comundus GmbH

Heerstr. 111
DE- 71332 Waiblingen

Geschäftsführer
Klaus Hillemeier

Tel: +49 (0)7151 96528-0
Fax: +49 (0)7151 96528-999

http://www.comundus.com

Alkacon Software GmbH

An der Wachsfabrik 13
DE - 50996 Köln

Geschäftsführer
Alexander Kandzior

Amtsgericht Köln
HRB 54613

Tel: +49 (0)2236 3826 - 0
Fax: +49 (0)2236 3826 - 20

http://www.alkacon.com

OpenCms 8.5.1 Handbuch Deutsch

August 2013

Handbuch-Version 1.2

Contents

1 Einleitung 7
2 Seiten-Editor 8
2.1 Toolbar – Symbolleiste 8
2.2 Editierpunkt 8
2.3 Inhalt hinzufügen 9
2.3.1 Inhalt neu erstellen 9
2.3.2 Bestehenden Inhalt suchen 10
2.4 Zwischenablage 11
2.5 Kontextmenü 11
2.5.1 Eigenschaften 12
2.5.2 Attribute 12
2.5.3 Gültigkeit 12
2.5.4 Gesperrt-Bericht 12
2.5.5 Kategorien zuweisen 12
2.5.6 SEO Optionen 13
2.5.7 Rückgängig 13
2.5.8 Zeige Workplace 13
2.6 Veröffentlichen 13
3 Sitemap-Editor 13
3.1 Editor öffnen 14
3.2 Sitemap-Editor Symbolleiste 14
3.3 Veröffentlichen 15
3.4 Seite erstellen 15
3.4.1 Containerseiten 15
3.4.2 Bearbeiten von Modellseiten 15
3.4.3 Typenseiten 16
3.4.4 Funktionsseiten 16
3.4.4.1 HTML Redirect 16
3.4.4.2 Navigationsebene 16
3.4.4.3 Funktionsdetailseiten 16
3.5 Zwischenablage 16
3.6 Zeige alle Dateien an 17
3.7 Kontextmenü 17
3.8 Editierpunkt 17
3.9 Seitentypen 17
3.9.1 Modellseiten 17
3.9.1.1 Erstellung einer Modellseite 18
3.9.1.2 Konfiguration 18

3.9.2 Detailseiten 18
3.9.2.1 Anwendungsfall 18
3.9.2.2 Konfiguration 19
3.9.2.3 Verwendung 19
3.9.3 Funktions-Detailseite 19
3.9.3.1 Anwendungsfall 19
3.9.3.2 Allgemeine Schritte 20
3.9.3.3 Erstellung dynamischer Funktionen 20
3.9.3.4 Erstellen der Funktion-Detailseiten 20
3.9.3.5 Verwendung 21
3.9.4 Navigationsebene 21
3.9.4.1 Anwendungsfall 21
3.9.4.2 Verwendung der Navigationsebene 21
3.9.4.3 Details zur Implementierung 21
3.9.5 Versteckte Einträge 22
3.9.5.1 Beschreibung 22
3.9.5.2 Verwendung 22
4 Inhaltseditor 23
4.1 Widgets 24
4.1.1 String-Widget 25
4.1.2 Boolean-Widget 25
4.1.3 Display-Widget 25
4.1.4 Select-Widget 26
4.1.5 Textarea-Widget 26
4.1.6 Radio-Button-Widget 26
4.1.7 Multiselect-Widget 27
4.1.8 Combo-Widget 27
4.1.9 Ressourcentyp-Combo-Widget 27
4.1.10 HTML-Widget 28
4.1.11 Lokalisierungs-Widget 29
4.1.12 Color-Picker-Widget 29
4.1.13 Date-Picker-Widget 30
4.1.14 Kategorie-Widget 31
4.1.15 Gruppen-Widget 31
4.1.16 Multi-Gruppen-Widget 32
4.1.17 VFS-Datei-Widget 32
4.1.18 VFS-Bilder-Widget 33
4.1.19 Bildergalerie-Widget 34
4.1.20 Download-Galerie-Widget 34
4.1.21 HTML-Galerie-Widget 35
4.1.22 Tabellen-Galerie-Widget 35
4.1.23 Link-Galerie-Widget 36
4.2 Select-Widget-Konfiguration 36
4.3 Implementierung von Custom-Widgets 37
5 Inline-Editor 38
5.1 Felder, die das Inline-Editing unterstützen 39
5.2 Konfiguration 39

5.2.1 Formatters 39
5.2.2 Verschachtelte Inhalte 40
5.2.3 Details 40
6 Elementgruppe 40
6.1 Beschreibung 40
6.2 Kombination mit Vorlagenseite 40
6.3 Gebrauch der Elementgruppe 41
7 Vererbungsgruppe 43
7.1 Beschreibung 43
7.2 Basisinformationen 43
7.3 Verwendung 43
7.3.1 Erstellung von Vererbungsgruppen 43
7.3.2 Verwendung vorhandener Vererbungsgruppen 43
7.3.3 Änderung von Vererbungsgruppen 43
7.3.3.1 Titel und Beschreibung ändern 44
7.3.3.2 Inhaltselemente hinzufügen 44
7.3.3.3 Inhaltselemente entfernen 45
7.3.3.4 Umsortierung von Inhaltselementen 45
7.3.3.5 Vererbungsstatus 45
7.3.3.6 Zuvor versteckte Elemente anzeigen 46
7.3.3.7 Speichern 47
7.3.3.8 Auflösen 47
7.3.3.9 Elementeinstellungen bearbeiten 47
7.4 Internas 47
8 Kollektoren 47
8.1 Implementierung 47
8.2 Konfiguration 48
8.3 Anwenden von Kollektoren in JSP-Dateien 49
8.4 Erstellen von Listen mit Drag & Drop Funktion 49
8.5 Verwendung von Kollektoren in Detailseiten 50
8.6 Code-Beispiel 50
9 XSD Choice-Element 50
9.1 Definition 50
9.2 Einzel- und Mehrfachauswahl 51
9.3 Inhaltseditor 52
9.4 Zugriff auf Werte in einer JSP 52
9.5 Beispiel 53
10 ADE-Konfiguration 54
10.1 Sitemap-Konfiguration 54
10.2 Modul-Konfiguration 54
10.3 Vererbung der Konfiguration 55
11 Integration der Solr Suche 55
11.1 Allgemeines 55
11.2 Suche nach Inhalten in OpenCms 55
11.2.1 Demo 55
11.2.2 Schnellstart Beispiel 55
11.2.2.1 Senden einer REST-ähnlichen Abfrage 55

