Apache Solr for TYPO3 - File Indexing

Release 8.0.0

Steffen Ritter, Timo Hund, Markus Friedrich, Rafael Kähm

2020-12-22 14:48

CONTENTS

EXT:solrfal is an addon to EXT:solr (>= 3.0) which allows to index files using the FileAbstractionLayer of TYPO3. solrfal is meant to work as of TYPO3 6.2 LTS and later.

Note: Please note that this extension is currently part of the Early Access Program (EAP). The EAP funds the dedicated developement of the core extension, the implementation of new solr server features and the maintenance. Please see: http://www.typo3-solr.com/en/solr-for-typo3/development-model/ if you want to contribute to the financial support.

CHAPTER

ONE

HOW TO USE

1. **Setup EXT:solr >=3.0.0**
    - configure page indexing and/or record indexing according to your needs
    - make sure you have the compatible EXT:solr version, please see version compatibility matrix
    - EXT:solrfal depends on system extension “filemetadata”, which will be activated automatically while activating the EXT:solrfal extension.
2. **Include the Static TypoScript** “Search - FAL File Indexing (solrfal)”
3. **Add scheduler task for EXT:solrfal**
    - The TASK “File Index Queue Worker” needs to be set up.
4. **Check scheduler TASKs for FAL in Core**
    - If you have external storages and/or it is possible that files change in a storage in ways, the TYPO3 backend is not involved it is mandatory to have the “File Abstraction Layer: Update storage index” running frequently (a.k.a every 5-10 minutes).
    - Solr uses the MetaData of FAL; if you have extractors for MetaData you also might run the “File Abstraction Layer: Extract metadata in storage” frequently.
5. Do your **individual TypoScript configuration**
6. **Setup EXT:tika** to enable search within file contents and metadata.

CHAPTER

TWO

CONCEPTS AND BACKGROUND INFORMATION

2.1 Indexing contexts

A context describes the “place” where a file is detected to be indexed.

EXT:solrfal knows three contexts of indexing.

- **PageContext**: files which are found during indexation of a frontend-page
- **RecordContext**:files which are attached to records (which are indexed by solr)
- **StorageContext**: files which just reside in a file-system of a storage

2.1.1 Site exclusive records

A file in fal can be referenced in any record across multiple sites. Therefore a contexts are checked for changes. In some cases we know, that a change will only effect the contexts of the current site, because the are only related to the current site by nature (e.g. pages, pages_language_overlay, tt_content, sys_files_references). When you want to configure tables to be treated the same way, you can configure the tables in “siteExclusiveRecordTables”.

2.2 Schema fields for files in Solr

EXT:solr extends the schema with file specific attributes which are automatically added to the solr document on indexing. See the following list of available and indexed fields:

- **fileStorage**: uid of the Storage the file resides in
- **fileUid**: uid of the file
- **fileMimeType**: the mimetype of the file
- **fileName**: filename (including extensions)
- **fileSize**: file size in bytes
- **fileExtension**: file extension (without .)
- **fileSha1**: content hash of the file (SHA1),
- **filePublicUrl**: publicUrl of the file object
- **fileReferenceType**: table name from which the file was referenced (e.g. tt_content, or tx_new)
- **fileReferenceUid**: uid of the record the file was referenced from

CHAPTER

THREE

TYPOSCRIPT REFERENCE

3.1 pageContext

Type Boolean

TS Path plugin.tx_solr.index.enableFileIndexing.pageContext

Default 1

Function enables indexing for files attached to Content-Element-Records while indexing the frontend pages

3.1.1 contentElementTypes

Type Array (key: CType from tt_content, value: fields in which file attachments should be extracted);

TS Path plugin.tx_solr.index.enableFileIndexing.pageContext.contentElementTypes

Default text => bodytext, header_link; textpic =>bodytext, header_link; uploads => media

Function For different content element types you may want files detected in different fields. The default configuration is to behave like solrfile.

If you have added a custom content element, you may want to configure fields here.

