

Extension-Dokumentation

Extension Key: abavo_search

Language: de

Keywords: Suche, Indexierung, Synonym, Facettierung

Copyright (2015), abavo GmbH, <dev@abavo.de>

The content of this document is proprietary.

Inhalt

Extension-Dokumentation.....	1
Inhalt	2
Einführung.....	3
Was macht die Extension?	3
Screenshot	3
Administrations-Handbuch	4
Datensatzsammlung Organisation	4
Indexer-Konfiguration erstellen.....	4
Synonyme	7
Facettierte Suche	8
Die Frontend-Plug-In´s.....	10
Einrichten des Scheduler-Tasks.....	11
Extension-Hooks	12
Stop-Wörter	14
TypoScript-Referenz	15
TS-Setup	15
TS-Constants.....	15

Einführung

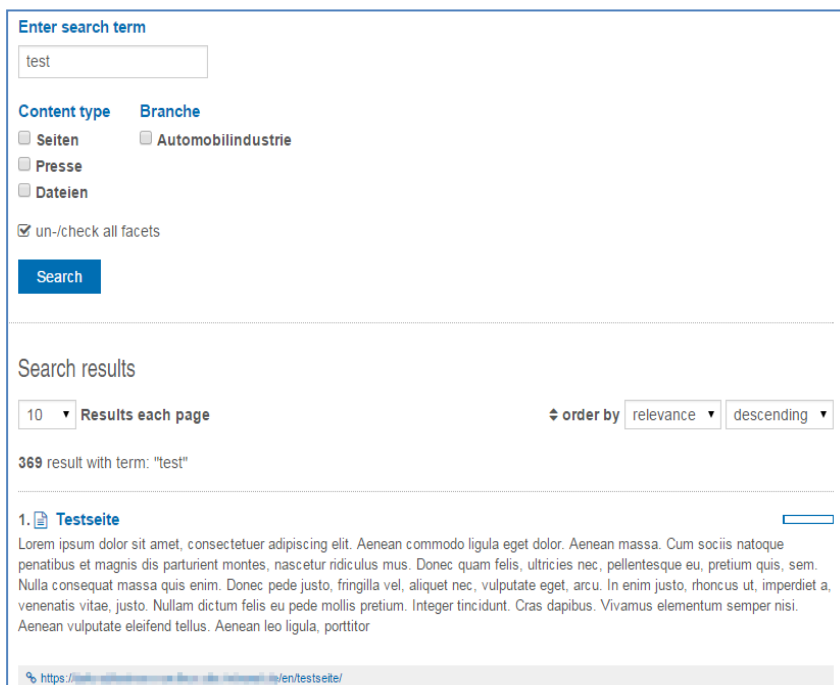
Was macht die Extension?

- Frontend-Suche für TYPO3-Inhalte und Office-Dokumente.
- Dabei kann die Basisfunktionalität flexibel und einfach mit wenigen Schritten erweitert werden.

Kern-Funktionen

- Nutzung des Extbase Frameworks und FLUID-Template-Engine
- Unterstützung von bestehenden und eigenen Stop-Wörtern über XML konfigurierbar
- Indexierung nicht global sondern abhängig von Konfigurationen in verschiedenen TYPO3-Sys-Ordnern
- Einfache Erweiterbarkeit über das Erstellen eigener Index-Hooks ohne Feldbeschränkung mit eigenen FlexForms
- Suche nach Begriffen ab 3 Zeichen (per TS-Setup konfigurierbar) möglich mit MySQL-Fallback „LIKE“ für „MATCH AGAINST“ mit MySQL
Variable 'ft_min_word_len', sollte eine Suchbegriffslänge unter dem nur serverseitig konfigurierbaren Limit sein (i.d.R. 4)
- Rückgabe der Dauer von Indexerstellung und Suchanfragen
- Bei Alternativ-Wörtern sind die Aufrufe auf 60 Anfragen/Minute auf den jeweiligen Besucher begrenzt
- Einbindung des Suchformulars unabhängig von Plugin über FLUID auch mehrmals auf einer Seite möglich
- Intelligentes, sich selbst bereinigendes, von Zugriffs-Berechtigung und sprachabhängiges Autocomplete (Autocomplete nur auf vorhandenen Index)
- Synonyme mit Reverse-Funktion und stilistischer Unterscheidung in den Suchergebnissen
- Facettierung über SysCategories

Screenshot



Enter search term

test

Content type Branche

☐ Seiten ☐ Automobilindustrie

☐ Presse

☐ Dateien


☒ un-/check all facets

Search

Search results

10 Results each page order by relevance descending

369 result with term: "test"

1.  **Testseite**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor

<https://www.abavo-search.com/en/testseite/>

Administrations-Handbuch

- Beschreibung der Konfigurationen und Verwendung der Datensätze und Plugins für TYPO3-Integratoren und fortgeschrittene Benutzer
- Einrichtung von Indexierungs-Jobs und Tipps zur Fehlerbehebung

Datensatzsammlung Organisation

- Erstellen Sie im TYPO3-Backend einen Systemordner, der als Datensatzsammlung und Basis für den Such-Index der später durchsucht, Synonyme und Indexer-Konfigurationen verwendet werden soll.