11.2.2.2 Abfragen der Antwort ... 56
11.2.2.3 Senden einer Java-API-Abfrage ... 58
11.2.2.4 Solr Querys mit der CmsSolrQuery-Klasse ... 58
11.2.3 Erweiterte Suchfunktionen ... 58
11.2.3.1 Abfragen von mehreren Cores (Indizes) ... 58
11.2.3.2 Den Standard-OpenCms-Solr-Kollektor verwenden ... 59
11.3 Indizierung von OpenCms-Inhalten ... 59
11.3.1 Konfiguration der Suche ... 59
11.3.1.1 Embedded/HTTP Server Solr ... 59
11.3.1.2 Suchindex (Indizes) ... 60
11.3.1.3 Indexquellen ... 61
11.3.1.4 Neuer Dokumenttyp ... 61
11.3.1.5 Die standardmäßige Solr Feldkonfiguration ... 61
11.3.1.6 Migration eines Lucene-Index in einen Solr-Index ... 62
11.3.2 Indizierte Daten ... 63
11.3.2.1 Das Solr-Index-Schema (schema.xml) ... 63
11.3.2.2 Standard-Index Felder ... 63
11.3.2.3 Benutzerdefinierte Felder für XML-Inhalte ... 64
11.3.2.4 Dynamische Feldzuordnungen ... 65
11.3.2.5 Benutzerdefinierte Feldkonfiguration ... 65
11.3.2.6 Erweitern der CmsSolrFieldConfiguration ... 65
11.4 Hinter der Solr-Kulisse ... 66
11.4.1 Der Request-Handler ... 66
11.4.2 Berechtigungsprüfung ... 66
11.4.3 Konfigurierbarer Post-Prozessor ... 66
11.4.4 Mehrsprachige Unterstützung ... 67
11.4.5 Auflösung mehrsprachiger Abhängigkeiten ... 67
11.4.6 Extrahierter Ergebnis-Cache ... 67
11.5 Häufig gestellte Fragen ... 67
11.5.1 Wie ist Solr allgemein integriert? ... 67
11.5.2 Wie sortiert man Text für bestimmte Sprachen? ... 68
11.5.3 Wie wird das Highlighting im Suchergebnis angewendet? ... 69
11.5.3.1 Unterstützt OpenCms das Highlighting im Suchergebnis? ... 69
11.5.3.2 Unterstützt das Java API von OpenCms Highlighting? ... 69
11.5.3.3 Ist Highlighting ein Performance-Killer? ... 70
11.5.4 Solr Indizierungs-Fragen ... 70
11.5.4.1 Bitte erklären Sie den Unterschied zwischen “Solr Online und Offline” ... 70
11.5.4.2 Wird bei einer Solr-Abfrage nur der Solr Index gebraucht? ... 70
11.5.5 Wo finde ich allgemeine Information über Solr? ... 70
11.5.5.1 Gibt es eine Möglichkeit, eine vollständige Sicherung des gesamten Index zu erstellen? ... 70
11.5.5.2 Wie kann man einen ausfallsicheren Index wiederherstellen? ... 70
11.5.5.3 Die Solr Ergebnisliste wird standardmäßig auf 50 begrenzt, wie kann man mehr als 50 Ergebnisse bekommen? ... 70
11.5.6 Solr Mailinglist Fragen ... 70
11.5.6.1 Eine ClassCastException wird geworfen, was kann ich tun? ... 70
11.5.6.2 Ist es möglich die Elemente mit maxoccurs > 1 abzubilden? ... 71

11.5.6.3 Wie werden OpenCmsDateTime-Elemente indiziert? ... 71
12 SEO ... 71
12.1 Einführung ... 71
12.2 Aliase ... 71
12.2.1 Einfache Aliasnamen ... 71
12.2.2 Rewrite-Aliase ... 71
12.2.3 Interna ... 72
12.2.4 SEO Optionsdialog ... 72
12.2.5 Aliase Bearbeiten-Dialog ... 73
12.2.6 Neue Aliase erstellen ... 74
12.2.7 Bearbeiten von vorhandenen Aliasen ... 75
12.2.8 Exportieren und Importieren von Aliasen ... 75
12.3 Automatische robots.txt- und XML-Sitemap-Generierung ... 77
12.3.1 XML-Sitemap-Generierung ... 77
12.3.2 Generierung einer robots.txt-Datei ... 78
13 CMIS ... 78
13.1 OpenCms und CMIS ... 78
13.2 Zugriff auf OpenCms via CMIS ... 79
13.3 CMIS Integration ... 80
13.3.1 Begrenzung der CMIS Implementierung ... 82
13.3.2 CMIS-Client-Programm-Beispiel ... 83
14 JSP Grundlagen ... 84
14.1 JSP Funktionen ... 85
14.2 Die JSP <cms>-Taglib ... 85
14.2.1 Wie wird die "taglib"-Direktive eingefügt? ... 85
14.2.2 Verfügbare Tags ... 85
14.2.3 Verfügbare EL-Funktionen ... 87
15 Impressum ... 88

1 Einleitung

Dieses Dokument ist geschrieben von Alkacon Software – Die OpenCms Experten.

Diese Dokumentation wurde von der comundus GmbH – einem der ersten OpenCms Solution Provider in Deutschland – ins Deutsche übersetzt und zukünftig mit fachlichen Erläuterungen und Grafiken ergänzt.

Das Ziel ist es, Entwicklern von der Einarbeitung in OpenCms bis zum vertraut werden mit tiefgreifenderen Entwicklungen und neuen Themen rund um OpenCms zu helfen. Die Dokumentation nutzt Links, um auf ein lokal installiertes OpenCms (Version >=8.5.1) einschließlich der OpenCms v8-Module (module-v8) zusammen mit den OpenCms Entwickler-Modulen (dev-demo) zu verweisen. Um die interaktive Entwicklung von OpenCms 8 Demo-zusammen mit dieser Dokumentation zu nutzen, nehmen wir an, dass Sie OpenCms installiert haben und es mit dem OpenCms Hauptservlet unter erreichbar ist: http://localhost:8080/opencms/opencms.

Wenn OpenCms bei Ihnen lokal läuft, sind Sie in der Lage, die Links in dieser Dokumentation zu verwenden. Die Links öffnen die entsprechenden DEMO Seiten innerhalb der "Wunderbaren Welt der Blumen".

Wenn Sie kein laufendes OpenCms mit den Alkacon Demo-Modulen haben, können Sie das neueste veröffentlichte OpenCms hier herunterladen.
Lesen Sie die OpenCms Server Installationsanleitung, die Sie in der heruntergeladenen ZIP-Datei (installation.html) finden und Sie werden sehen wie einfach es ist, OpenCms einzurichten.

2 Seiten-Editor

Der Seiten-Editor ist die Hauptansicht für Redakteure, welche einem das Hinzufügen, Verschieben und Bearbeiten des Seiteninhalts während der Seitenvorschau erlaubt.

