the Push API helper class by using the according method (see Chapter 5.2.2.17).

6 FirstSpirit Module: Exalead Content-Update

6.1 Use

The FirstSpirit "Exalead Content Update" module is used to write content generated in FirstSpirit in the search index by using the Push-API immediately after they have been generated, so that this content is not indexed with the next crawler run, but instead can be indexed immediately.

If permissions assigned via metadata exist, they are adopted and written in the index together with the respective document. This requires installation of the PermissionService in FirstSpirit.

In addition, FirstSpirit Personalisation is required to enable personalised search queries to be set on the web server side, which use the permissions from the metadata of the PermissionService. Here it must be noted that the group names of the PermissionService must correspond to the group names in FirstSpirit Personalisation.

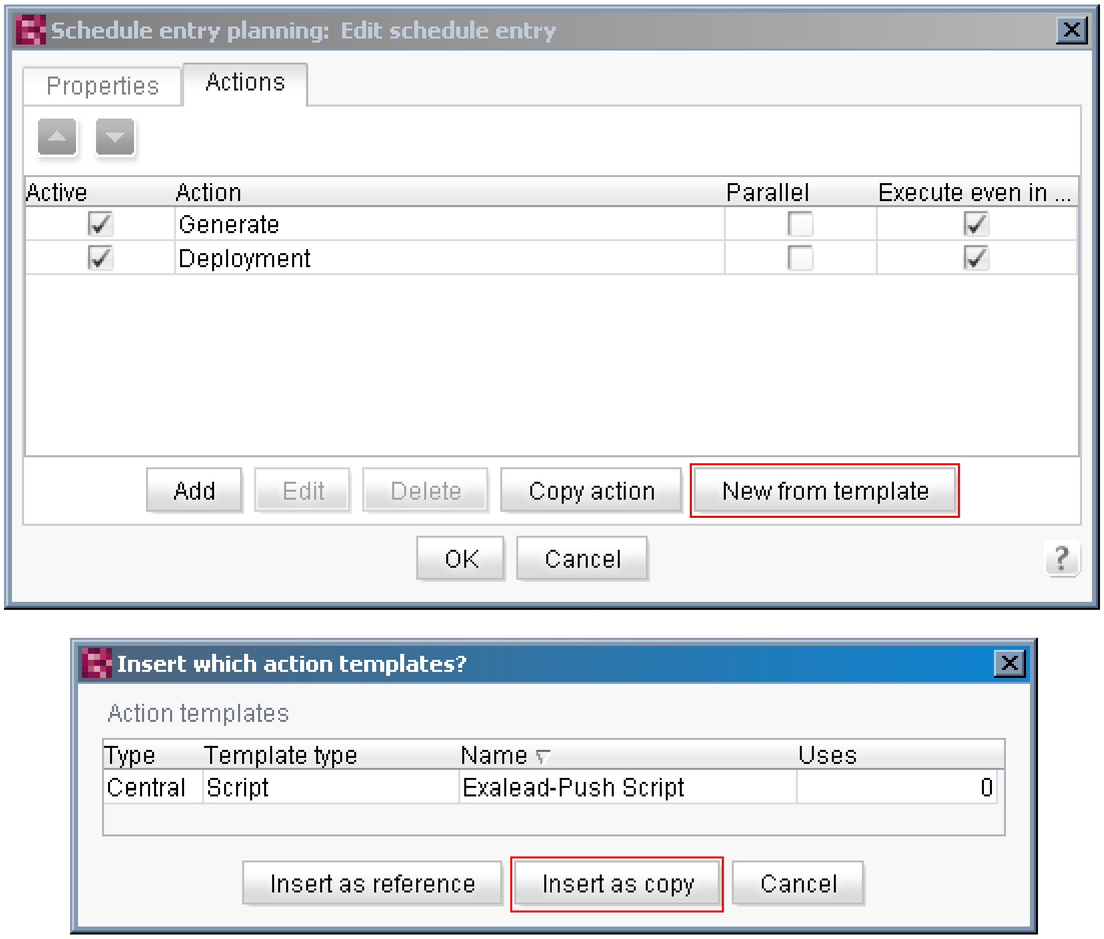
6.2 Installation

To install the Exalead Content-Update module, it is only necessary to load the corresponding FSM file onto the FirstSpirit server. The installation is carried out as described in the Manual for Administrators.

When the module is installed, an action template is automatically created, which can be used within a schedule to generate the action for calling the Push-API helper class.

The action generated in this way contains a script, which initiates transfer of the generated content into the search index. This script must be called within a schedule following a generation.