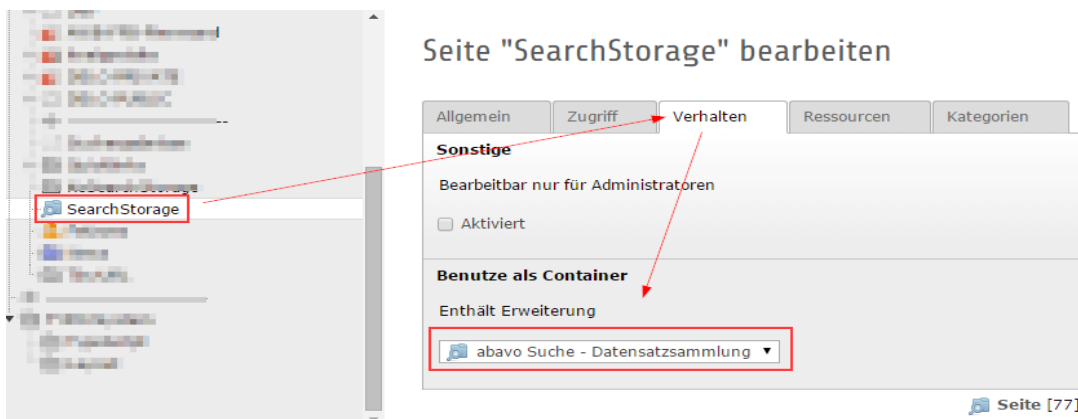
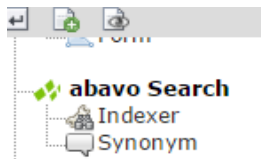


Abb.: Zuweisung als Typ „abavo Suche – Datensatzsammlung“

Indexer-Konfiguration erstellen

Über die Listenansicht können nun im zuvor erstellten Systemordner TYPO3-Typisch Datensätze des Typs „Indexer“ erstellt werden.



Basiskonfiguration

Allgemein **Erweitert**

Sichtbarkeit

Start:
 Stopp:
 Verbergen: ☐

Grundeinstellungen

Titel:
 Typ:
 SimpleCalendar Indexer
 Page Indexer
 FAL Indexer
 News Indexer

Search Storage PID:
 SearchStorage
☐ Seite
 Target-PID:
 Suchergebnisse
☐ Seite

Index zusätzliche Kategorien zuweisen

Element finden
 Kategorie

Sichtbarkeit

Standard-Sichtbarkeitseinstellungen

Grundeinstellungen

Titel: Bezeichnung der Index-Datensätze im FE (default L=0)

Typ: Auswahl aktiver Indexer-Hooks

Search Storage PID: Seite der def. Datensatzsammlung

Target-PID: Zielseite für Index-Content (z.B. News: Seite mit News-Detail-Plugin); Typ-Abhängig;

Index zusätzliche Kategorien zuweisen

Index-Datensätze erhalten automatisch die hier gewählten Kategorien zusätzlich zugewiesen.

Titel-Local-Mapping (zeilengetrennt; z.B: en=MyTitle)

en=Pages

Titel-Local-Mapping

Filter-Beschriftung local-bezogen, pro Sprache, zeilengetrennt im Frontend (erweiterte Suche)

Die Extension bringt als Basis bereits einige gängige Standard-Indexer-Hooks mit, die verwendet werden können. Der Bereich „Indexer FlexForm“ ist dynamisch je nach Indexer-Hook.

Seiten Indexer

Im Reiter „Erweitert“ werden die spezifischen Einstellungen getroffen.

Indexer FlexForm

Page Indexer Configuration

Indexer Class
PageIndexer

Website-Sprache

☒ Standard [0]
☒ English [1]

Allgemeine Datensatzsammlung:

Home

Seite

Additional CTypes comma separated (Defaults:text, textpic, image, bullets, table, html, header, uploads, shortcut)

Website-Sprache

Sprachen die für die Indexierung berücksichtigt werden sollen.

Allgemeine Datensatzsammlung

Von welchen Seiten ausgehend soll indexiert werden?

Additional cTypes

Erfahrene Benutzer können hier kommasepariert zusätzliche „CTypes“ für z.B. Eigenentwicklungen mit berücksichtigen.

Die FE-Gruppen einzelner Seiten werden automatisch mit übernommen, sowie Seiten-von-Seiten und Inhalt-von-Inhalt mit indexiert.

FAL Indexer (Datei-Abstraktionsschicht)

Basis Konfiguration

Indexer FlexForm

FAL Indexer Configuration

Indexer Class
FALIndexer

FAL Storage
fileadmin/ (auto-created)

Relative Paths comma separated
datasheet/, eigene_dateien/, doc/

Exclude relative Paths comma separated (use "/" as prefix)
/doc/subfolder/

File-Extensions

☒ PDF-Documents *.pdf (Needs module pdftotext,pdfinfo on server side)
☒ MS Word 97-2003 *.doc (Needs module catdoc on server side)
☒ MS Word Office Open XML *.docx (Needs module unzip,grep,sed on server side)
☒ MS Excel 97-2003 *.xls (Needs module xls2csv on server side)
☒ MS Excel Office Open XML *.xlsx (Needs module unzip,grep,sed on server side)
☒ MS Powerpoint 97-2003 *.ppt (Needs module catppt on server side)
☒ MS Powerpoint Office Open XML *.pptx (Needs module unzip,grep on server side)