Außerdem erlaubt er den Zugriff auf alle notwendigen Redakteursaufgaben, wie der Änderung der Sitemap, dem Veröffentlichen von Inhalten und die Verwaltung von Dateiattributs- oder Dateieigenschaften.

2.1 Toolbar – Symbolleiste

Die Symbolleiste wird durch Anklicken des farbigen Editor-Symbols , welcher sich rechts oben befindet, ein- bzw. ausgeblendet. Dieses ist nur in der Offline-Vorschau der Webseite sichtbar. Die Symbolleiste ist am oberen Rand des Fensters angedockt und zeigt die verfügbaren Optionen zur Bearbeitung der angezeigten Seite.

Verfügbare Optionen

Veröffentlichen: Dieses Icon öffnet den Veröffentlichen-Dialog.

Zwischenablage: Dieses Icon öffnet die Zwischenablage. Darin sind die Favoriten und die letzten Änderungen enthalten.

Inhalt hinzufügen: Über dieses Icon kann man per Drag & Drop einer Container-Page Inhalte hinzufügen.

Editierpunkt: Klickt man auf den „Editierpunkt“, werden editierbare Elemente mit grauen Editierpunkten gekennzeichnet.

Kontext: Zeigt das Kontextmenü der aktuellen Seite.

Sitemap: Wechselt zum Sitemap-Editor.

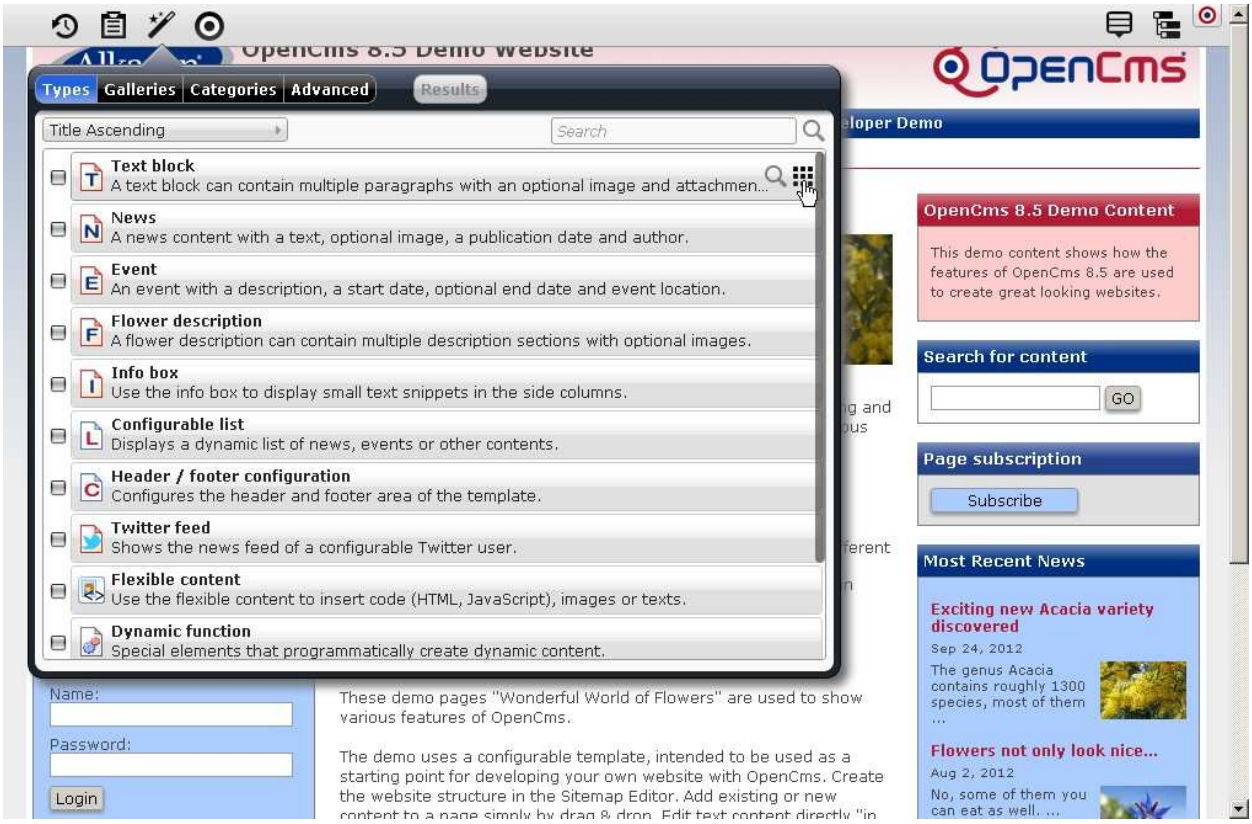
2.2 Editierpunkt

Beim Klicken auf den **Editierpunkt** in der Symbolleiste kann der Bearbeiten-Modus für alle Elemente auf der Seite ein- und ausgeschaltet werden. Der Editierpunkt erscheint auf allen Elementen, die bearbeitet werden können.

2.3 Inhalt hinzufügen

2.3.1 Inhalt neu erstellen

Durch Anklicken von **Inhalt hinzufügen** öffnet sich das Dialogfenster der Galerie und zeigt eine Liste mit allen für diese Containerseite verfügbaren Inhaltstypen.



Dieses Dialogfenster ermöglicht dem Redakteur eine neue Seite zu erstellen bzw. eine bestehende auszuwählen.

Beim Erstellen einer neuen Seite fügt er einfach den gewünschten Inhaltstyp seiner Wahl per Drag & Drop in die Seite ein. Dabei wird intern automatisch eine neue Instanz des Inhaltstyps erstellt.