Note: Within the generation action it is necessary to ensure that use of the ACL database has been enabled there.



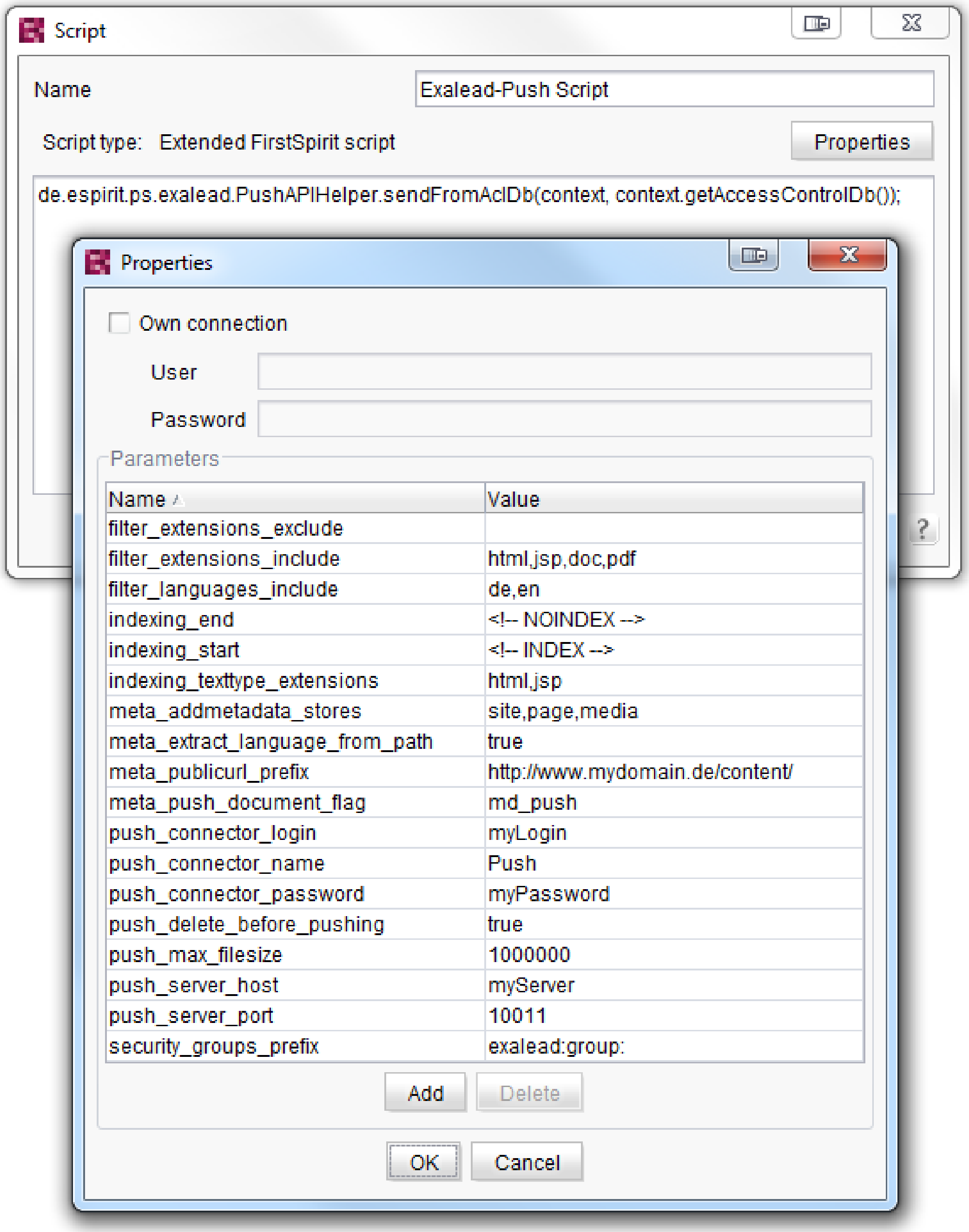
Active	Action	Parallel	Execute even in ...
✓	Generate	☐	✓
✓	Deployment	☐	✓

Type	Template type	Name	Uses
Central	Script	Exalead-Push Script	0

Figure 6-1: Adding the action template

6.3 Configuration

The script is configured using the script properties, as can be seen in Figure 6-2.