FAL Storage

Auswahl des FAL-Mounts (aktuell wird nur das gleiche Dateisystemunterstützt)

Relative Paths

Kommaseparierte, relativ zum FAL-Mount Pfad liegende Verzeichnisse mit abschließendem „Slash“;

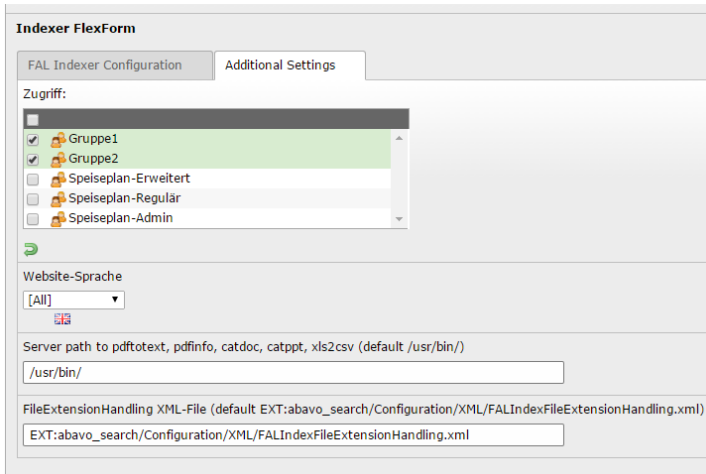
Exclude relative Paths

Unterverzeichnisse, der in „Relative Paths“ angegebenen Stammverzeichnisse, die von der Indexierung ausgeschlossen werden.

File-Extensions

Datei-Typen die aufgrund ihrer Datei-Endung berücksichtigt sind (Serveranforderungen stehen jeweils in Klammern)

Erweitert Einstellungen



FE-Benutzer-Gruppen

Suchergebnis auf FE-Gruppen einschränken.

Website-Sprache

Sprache, die der Index für die Indexierung berücksichtigt werden sollen.

Server Pfad zu Standard-Binarys

Falls von Standard abweichend, kann dieser Pfad angepasst werden.

FileExtensionHandling

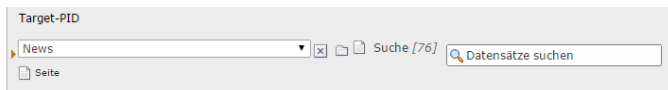
Dabei handelt es sich um einen Spezialfall. In dieser XML-Konfigurationsdatei wird mit regulären Ausdrücken bestimmt, wie Datei-Inhalte je nach Typ bereinigt werden. Diese XML kann outgesourct und somit customized werden.

Auszug FALIndexFileExtensionHandling.xml

```
<mapping id="docx">
  <script>EXT:abavo_search/Resources/Private/Scripts/catdocx.sh</script>
  <cleanRegex>
    <pattern>/([\n\t]|\xffJPG|\x89PNG|MERGEFORMAT)/</pattern>
    <replace> </replace>
  </cleanRegex>
</mapping>
```

News Indexer (Hook für Extension tx_news)

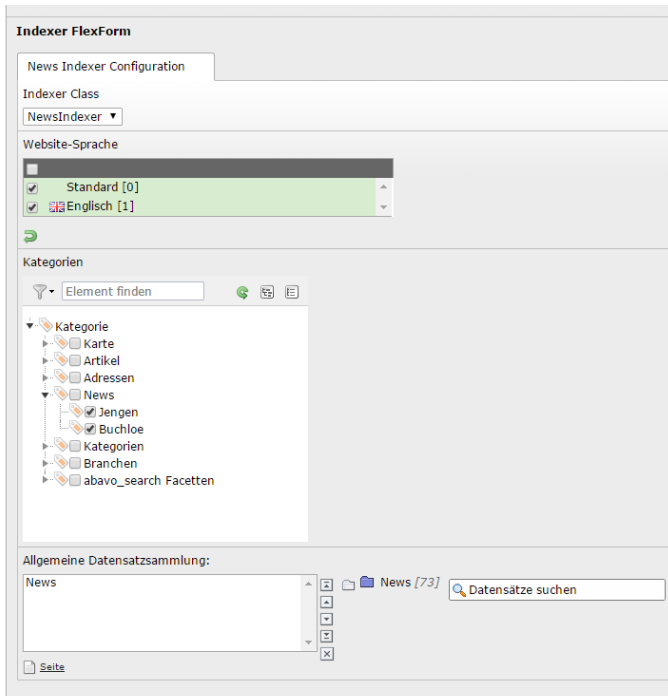
Dieser Hook besitzt ein zusätzliches Feld in der Basis-Konfiguration (Reiter Allgemein):



Target-PID
 News [x] Suche [76] Datensätze suchen
 Seite

Target-PID

Seite mit TX_NEWS-Detail-Ansicht Plugin



Indexer FlexForm

News Indexer Configuration

Indexer Class
 NewsIndexer

Website-Sprache
☒ Standard [0]
☒ Englisch [1]

Kategorien
 Element finden
 Kategorie
 Karte
 Artikel
 Adressen
 News
 Jungen
 Buchloe
 Kategorien
 Branchen
 abavo_search Facetten

Allgemeine Datensatzsammlung:
 News [x] News [73] Datensätze suchen
 Seite

Website-Sprache

Sprache, der die Index für die Indexierung berücksichtigt werden sollen.