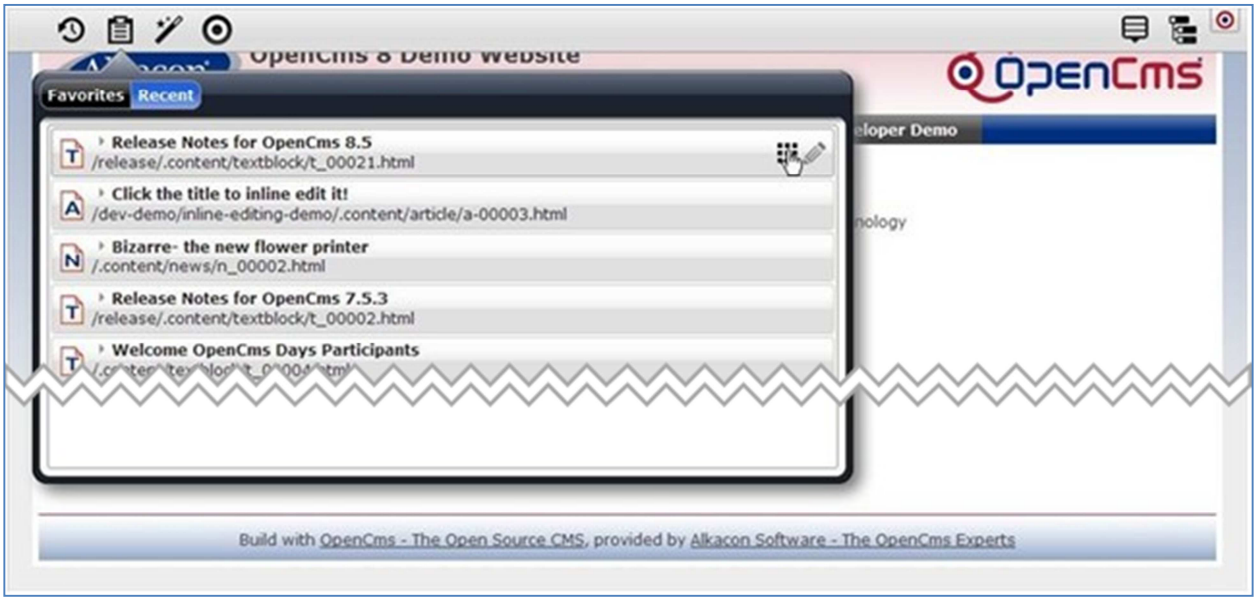
Beim Hinzufügen werden die Bereiche hervorgehoben, auf denen der Inhalt platziert werden kann:

2.3.2 Bestehenden Inhalt suchen

Beim Klicken des Symbols **Inhalte hinzufügen** und der Auswahl eines oder mehrerer Inhaltstypen können vorhandene Inhalte wiederverwendet werden. Das Suchergebnis kann durch die Auswahl einzelner oder mehrerer Suchkriterien in den Tabs (Typen, Galerien, Kategorien und Erweitert) verfeinert werden. Ein Klick auf den Suchergebnisse-Tab aktualisiert die Suchergebnisliste. Diese enthält alle Inhalte, die den angegebenen Suchkriterien entsprechen. Die Suchkriterien sind ODER-verknüpft.

2.4 Zwischenablage

Beim Anklicken des Symbols **Zwischenablage** in der Symbolleiste werden die persönlichen Favoriten und eine Liste der Elemente angezeigt, die in letzter Zeit am häufigsten verwendet wurden. Per Drag & Drop kann Inhalt aus der Zwischenablage zur Seite hinzugefügt werden.



Es können Inhaltselemente zu den eigenen Favoriten hinzugefügt werden, indem man das Symbol **zu Favoriten hinzufügen** anklickt.

2.5 Kontextmenü

Im Kontextmenü stehen nachfolgende Möglichkeiten zur Auswahl:

Eigenschaften: OpenCms-Eigenschaften der aktuellen Containerseite werden angezeigt und können verändert werden.

Attribute: Es werden die Attribute des Containers angezeigt, wie z.B. Erstellungs-, Änderungsdatum usw.

Verfügbarkeit: Öffnet den Verfügbarkeitsdialog

Gesperrt-Bericht: Zeigt die gesperrten Ressourcen der aktuellen Containerseite an.

Kategorien zuweisen: Dialog, um einer Containerseite Kategorien zu zuweisen.

SEO Optionen: Dialog für Einstellungen zur Suchmaschinenoptimierung.

Rückgängig: Die letzte veröffentlichte Version der Containerseite wird wieder hergestellt.

Zeige Workplace: Öffnet den OpenCms Workplace.

Logout: Abmelden vom OpenCms Workplace.

2.5.1 Eigenschaften

Beim Auswählen von **Eigenschaften** im Kontextmenü, können die Seiten-Eigenschaften der Containerseite geändert werden. Der **neu gestaltete Eigenschaften-Dialog** erlaubt das Ändern aller Eigenschaften. Es gibt drei Teilmengen der Eigenschaften: **Basis**, **Individuell** und **Geteilte Eigenschaften**. Die Basis-Eigenschaften sind eine konfigurierbare Teilmenge aller verfügbaren Eigenschaften.

2.5.2 Attribute

Der Dialog **Attribute** zeigt neben den allgemeinen Informationen der Containerseite erweiterte Metadaten wie den Seitentitel, den Ressourcentyp, die Dateigröße, den Status der Datei (neu, geändert, gelöscht usw.) und die Information wann die Seite erstellt wurde und von wem. Außerdem zeigt es das letzte Änderungsdatum und das Projekt zu dem es gehört. Die Sprache und die Berechtigung des aktuellen Benutzers werden hier auch angezeigt.

2.5.3 Gültigkeit

Bei Auswahl der Option **Gültigkeit** im Kontextmenü kann die Verfügbarkeit des entsprechenden Containers eingestellt werden. Folgende Möglichkeiten stehen zur Auswahl:

- **Später Veröffentlichen →Veröffentlichungsdatum:** Durch die Auswahl von **Später Veröffentlichen** im Kontextmenü, wird die Seite zum angegebenen Datum veröffentlicht.
- **Gültig von:** Bei Auswahl von **Gültig von** wird die Seite ab dem eingegebenen Datum online sichtbar sein (wenn sie veröffentlicht wurde).
- **Gültig bis:** Bei Auswahl von **Gültig bis** wird die Seite bis zu dem eingegebenen Datum online sichtbar sein (wenn sie veröffentlicht wurde).

Ein Datum kann entweder über Eingabe von Zahlen in die entsprechenden Felder erfolgen oder über die Auswahl im Kalender.

2.5.4 Gesperrt-Bericht

Der **Gesperrt-Bericht** gibt Auskunft über den Bearbeitungsstatus der ausgewählten Datei. Ist die Datei von einem anderen Benutzer gesperrt, kann man sich die Benutzerinformationen sowie die Projektinformationen anzeigen lassen. Dieser Dialog zeigt auch vererbte Sperrungen an.

2.5.5 Kategorien zuweisen

Diese Auswahl ermöglicht das einfache Zuordnen von Kategorien zur aktuellen Seite.

2.5.6 SEO Optionen

Bei der Auswahl von SEO-Optionen wird der Content Manager geöffnet. Dieser zeigt Einstellungen zu relevanten SEO Eigenschaften der Containerseite an, d.h. Titel, Beschreibung und Keywords. Zusätzlich ist es möglich einen Alias für die Seite zu definieren. Es können neue Regeln erstellt oder zwischen den bestehenden ausgewählt werden. Bei den Rewrite-Rules kann man zwischen einem temporären Redirect (302 returned), einem permanenten Redirect (301) oder dem einfachen Anzeigen der Seite (200) auswählen. Bereits gesetzte Aliasnamen werden unten in der Maske angezeigt.