Name Exalead-Push Script

Script type: Extended FirstSpirit script

```
de.espirit.ps.exalead.PushAPIHelper.sendFromAclDb(context, context.getAccessControlDb());
```

☐ Own connection

User

Password

Parameters

Name	Value
filter_extensions_exclude	
filter_extensions_include	html,jsp,doc,pdf
filter_languages_include	de,en
indexing_end	<!-- NOINDEX -->
indexing_start	<!-- INDEX -->
indexing_texttype_extensions	html,jsp
meta_addmetadata_stores	site,page,media
meta_extract_language_from_path	true
meta_publicurl_prefix	http://www.mydomain.de/content/
meta_push_document_flag	md_push
push_connector_login	myLogin
push_connector_name	Push
push_connector_password	myPassword
push_delete_before_pushing	true
push_max_filesize	1000000
push_server_host	myServer
push_server_port	10011
security_groups_prefix	exalead:group:

Figure 6-2: Script properties

`filter_extensions_exclude`

This parameter accepts a comma-separate list of file types which are not to be written in the search index. This parameter is only evaluated if `filter_extensions_include` is empty.

If both `filter_extensions_include` and `filter_extensions_exclude` are empty, all generated files (including referenced pictures, stylesheets, etc.) are written in the search index.

`filter_extensions_include`

This parameter accepts a comma-separated list of file types. If this list is not empty, only files which correspond to this list are written into the search index. If the list is empty, the `filter_extensions_exclude` parameter is also evaluated.

`filter_languages_include`

The use of this parameter is only meaningful in combination with the use of `meta_extract_language_from_path`. You can use this parameter to specify a comma separated list of language abbreviations. Only pages and documents in these languages will be put into the index. Documents which are not assigned to any language (language-independent) can be selected by the abbreviation 'xx'.

Examples:

- No language is specified: All documents will be put into the index.
- `en`: Only documents of the language 'en' will be put into the index.
- `de,en`: Documents of the languages 'de' and 'en' will be put into the index.
- `xx`: Only language-independent documents will be put into the index.
- `en,xx`: Documents of the language 'en' and language-independent documents will be put into the index.

`indexing_end`

This parameter sets the string for marking the beginning of an area which is not to be indexed. The default value is `<!-- NOINDEX -->`

`indexing_start`

This parameter sets the string for marking the end of an area which is not to be indexed. The default value is `<!-- INDEX -->`

`indexing_texttype_extensions`

List of filename extensions, separated by comma, of documents saved in the text format. Only for those documents areas can be defined which are to be excluded from the indexing. The default value is: `html,jsp`