News-Kategorien

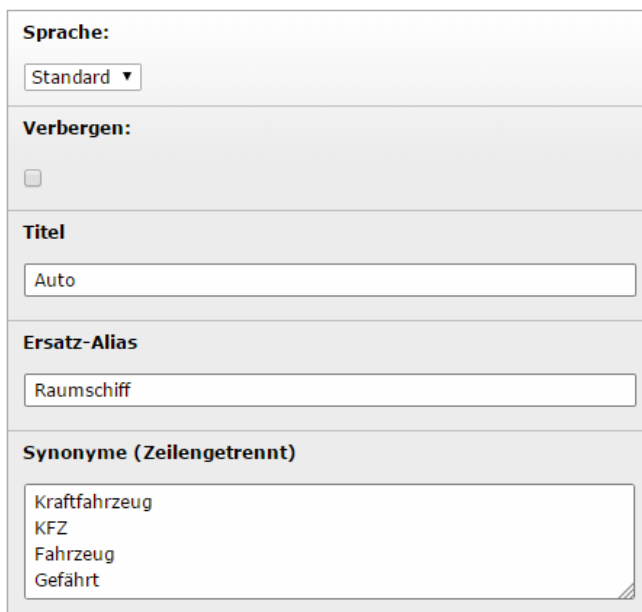
Es werden nur News-Datensätzen mit gewählten Kategorien Indexiert

News-Datensatzsammlung

Storage der News-Datensätze.

Synonyme

Die Suche unterstützt die Verwendung von Synonymen. Dies ermöglicht die Suche nach definierten Begriffen um weitere verwandte Suchbegriffe gezielt zu erweitern. Wenn nach einem Begriff gesucht wird, der Synonym-Datensatz angelegt wurde, im Bsp. Hier „Auto“, wird automatisch zusätzlich nach allen damit verbundenen Synonymen (bsp.: „Kraftfahrzeug, KFZ, Fahrzeug, Gefährt“) gesucht. Dies funktioniert auch umgekehrt (eine Suche nach „KFZ“ wird um „Auto, Kraftfahrzeug, Fahrzeug, Gefährt“ erweitert).



Sprache:
 Standard

Verbergen:
☐

Titel
 Auto

Ersatz-Alias
 Raumschiff

Synonyme (Zeilengetrennt)
 Kraftfahrzeug
 KFZ
 Fahrzeug
 Gefährt

Sprache

Für welche Sprache ist das Synonym gültig?
 Sprachübergreifende Begriffe werden unterstützt.

Titel

Hauptbegriff der ersetzt werden soll.

Ersatz-Alias

Ist ein Synonym(Titel) thematisch überholt und sollen die Synonyme bestehen bleiben, kann man diesen mit einem Alias ersetzen. (1 Generation)

Synonyme

Begriffe zeilengetrennt, nach denen zusätzlich gesucht werden soll.

Facettierte Suche

Das Frontend-Plug-In „*abavo Suche – Eingabeformular*“ unterstützt die Facettierte Suche ([siehe Plug-In's](#)).

Grundsätzlich werden als Facetten sämtliche „*Indexer-Datensätze*“ (Mehrfachdefinition des gleichen Indexer-Typs sind möglich) und Kindes-Kind-Elemente einer definierten „*Root-Kategorie*“ unterstützt.

Der TYPO3-Datensatztyp „*SysCategory*“ ermöglicht eine zusätzliche, benutzerdefinierte Facettierung von Suchergebnissen. Dazu muss im Storage-Folder der [Suche-Datensatzsammlung](#) eine „*Root-Kategorie*“ mit Sprache „*[Alle]*“ angelegt werden. Damit kann anschließend eine „*Facetten-Grupp*“ (SysCategory) als „*Eltern*“ und Schlussendlich die zusätzlichen Facetten (SysCategory) als „*Kind-Elemente*“ angelegt und referenziert werden. Ist dies korrekt angelegt, sieht dies dem Fall-Beispiel ähnlich:

Backend

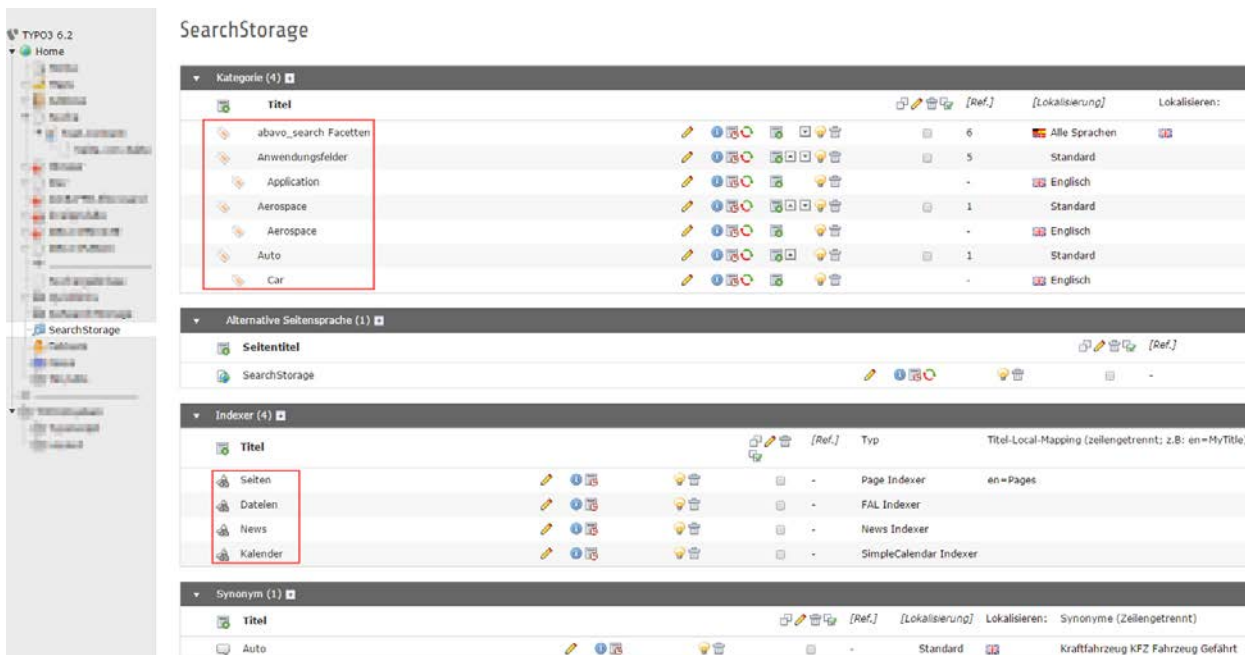


Abb.: Beispiel Suche-Datensatzsammlung im Überblick

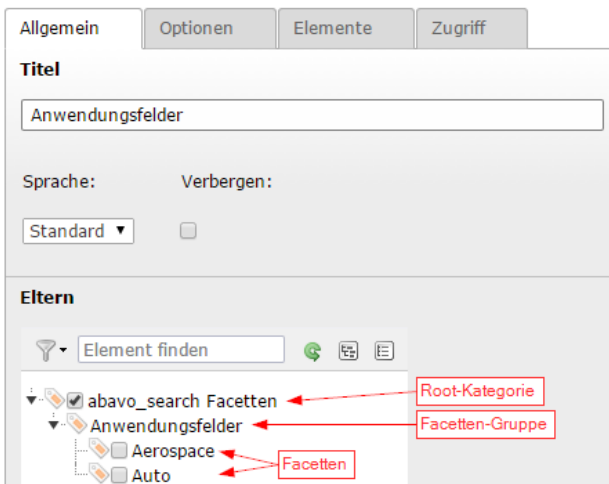


Abb.: Beispiel System-Kategorien-Baumstruktur

Titel

Frontend-Titel

Hinweis: Bei einer Lokalisierung wird nur das Titel-Feld übersetzt.

Eltern

- Als Facetten-Gruppe mit Root-Kategorie referenziert
- Als Facette mit Facetten-Gruppe referenziert

TIPP: Um die Verwaltung der Facetten (Kategorien) zu erleichtern und die Übersicht zu behalten, ist es empfohlen, die Baumansicht der SysCategories auf die UID der „*Root-Kategorie*“ im Storage-Folder-Seiten-TS-Config einzuschränken:

```
TCEFORM.sys_category.parent.config.treeConfig.rootUid = <UID>
```


Frontend

▼ **Filtern und Sortieren**

Inhaltstyp

☒ Seiten ☒ Dateien ☒ News ☒ Kalender → **Indexer-Konfigurationen**

Anwendungsfelder

☐ Aerospace ☐ Auto → **Facetten-/Gruppe**

☒ Alle aus-/abwählen

10 Ergebnisse pro Seite

Sortieren nach Relevanz absteigend

Abb.: Das erweiterte Suche-Formular mit Beispiel-Facetten

Durch Zuordnung der selbst definierten Facetten in den verschiedensten Inhaltstypen und Berücksichtigung in den jeweiligen Indexer-Klassen, ist so eine zusätzliche Facettierung möglich.

Standardmäßig unterstützen folgende Indexer-Klassen (Indexer-Typ) die Kategorie-Zuordnung bei der Indexierung:

- Seiten
- TX_NEWS

Eine Erweiterung der Facettierung für Dateien (FAL) wäre grundsätzlich vorstellbar.

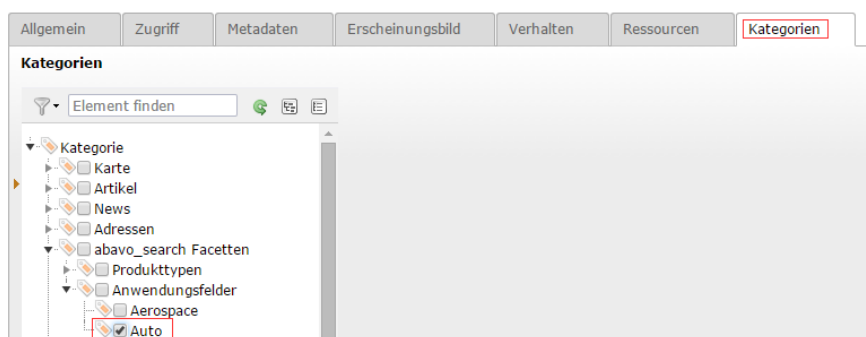


Abb.: Beispiel der Zuordnung der Facette „Auto“ in einem Seiten-Datensatz.