2.5.7 Rückgängig

Über die Auswahl Rückgängig wird der letzte veröffentlichte Status der aktuellen Containerseite wiederhergestellt.

2.5.8 Zeige Workplace

Wählen Sie die Option Workplace anzeigen im Kontextmenü, um den OpenCms Workplace in einem neuen Fenster zu öffnen. Im OpenCms-Workplace stehen erweiterte Optionen zum Verwalten von Inhalten und Ressourcen sowie Administrationsoptionen zur Verfügung. Der Workplace sollte nur für fortgeschrittene Benutzer, wie Administratoren, Entwickler oder wirklich erfahrenen Redakteuren zugänglich sein.

2.6 Veröffentlichen

Nachdem Sie Ihre Arbeiten beendet haben und Ihre Änderungen für alle online sichtbar machen wollen, müssen diese veröffentlicht werden. Eine Veröffentlichung macht den geänderten Content in der online Version der Website sichtbar. Um die geänderten Seiten zu veröffentlichen, einfach die Auswahl **Veröffentlichen** wählen. Nun werden alle unveröffentlichten Objekte des Benutzers aufgelistet, die hinzugefügt, verändert oder gelöscht wurden. Nicht veränderte Seiten werden nicht in der Liste angezeigt. Für eine bessere Übersicht ist die Veröffentlichungs-Liste in verschiedene Bearbeitungsabschnitte aufgeteilt. Es ist möglich einzelne Seiten auszuwählen oder alle angezeigten Seiten aus- oder abzuwählen oder aber alle Seiten eines Bearbeitungsabschnitts aus- oder abzuwählen. Es kann eine einzelne Ressource aus der Benutzer Veröffentlichungsliste entfernt werden. Wurde eine Ressource gelöscht, wird diese beim erneuten öffnen nicht wieder in der Liste angezeigt. Es kann eine Option ausgewählt werden, damit auch alle Verknüpfungen und zusammenhängende Ressourcen, wenn vorhanden, mit veröffentlicht werden.

- **Mit zusammenhängenden Ressourcen veröffentlichen** alle neuen/geänderten Ressourcen, die mit der ursprünglichen Ressource (z.B. Bilder oder verknüpften Ressourcen) zusammenhängen.
- **Veröffentliche alle Verknüpfungen** veröffentlicht Ressourcen, die mit der ursprünglichen Ressource direkt verlinkt sind und geändert werden, wenn die ursprüngliche Ressource geändert wird.

3 Sitemap-Editor

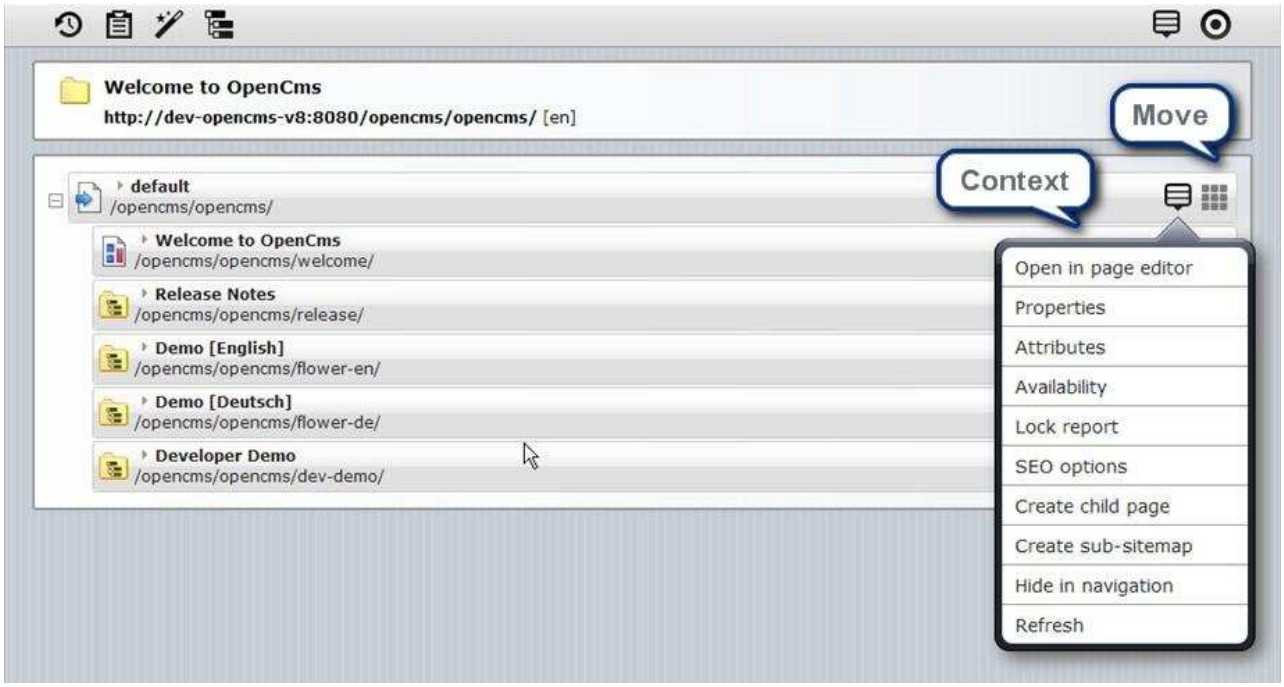
Im Sitemap-Editor kann die Navigation verändert werden, indem man eine neue Containerseite hinzufügt oder aber die bestehenden Containerseiten in der Reihenfolge ändert. In der Sitemap wird die Struktur der Website angezeigt. Der Benutzer kann Containerseiten im Verzeichnisbaum verschieben und kann zusätzliche Informationen wie z.B. Titel eingeben, kann neue Unterverzeichnisse erstellen oder kann die Reihenfolge der Seiten per Drag & Drop

verändern. Die Sitemap-Hierarchie beschreibt die Navigationsstruktur der Website und ist mit verantwortlich für die URL, die veröffentlicht wird und somit für alle sichtbar ist.

Es bildet außerdem den Verzeichnisbaum im Virtual File System (VFS) von OpenCms ab, aber er ist nicht exakt der Gleiche.

3.1 Editor öffnen

Wählen Sie in der Toolbar **Sitemap** aus und der Sitemap-Editor wird geöffnet. Nun wird der Verzeichnisbaum in der Reihenfolge angezeigt wie dieser in der Navigation erscheint. Beim Bewegen des Mauszeigers über eine Seite in der Sitemap, öffnet sich automatisch seitlich ein Menü: **Zeige Menü** und **Verschieben**



3.2 Sitemap-Editor Symbolleiste

Veröffentlichen: Zeigt den Veröffentlichen-Dialog an.