3.1.2 fileExtensions

Type String, comma separated values (file extensions without .) or *

TS Path plugin.tx_solr.index.enableFileIndexing.pageContext.fileExtensions

Default *

Function allows to restrict the files being indexed by the file extension

3.1.3 enableFields

Type array (of column names in sys_file_metadata)

TS Path plugin.tx_solr.index.enableFileIndexing.pageContext.enableFields

Allowed keys endtime, accessGroups

Function Use the enableFields from the Page it is referenced at for file

3.2 storageContext

Type boolean (1/0);

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext

Default 0

Function enables indexing of all files in a storage

3.2.1 [StorageUid]

Type array

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid]

Index affected StorageUid (f.e. fileadmin/ generally 1)

Function Enables a detailed indexing configuration per Storage, see properties for details

3.2.2 languages

Type string, comma separated values (integers list of sys_language uids)

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid].languages

Default 0

Function define for which languages this storge should be indexed

3.2.3 fileExtensions

Type string, comma separated values (file extensions without .) or *

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid].fileExtensions

Default *

Function allows to restrict the files being indexed by the file extension

3.2.4 enableFields

Type array (of column names in sys_file_metadata)

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid].enableFields

Allowed keys endtime, accessGroups

Function FAL generally does not have enable fields, but metadata ships fields which can be used for that purpose. With this configuration you define “enableFields” just for indexation.

3.2.5 folders

Type string, comma separated values

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid].folders

Default *

Function List of valid directories, relative to storage root directory

Since 3.1

3.2.6 excludeFolders

Type string, comma separated values

TS Path plugin.tx_solr.index.enableFileIndexing.storageContext.[StorageUid].excludeFolders

Default

Function List of directories to exclude, relative to storage root directory

Since 3.1

3.3 recordContext

Type boolean

TS Path plugin.tx_solr.index.enableFileIndexing.recordContext

Default 1

Function enables indexing of all file attachment at records;

needs further configuration in the index queue: a, table needs to indexed at all, b, attachment indexation needs to be activated for that table

3.4 attachments

Type boolean

TS Path plugin.tx_solr.index.queue.[indexingConfiguration].attachments

Default 0

Function enables file attachment detection when indexing the record

3.5 fields

Type string, comma separated values (column names of tables)

TS Path plugin.tx_solr.index.queue.[indexingConfiguration].attachments.fields

Default *

Function define in which columns of an record files should be detected

3.6 plugin.tx_solr.index.queue._FILES

Configuration array to configure the index.queue processing for files. The configuration will be merged. This means that every context specific configuration inherits the default configuration. In addition, if there is a special configuration within context (like per table or storage) these will inherit the base configuration of the context. Each configuration is to be defined like plugin.tx_solr.index.queue.[indexingConfiguration].fields

3.7 default

Type array

TS Path plugin.tx_solr.index.queue._FILES.default

Function Mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file indexed

Default

```
title = title
description = description
altText_stringS = alternative
width_intS = width
height_intS = height

category_stringM = SOLR_RELATION
category_stringM {
    localField = categories
    foreignLabelField = uid
    enableRecursiveValueResolution = 1
    multiValue = 1
}
```

3.8 pageContext

Type array

TS Path plugin.tx_solr.index.queue._FILES.pageContext

Function Additional mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file indexed in pageContext

3.9 storageContext

Type array

TS Path plugin.tx_solr.index.queue._FILES.storageContext

Default empty

Function See introduction and following two entries.

3.10 default

Type array

TS Path plugin.tx_solr.index.queue._FILES.storageContext.default

Function Additional mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file indexed in storageContext

3.11 [StorageUid]

Type array

TS Path plugin.tx_solr.index.queue._FILES.storageContext.[StorageUid]

Function Additional mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file of Storage [StorageUid] indexed in storageContext

3.12 recordContext

Type array

TS Path plugin.tx_solr.index.queue._FILES.recordContext

Default empty

Function See introduction and following two entries.

3.13 default

Type array

TS Path plugin.tx_solr.index.queue._FILES.recordContext.default

Function Additional mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file indexed as attachment in recordContext

3.14 [__tablename__]

Type array

TS Path plugin.tx_solr.index.queue._FILES.recordContext.[__tablename__]

Function Additional mapping of Solr field names on the left side to database table field names or content objects on the right side. Used for every file found attached to record of [__tablename__] indexed in recordContext

CHAPTER

FOUR

AVAILABLE SIGNALS

In case you need to adapt or extend the behaviour of solrfal the following signals exist you may consume in your slots.

4.1 DocumentFactory::fileMetaDataRetrieved

Shortly after the (translated) MetaData record of a file is retrieved this Signal is emitted. The slot must take the IndexQueueItem as first parameter and an ArrayObject as second parameter. The ArrayObject contains the metadata you may modify in your slot. The modified MetaData will then be hand over to the TypoScript “Service”. As an result fields added in the slot to that Signal can be addressed from your regular TypoScript setup.