Die Frontend-Plug-In's

Die Extension stellt zwei Plug-In's bereit. Jeweils eines für das *Suchformular* und *Suchergebnisse*. Das Formular kann selbst kann auch unabhängig vom Plugin über FLUID mit Zielzeit des Such-Ergebniss-Plugins verwendet werden.

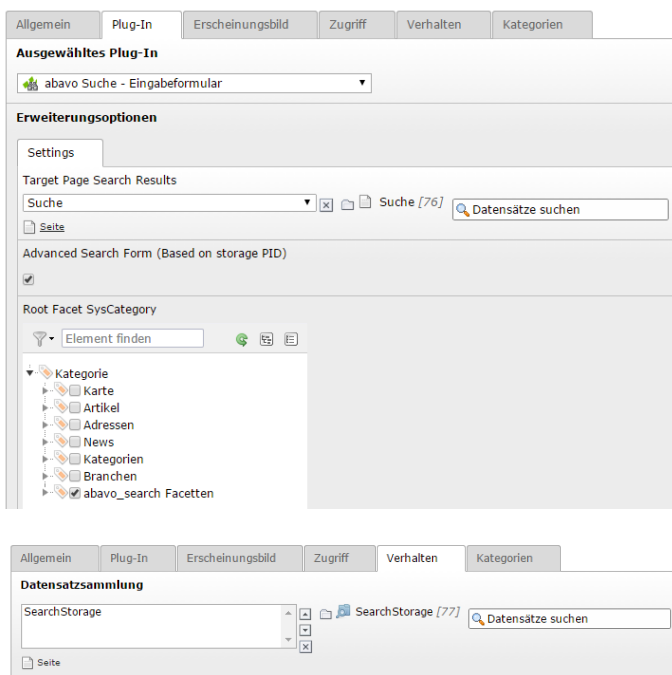
Such-Formular ohne Plugin mit FLUID

```
<f:form extensionName="AbavoSearch" controller="Search" action="search" pageUid="{settings.searchresultpage}" class="search-form">
  <f:form.textfield name="tx_abavosearch_piresults[search]" class="search-word autocomplete"
    placeholder="{f:translate(extensionName: 'AbavoSearch', key: 'form.autocomplete')}"
    additionalAttributes="{data-pages:settings.searchstoragepage, autocomplete: 'off'}" />
  <f:form.submit value="{f:format.htmlentitiesDecode(value: '&#xf002;')}" />
</f:form>
```

Hinweis: Dieses Snippet zeigt ausschließlich das einfache Such-Formular.

Eingabeformular

Über den Content-Wizard des TYPO3-BE kann das Inhaltselement Plugin „abavo Suche – Eingabeformular“ eingefügt werden.



Target Page Search Result

Ziel-Seite mit Such-Ergebniss-Plugin

Advanced Search Form

Aktiviert die erweiterte Suche mit Filter und Facettierung;
Basiert auf Datensatzsammlung des Plugins (!);

Root Facet SysCategory

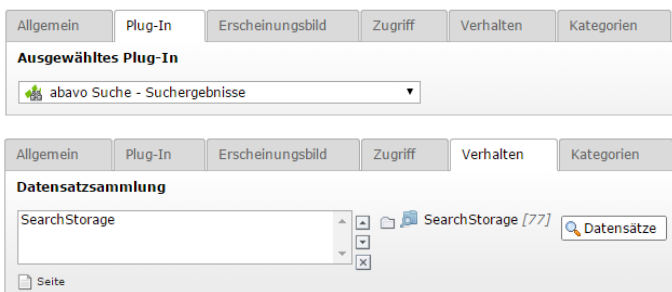
Ist eine erweiterte Suche und Facettierung gewünscht,
muss eine Root-Kategorie gewählt werden.

Datensatzsammlung

Speicherort(e) aller Suche betreffenden Datensätze;

Suchergebnisse

Das Plugin für die Suchergebnisse wird wie zuvor das Plugin für das Eingabeformular über den Content-Wizard eingefügt.



Auswahl des Suchergebnis-Plugins

Datensatzsammlung

Speicherort(e) aller Suche betreffenden Datensätze;

Einrichten des Scheduler-Tasks

Damit die Suche Ergebnisse liefern kann, erfordert dies einen aufgebauten Such-Index. Dies ist mit Hilfe des TYPO3-Schedulers möglich. Die *abavo_search* stellt einen Task bereit, der StoragePID-bezogen Indexiert.