Zwischenablage: Zeigt gelöschte oder veränderte Seiten an.

Seite erstellen: Hiermit wird das Erstellen einer Seite ermöglicht.

Alle Seiten anzeigen: Wechselt zur VFS Explorer Ansicht, die alle Seiten anzeigt.

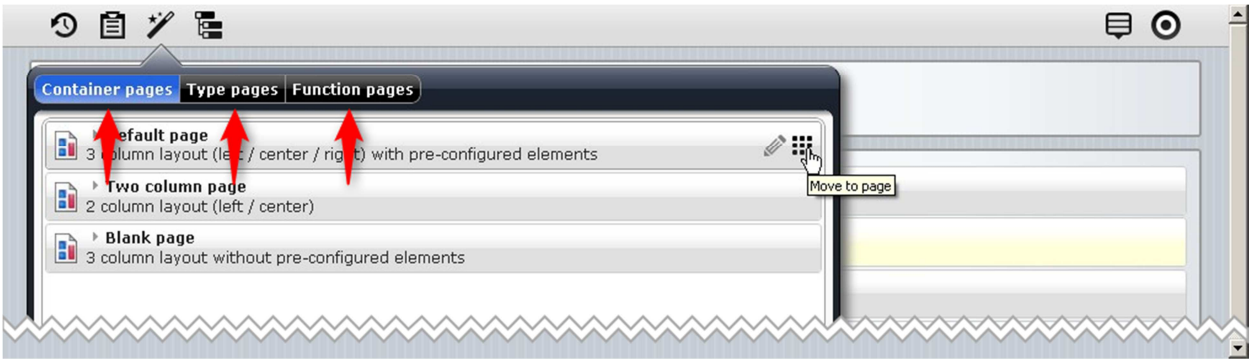
Kontextmenü: Zeigt das Kontextmenü der Sitemap.

Editierpunkt: Zurück zum Seiten Editor.

3.3 Veröffentlichen

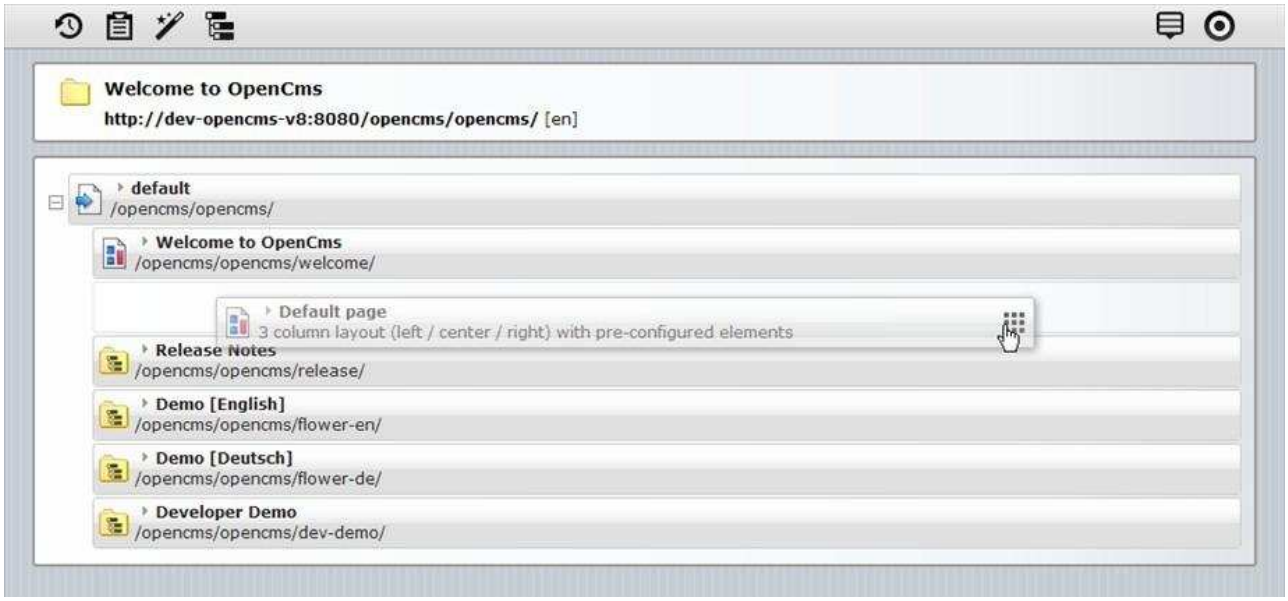
3.4 Seite erstellen

Hiermit kann eine neue Seite per Drag & Drop erstellt werden. Es kann aus drei verschiedenen Optionen ausgewählt werden: **Containerseiten, Typseiten, Funktionsseiten**.



3.4.1 Containerseiten

Abhängig von den konfigurierten Modellseiten, die der Template Designer zur Verfügung gestellt hat, können Sie aus verschiedenen Modellseiten, entsprechend Ihren Anforderungen auswählen. Sie können eine neue Containerseite, wo immer Sie diese brauchen, in die Sitemap ziehen.



Wenn die angezeigte Seite Unterseiten enthält, die nicht angezeigt werden (zum Anzeigen einfach das (+)-Zeichen anklicken), können neue Seiten auf der gleichen Navigationsebene eingefügt werden. Die Unterseiten werden angezeigt, wenn man mit der Maus während einer Drag & Drop-Aktion über einen Ordner-Symbol fährt.

3.4.2 Bearbeiten von Modellseiten

Um Modellseiten zu editieren, öffnen Sie im **Seite erstellen** Dialog den Containerseiten Tab. Wählen Sie dann die Option **Editieren** aus dem Kontextmenü einer neuen Containerseite. Sie müssen diese Operation bestätigen, da Sie vorhaben, die Vorlage für **alle neuen Seiten** zu