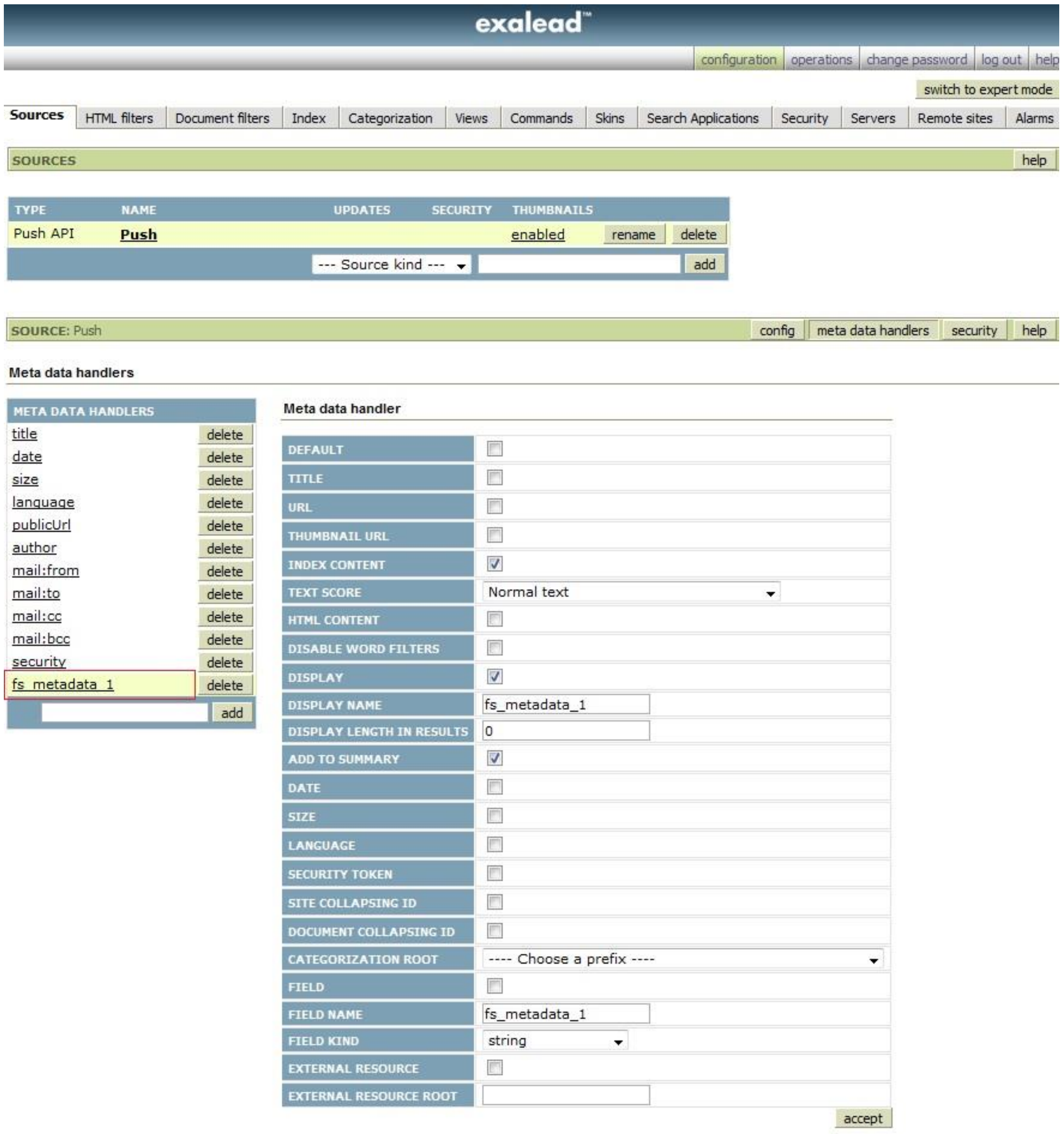
`meta_addmetadata_stores`

Use this parameter to define if metadata are to be put to the index in addition to the content of documents. The value of this parameter indicates the FirstSpirit store from which the metadata are to be taken for the respective document: `site` (Site Store), `media` (Media Store) or `page` (Page Store). These values can be combined, if they are separated by a comma.

Examples:

- `site` = The metadata of the Site Store will be used.
- `media,page` = The metadata of the Media and of the Page Store will be used and – if necessary – combined.

The Exalead server needs to be configured adequately for being able to exploit the metadata:



TYPE	NAME	UPDATES	SECURITY	THUMBNAILS		
Push API	Push			enabled	rename	delete

Meta data handlers

META DATA HANDLERS	
title	delete
date	delete
size	delete
language	delete
publicUrl	delete
author	delete
mail:from	delete
mail:to	delete
mail:cc	delete
mail:bcc	delete
security	delete
fs_metadata_1	delete
	add

Meta data handler

DEFAULT	☐
TITLE	☐
URL	☐
THUMBNAIL URL	☐
INDEX CONTENT	✓
TEXT SCORE	Normal text
HTML CONTENT	☐
DISABLE WORD FILTERS	☐
DISPLAY	✓
DISPLAY NAME	fs_metadata_1
DISPLAY LENGTH IN RESULTS	0
ADD TO SUMMARY	✓
DATE	☐
SIZE	☐
LANGUAGE	☐
SECURITY TOKEN	☐
SITE COLLAPSING ID	☐
DOCUMENT COLLAPSING ID	☐
CATEGORIZATION ROOT	---- Choose a prefix ----
FIELD	☐
FIELD NAME	fs_metadata_1
FIELD KIND	string
EXTERNAL RESOURCE	☐
EXTERNAL RESOURCE ROOT	

Figure 6-3: Meta data – Setting of the Exalead server

The name of the *Meta data handler* must exactly correspond to the variable name of the meta date as configured in FirstSpirit.

The following FirstSpirit input components are supported:

- `CMS_INPUT_TEXT`
- `CMS_INPUT_DATE`
- `CMS_INPUT_COMBOBOX`
- `CMS_INPUT_CHECKBOX`
- `CMS_INPUT_LIST`

- `CMS_INPUT_TOGGLE`

In the case of `CMS_INPUT_TEXT` and `CMS_INPUT_DATE` the content of the input component will be transferred as value to Exalead. In the case of `CMS_INPUT_COMBOBOX`, `CMS_INPUT_LIST` and `CMS_INPUT_CHECKBOX` the label of the respective selection which is defined in the metadata template will be transferred as value to Exalead. In the case of `CMS_INPUT_TOGGLE` the value `true` will be transferred if the component is selected.