Die wesentlichen Hauptaufgaben des Tasks sind wie folgt gegliedert:

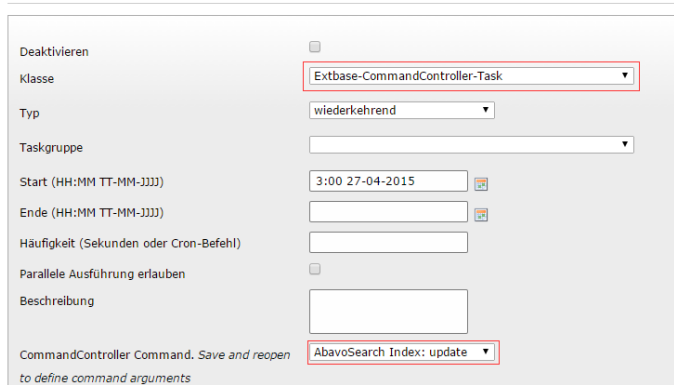
- Aufbau des Such-Index
- Bereinigung ungenutzter Index-Daten
- Bereinigung ungenutzter Autovervollständigung

Bei den Bereinigungsjobs wird überprüft, ob der referenzierte Datensatz noch im System sichtbar existent ist, ansonsten wird die Referenz entfernt. Dadurch erscheinen nur Suchergebnisse und Autovervollständigungen, die zu einem Ergebnis führen.

Konfiguration

Da der Task die „Extbase-CommandController-Task“ Technologie nutzt wird dieser entsprechend im Backend unter Scheduler eingerichtet. Die nicht markierten Felder folgend dem TYPO3-Standard für Task-Felder.

Task hinzufügen



Klasse

Extbase-CommandController-Task

CommandController Command

AbavoSearch Index: update

Nach erstmaligem Speichern stehen die Task-Abhängigen Einstellungen bereit.



Argument: storagePid, comma separated

Kommaseparierte Liste von StoragePID's, wo sich Indexer-Konfigurationen befinden.

TIPP: Durch die Aufteilung der Suchindex-Konfigurationen auf verschiedene StoragePID's können die verschiedensten Indexierungs-Strategien umgesetzt werden, da die StoragePID des Indexes von der jeweiligen Indexer-Konfiguration abhängt.

Praxisbeispiel: Der FAL-Index benötigt intensiv Zeit zum Aufbau und soll im Gegensatz zum Seiten-Index nur einmal täglich aktualisiert werden.

Extension-Hooks

Über sogenannten „Extension-Hooks“ können die verschiedensten Verhaltensweisen von TYPO3-Extensions beeinflusst werden. Diese Möglichkeit der Modifikation stellt auch die *abavo_search* flexibel bereit.

Index-Modification-Hook

Soll der Index bei der Erstellung modifiziert werden, so können beliebig viele „Index-Modification-Hooks“ zu einem Index-Typen registriert werden.

Die Extension selbst nutzt dieses Prinzip bereits in der Basis für den TX_News-Indexer und kann als Orientierung für die Verwendung in zum Beispiel einer eigenen Extension (*ext_localconf.php*) oder Systemweit (*LocalConfiguration.php*) ergänzt werden.

Registrieren

In eigener *ext_localconf.php* oder *typo3conf/LocalConfiguration.php*:

```
$GLOBALS['TYPO3_CONF_VARS']['EXTCONF']['abavo_search']['modifyIndex']['Example'][] =
'Abavo\AbavoSearch\Hooks\ExampleIndexModifier';
```

Wichtig ist dabei die korrekte Schreibweise des Typs (grün) und des Namespaces (blau).

In der *ext_localconf.php* der *abavo_search* stehen dazu Orientierungsbeispiele.

Modify-Klasse

Als Basis sollte immer die Beispielklasse verwendet werden:

EXT:abavo_search/Classes/Hooks/ExampleIndexModifier.php

Dies PHP-Klasse kann dann dupliziert, umbenannt und entsprechend den Anforderungen modifiziert werden. Es wird in der Modify-Klasse automatisch über den Index-Modification-Hook die Methode „*modifyIndex*“ verwendet. Daher muss diese Methode vorhanden sein.

Code-Beispiel:

Dem Index-Titel ein Prefix „MyPrefix: “ voranstellen

```
...
class ExampleIndexModifier
{
    function modifyIndex(&$recordObj, &$title, &$abstract, &$content)
    {
        try {
            $title = 'MyPrefix: '.$title;

        } catch (\Exception $e) {
            \TYPO3\CMS\Extbase\Utility\DebuggerUtility::var_dump($e->getMessage(), 'ExampleIndexModifier');
        }
    }
}
...
```

Methoden-Beschreibung: modifyIndex

@param object \$recordObj Das jeweilige Record Objekt (Zugriff auf Getter und andere Objekt-Eigenschaft/-Methoden)

@param string \$title Der Index-Titel in den Suchergebnissen

@param string \$abstract Der Inhalt des Feldes „abstract“

@param string \$content Der vollständige Index-Content

Indexer-Configuration-Hook – Der eigene Indexer

Ist es gewünscht einen eigenen Suchindex auf zu bauen und dafür eine eigene Konfiguration bereit zu stellen, kann dafür ein eigener Indexer mit einer völlig flexiblen Konfiguration erstellt werden.