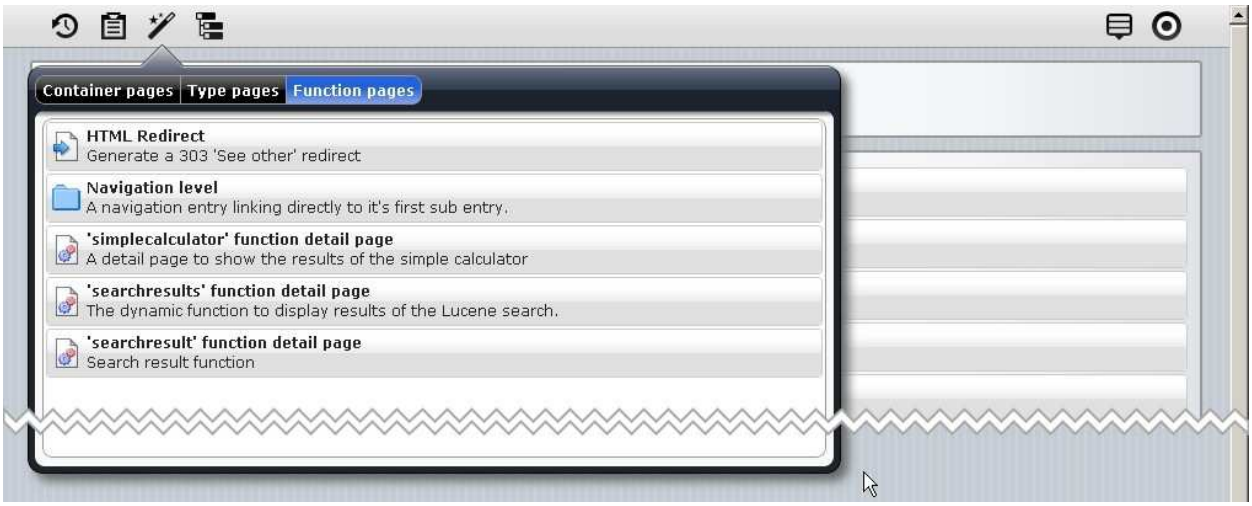
editieren. Diese Anpassung ändert bereits bestehende Seiten nicht. Diese Option öffnet die Vorlagenseite im Seiteneditor. Jetzt können Sie Elemente hinzufügen oder löschen. Lesen Sie mehr über Vorlagenseiten im Kapitel Modellseiten.

3.4.3 Typenseiten

Nicht für jede Ressource mit Inhalten muss eine neue Seite erstellt werden. Eine Detailseite ist ausreichend, um alle Inhalte eines Ressource-Typs anzuzeigen. Detailseiten werden verwendet, um auf Suchergebnisse, Inhalte aus Teaserboxen oder Content Kollektor Listen zu verlinken. Der detaillierte Inhalt wird in der Hauptspalte der Detailseite (abhängig von Template und Spalten) angezeigt und man kann durch einen automatisch generierten Link, welcher durch die **Titel**-Eigenschaft des Detailinhalts erzeugt wird, darauf zugreifen.

3.4.4 Funktionsseiten

Beim Anklicken von **Funktionsseite** im **Seite erstellen**-Dialog wird eine Auswahl an Seiten für spezielle Funktionen angezeigt.



3.4.4.1 HTML Redirect

Ein HTML Redirect könnte durch die Seitennavigation erreichbar sein und den Browser zu einer anderen URL innerhalb der Website oder zu einem externen Link weiterleiten. Er könnte auch von der Navigation ausgeschlossen, aber notwendig sein, wenn eine Unterseite an einen anderen Ort verschoben wurde, um tote Links durch gespeicherte Lesezeichen von Benutzern zu verhindern.

3.4.4.2 Navigationsebene

Lesen Sie mehr über die Navigationsebenen im Kapitel Navigationsebene

3.4.4.3 Funktionsdetailseiten

Lesen Sie mehr über die Funktionsdetailseiten im Kapitel 3.9.3.

3.5 Zwischenablage

Wenn die Option **Zwischenablage** über die Symbolleiste des Sitemap-Editors ausgewählt wird, öffnet sich ein neues Fenster. Es werden zwei Listen angezeigt, die kürzlich im Sitemap-Editor verändert wurden.

Geändert: Die **Geändert-Liste** wird standardmäßig angezeigt und beinhaltet Seiten, die im Seiten-Editor verändert wurden. Wenn Sie mit dem Mauszeiger über einen Listeneintrag fahren, erscheint auf der rechten Seite der Toolbar die Option, die Seite im Sitemap-Editor anzuzeigen. Wurde eine Auswahl getroffen, wird das Fenster geschlossen und die ausgewählte Seite wird angezeigt.

Gelöscht: In dieser Liste werden alle gelöschten Seiten der Sitemap angezeigt, die nicht veröffentlicht wurden. Durch Überfahren eines Eintrags mit dem Mauszeiger, wird die Wiederherstellen-Funktion angezeigt, welche den Originalstand der Ressource wieder herstellt bevor diese gelöscht wurde.

3.6 Zeige alle Dateien an

Wenn es notwendig ist, zeigt die **Zeige alle Dateien an-Option** den Verzeichnisbaum und alle enthaltenen Ressourcen im virtuellen Dateisystem der aktuellen OpenCms-Site an. Wenn diese Option ausgewählt ist, ist der Dialog **Seite erstellen** und auch das **Drag & Drop** deaktiviert. Im Verzeichnisbaum können über das Kontextmenü verschiedene Optionen ausgewählt werden. Beachten Sie, dass diese Option vielleicht nicht alle Ressourcen anzeigt, es ist abhängig davon, mit welchem Browser auf die Webseite zugegriffen wird. Somit können nicht alle Ressourcentypen richtig angezeigt werden.

Werden alle Ressourcen angezeigt, wird für jedes Element nur das Kontextmenü angezeigt. Das **Verschieben der Elemente** ist ausgeschaltet.

3.7 Kontextmenü

Das Kontextmenü zeigt alle verfügbaren Optionen:

- Eltern-Sitemap anzeigen
- Aliase bearbeiten
- Aktualisieren
- Zeige Workplace
- Abmelden

3.8 Editierpunkt

Durch einen Klick auf den Editierpunkt kehrt man zur neuesten, mit dem Seiteneditor geänderten Seite, zurück.

3.9 Seitentypen

3.9.1 Modellseiten

Für OpenCms 8 Templates muss mindestens eine Modellseite oder mehrere Modellseiten innerhalb des **Seite hinzufügen** Dialogs im Sitemap-Editor erstellt und konfiguriert werden. Beim Erstellen einer neuen Seite im Verzeichnisbaum, wird die Modellseite als Standard verwendet. Die bereits in der Modellseite hinzugefügten Inhalte werden automatisch in alle neu erstellten Seiten übernommen. Typischerweise gestaltet und konfiguriert der Template-Entwickler eine Modellseite sobald er die Template- und Inhaltstypen-Implementierung beendet hat, aber bevor die Inhalte der Webseite editiert werden.

- Klicken Sie hier, um die Modellseiten-Demo zu öffnen
- Klicken Sie hier, um die Modellseiten-Demo mit Elementgruppen zu öffnen

3.9.1.1 Erstellung einer Modellseite

1. Erstellen Sie eine neue Containerseite in der Explorer Ansicht.
2. Wenn die Template-Eigenschaften dem Seiten-Ordner nicht zugeordnet sind, setzen Sie die Template-Eigenschaften entsprechend.
3. Erstellen Sie verschiedene Modellseiten, eine für jedes Template, wenn die Website verschiedene Seiten mit unterschiedlichen Template-Konfigurationen hat.
4. Verschieben Sie die Modellseiten in den folgenden Ordner des Unterverzeichnisses: `/{sitefolder}/{sub sitemap path}/.content/.new/`
5. Öffnen Sie die Vorschau durch Anklicken der Seite und wählen Sie die Elemente aus, die beim Erstellen einer neuen Seite übernommen werden sollen.