`meta_extract_language_from_path`

This parameter can be used to control whether the document language is to be determined using the path of the generated file. If this parameter is not set (or the value `false` is transferred), the Exalead server uses an internal algorithm to decide which language is assigned to a document. By setting this parameter to the value `true`, the language is determined using the language abbreviation found in the file path:

/de/content/start.html	->	Language: German
/en/content/start.html	->	Language: English
/de_1/content.pdf	->	Language: German
/media/en/docs/flyer.pdf	->	Language: English
/media/docs/flyer.pdf	->	no assigned language

However, if using automatic extraction of the document language on the basis of the file path, it must be noted that the name of each directory consisting of two letters located directly below the media directory is interpreted as the language of the documents contained in the directory.

Example:

/media/sg/styleguide.pdf	->	Language: sg

Better:

/media/guides/styleguide.pdf	->	no assigned language

In the case of documents without an assigned language, Exalead performs the analysis and assigns the document to a language.

Extracting the language from the path of a generated file is only possible when 'Default URLs' is chosen for PathGeneration in the generate task.

`meta_publicurl_prefix`

This parameter is used to give the prefix for the paths of the deployed files. At the same time, the generation directory in the absolute path of the generated files is replaced by this prefix, when the files are written in the search index. Thus, in the example above, the path:

`<Generation_directory>/en/company/contact.jsp`

becomes the path:

http://www.mydomain.en/content/en/company/contact.jsp

`meta_push_document_flag`

Use this optional parameter to specify the name of a metadata input component in FirstSpirit, which enables you to exclude individual documents (pages / media) from indexing.

The input component must be of the type `CMS_INPUT_TOGGLE`. If the checkbox is selected, the related document will not be indexed.

Please notice, that only values of this input component in the Site or Media Store will be evaluated, selected checkboxes of this input component in the Page Store will not be taken into account for the indexing.

Example configuration in the metadata template:

```
<CMS_INPUT_TOGGLE name="md_push" singleLine="no">

    <LANGINFOS>

        <LANGINFO lang="*" label="Disable push"/>

    </LANGINFOS>

</CMS_INPUT_TOGGLE>
```

`push_connector_login`

Login for access to the push connector (an Exalead server setting) `push_connector_name`

Name of the push connector (an Exalead server setting)

`push_connector_password`

Password for access to the push connector (an Exalead server setting)

`push_delete_before_pushing`

This parameter can be used to control whether all documents written in the search index to date are removed before a renewed write process. This is recommended for a full generation; however, this should be disabled for a partial generation.

`push_max_filesize`

Use this optional parameter to define a maximum size (in bytes) for documents which are to be transferred to the Exalead server. If the parameter is not given, or no value is defined for it, the default value will be used (10485760 Bytes = 10 MB).

`push_server_host`

Name of the Exalead server on the network

`push_server_port`

Port, under which the push connector can be reached (Exalead server setting)

`security_groups_prefix`

This parameter is used to inform the script which prefix it should use on setting the group permissions. The same value as the one used in the configuration of the SearchServlet should be used (see Chapter 3.7.1).

6.4 Removing single documents from the index

The Exalead Content-Update module provides a service that can be used within the FirstSpirit client to remove single documents from the index that were previously pushed to the index.

To use this service it's necessary to first obtain an instance of the service:

```
exaleadService =
context.getConnection().getService("EnterpriseSearchService");
```

The next step will be to initialize the connection to the Exalead server:

```
exaleadService.initPAPI(String exalead_hostname, String port, String
push_source, String push_login, String push_password);
```

The values of each parameter are:

- `exalead_hostname =` name of the Exalead server.
- `port` = Push-API port of the Exalead server (baseport + 11, e.g.: 10011)
- `push_source =` name of the used push source
- `push_login =` login for the push source (an empty string if the push source is not secured)
- `push_password =` password for the push source (an empty string if the push source is not secured)

Example:

```
exaleadService.initPAPI(myServer, 10011, Push, myLogin, myPassword);
```

After initialization the service can be used to remove single documents from the index:

```
exaleadService.removeFromIndex(String document_path, String projectId);
```

The value of the parameter `document_path` has to be the generation path of the document and `projectId` is the project id of the project.

Example:

```
exaleadService.removeFromIndex("de/aboutus/company/company.jsp", 2574);
```

This method call has to be repeated for every project language and every output channel:

```
exaleadService.removeFromIndex("de/aboutus/company/company.jsp", 2574);
exaleadService.removeFromIndex("en/aboutus/company/company.jsp", 2574);
exaleadService.removeFromIndex("de_1/aboutus/company/company.jsp", 2574);
exaleadService.removeFromIndex("en_1/aboutus/company/company.jsp", 2574);
```

To apply the changes made (removing of documents from the index) it is necessary to perform a manual commit on the index:

```
exaleadService.commitChanges();
```

Without manual commit the changes will be applied during the next scheduled commit (configurable within the Exalead server).