Registrieren

In eigener `ext_localconf.php` oder `typo3conf/LocalConfiguration.php`:

```
$GLOBALS['TYPO3_CONF_VARS']['EXTCONF']['abavo_search']['indexers'][] = [
    'Mytype' => [
        'Label' => 'Example Indexer',
        'FlexForm' => 'FILE:EXT:abavo_search/Configuration/FlexForms/ExampleIndexerConfig.xml',
        'targetPid' => false
    ]
];
```

Mytype	Unique Index Type Key	Array
Label	Titel in Typen-Auswahl in Indexer-Konfiguration	String
FlexForm	FlexForm Konfiguration; Hier ist die gängige TypoScript schreibweise möglich; Basis-Konfig: <code>FILE:EXT:abavo_search/Configuration/FlexForms/ExampleIndexerConfig.xml</code>	T3-FlexForm konformes XML
targetPid	Feld TargetPid anzeigen	Boolean

Indexer-Klasse

Als Basis ist es empfohlen die Example-Klasse zu verwenden:
 EXT:abavo_search/Classes/Indexers/ExampleIndexer.php

Als ersten Schritt sollte diese dupliziert und Dateiname sowie Klassenname sprechend umbenannt werden. Anschließend kann die weitere Erstellung, den Anforderungen entsprechend fortgesetzt werden.

Code-Beispiel

```
function __construct($settings)
{
    $this->settings = $settings;
    $objectManager = \TYPO3\CMS\Core\Utility\GeneralUtility::makeInstance('\TYPO3\CMS\Extbase\Object\ObjectManager');
    $this->exampleRepository = $objectManager->get('\Vendor\Domain\Repository\ExampleRepository');
}

public function getData(\Abavo\AbavoSearch\Domain\Model\Indexer $indexer)
{
    ...

    $resources = $this->exampleRepository->findByFooBar();

    foreach ($resources['result'] as $result) {

        /*
         * PLACE CODE HERE
         */

        // Index-Modify-Hook

        $title = $result['title'];
        $content = $result['content'];
        $abstract = $result['abstract'];
        ...

        $this->modifyIndexHook('Example', $result, $title, $content, $abstract);
        ...

    }

    return $this->data;
}
```

Das Basis-Beispiel liefert bereits die wesentlichen Bestandteile mit, sodass eine schnell Erstellung eigener Indexer Möglich ist. Über `$this->settings` kann auf die settings der Indexer-Konfiguration und der FlexForm zugegriffen werden. Bestehende Indexer können Orientierungshilfen sein.

Stop-Wörter

Ein weiteres sehr praktisches Feature ist die Verwendung von „*White- und Blacklists*“ bei Such-Anfragen. Dabei filtert die Extension Suchanfragen, die in der Regel zu keinem sprechenden Suchergebnissen führen (<http://de.wikipedia.org/wiki/Stopwort>).

Die Whitelist steht über der Blacklist und dessen enthaltene Begriffe sind daher grundsätzlich von der Filterung der Blacklist ausgeschlossen.

Um dieses Feature flexibel erweiterbar zu halten, besteht die Möglichkeit, eigene Listen zu ergänzen. Über die TS-Setup-Konfiguration wird dazu der lokale Pfad im „*fileadmin*“, wo sich die Listen befinden sollen einfach erweitert.

Beispiel

TS-Setup

```
plugin.tx_abavosearch.stopWordList{
    1 = fileadmin/xml/abavo_search/CustomBlackLists/Set1/
    2 = fileadmin/xml/abavo_search/CustomBlackLists/Set2/
}
```

Code-Auszug

EXT:abavo_search/Resources/Private/WordLists/en.StopWords.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<words version="1.0.0">
    <word title="about" lang="en" />
</words>
```

Dieses Beispiel enthält eine Blacklist für das local „en“. Jeder Set-Pfad kann mehrere Listen mit mehreren Sprachen mit entsprechendem local-Prefix enthalten.

TypoScript-Referenz

TS-Setup

plugin.tx_abavosearch .view

Property:	Data type:	Description:	Default:
templateRootPaths.1	string	Custom template root path	
partialRootPaths.1	string	Custom partial root path	
layoutRootPaths.1	string	Custom layout root path	

plugin.tx_abavosearch .settings

Property:	Data type:	Description:	Default:
stopWordList.<1,2...>	string	Custom stop word lists; One list a numeric key >0 (0=BaseList)	
whiteWordList.<1,2...>	string	Custom white word lists; One list a numeric key >0 (0=BaseList)	
maxSearchStrinLen	int+	Max string length of search term	40

TS-Constants

plugin.tx_abavosearch .settings

Property:	Data type:	Description:	Default:
cssFile	string	Path to css file	typo3conf/ext/abavo_search/Resources/Public/Css/Style.min.css
cssIconfont	string	IconFont CSS:Path to icon font css file	typo3conf/ext/abavo_search/Resources/Public/Icons/Vendor/fontello/css/fontello.min.css
cssIconfontanim	string	IconFont Animation CSS:Path to icon font animation css file	typo3conf/ext/abavo_search/Resources/Public/Icons/Vendor/fontello/css/animation.min.css
jqueryUi	string	Path to jQuery-UI Javascript	https://code.jquery.com/ui/1.11.3/jquery-ui.min.js
stopWordList	string	Path to Stop-Word-Lists	EXT:abavo_search/Resources/Private/WordLists/
whiteWordList	string	Path to White-Word-List	EXT:abavo_search/Resources/Private/WordLists/
itemsPerPage	integer	Results each page: Fallback if form value is invalid	10
minSearchStrinLen	int+	Min search string len for search	3
targetPid	int+	Page with search results plugin	
jsGlobalInclude	bool	Include Form.js with page.includeJS: Global TS-Include instead of form include	0