3.9.1.2 Konfiguration

Konfigurieren Sie die Modellseite in der Sitemap-Konfigurationsdatei:

`/{sitefolder}/{sub sitemap path}/.content/.config`

Stellen Sie im Tab **Modellseiten** für jede Modellseite eine Verknüpfung her. Nun werden die verknüpften Modellseiten in der Liste der Containerseiten im **Seite erstellen-Dialog** im Sitemap-Editor angezeigt. Füllen Sie die Eigenschaften: Titel und Beschreibung aus.

Wenn Sie diese Modellseite als Standard einsetzen, wird sie als Vorlage für das Erstellen einer neuen Seite im oberen Teil des Kontextmenüs im Sitemap-Editor angeboten.

Verwenden Sie das Kontrollkästchen für die Modellseiten-Konfiguration, um die Sichtbarkeit der geerbten Modellseiten von den oberen Sitemaps zu steuern.

Bitte beachten Sie, dass Sie die Modellseite mit anderen Techniken wie Elementgruppen und Vererbungsgruppen kombinieren können. Dadurch ist eine verbesserte Organisation von gemeinsamen Inhalten auf der Website möglich.

3.9.2 Detailseiten

Eine Detailseite ist eine Technik mit der Sie benutzerfreundliche URLs für Inhalte erstellen können, die nicht in einer Containerseite sondern im /.content/ Verzeichnis erstellt wurden. Zum Beispiel durch Gebrauch eines Kollektors.

- Klicken Sie hier, um die Detailseiten-Demo zu öffnen

3.9.2.1 Anwendungsfall

In OpenCms 8 werden alle Ressourcen, die auf eine Containerseite per Drag & Drop gezogen wurden, automatisch in ein Verzeichnis aufgenommen, welches entweder in der Modul-Konfiguration oder in der Sitemap konfiguriert wurde. Normalerweise enthalten die Ressourcen-Pfade `/.content/`. Wenn Ressourcen aus einer Liste verlinkt werden, sollte der Editor nicht jede verknüpfte Ressource auf eine eigene Containerseite ziehen, um eine „saubere“ URL zu generieren. Die Idee ist, eine einzelne Seite in der Navigation zu erstellen, die Detailseite genannt wird. Sie nutzt die URL der Detailseite, kombiniert mit den Titel-Eigenschaften der Ressource und generiert daraus einen Ressourcen-Link ohne `/.content/` in der URL.

Generierte URL:

```
/dev-demo/collector-with-detail-page/generated_resource_title
```

`/dev-demo/collector-with-detail-page/` ist der Pfad der Detailseiten und durch `generated_resource_title` wird der URL-Anhang aus den zugeordneten Eigenschaften generiert.

3.9.2.2 Konfiguration

Die Konfiguration bezieht sich auf das Modul, das das Template und den Ressourcentyp definiert. Im Folgenden wird die XML-Schema- Definition auf den Inhaltstyp angepasst. Wenn der Typ kein Titel-Feld hat, benutzen Sie ein Feld, das dem Titel entspricht.

```
<mapping element="Title" mapto="urlName" />
```

Fügen Sie das Attribut `detailview="true"` in den `<cms:container>`-Tag des JSP-Templates hinzu, wo der XML-Content zu sehen sein soll. Normalerweise ist das die mittlere Spalte.

```
<cms:container name="centercolumn" type="center" width="450" detailview="true" .../>
```

Wenn Sie einen Kollektor benutzen wollen, um den Ressourcenpfad `${content.filename}` mit `<cms:link>` aufzulisten, übernimmt der `<cms:link>` die richtige generierte URL.

```
<%@page buffer="none" session="false" taglibs="c,cms" %>
<div>
   <cms:contentload collector="myCollector" param="..." editable="true">
       <%-- Access the content --%>
        <cms:contentaccess var="content" />
       <c:set var="link"><cms:link>${content.filename}</cms:link></c:set>
       <a href="${link}">${content.value.Title}</a>
   </cms:contentload>
</div>
```

3.9.2.3 Verwendung

- Öffnen Sie den **Seite erstellen**-Dialog im Sitemap-Editor
- Vom Tab **Typ-Seite** verwenden Sie die Seite für den Ressourcen-Typ durch einen Kollektor gelistet wird.
- Die Detailseite ist automatisch von OpenCms in der Sitemap Konfigurationsdatei konfiguriert.
- Jetzt sollten Links automatisch in einer Kollektor-Liste generiert werden.

3.9.3 Funktions-Detailseite

Die Funktions-Detailseite ist ein Detailseiten-Mechanismus für dynamische Funktionen. Dieser Dokumentationsteil beschreibt wie Sie eine Detailseite mit dynamischen Funktionen erstellen und konfigurieren. Sie nutzt Links einer lokal installierten OpenCms Version >=8.5

- Klicken Sie hier, um die Dynamic-Function-Demo zu öffnen
- Klicken Sie hier, um die Function-Detail-Page-Demo zu öffnen

3.9.3.1 Anwendungsfall

Daten in einem Formular sammeln, welches auf mehreren Seiten der Website angezeigt werden kann. Anzeigen der Ergebnisse der Benutzeranfrage auf einer Seite der Navigation. Zum Beispiel, die Integration eines Suchformulars auf mehreren Seiten der Website welches die Suchergebnisse auf nur einer Seite (einen Navigationspunkt) der Website darstellt. Seit der Version OpenCms 8.0.3 können Sie Funktions-Detailseiten nutzen, um solche Ergebnisseiten zu definieren.

3.9.3.2 Allgemeine Schritte

Folgende Schritte sind erforderlich, um eine dynamische Funktion und ihre Funktions-Detailseite zu erstellen:

- Erstellen Sie das dynamische Funktions-Element innerhalb des Moduls.
- Erstellen und Konfigurieren der Funktions-Detailseite für das Erstellen von dynamischen Funktionen
- Ziehen Sie, die dynamische Funktion in eine Containerseite.
- Ziehen Sie mit Hilfe des Sitemap-Editors die Funktion-Detailseite auf die Website-Struktur.

3.9.3.3 Erstellung dynamischer Funktionen

Da nur Benutzer mit der Rolle TEMPLATE_DEVELOPER die dynamischen Funktionselemente editieren können, sollte die Funktion im Modulverzeichnis erstellt werden:

`/system/modules/[module name]/functions/`

Um das Funktionselement von **Inhalte hinzufügen-Dialog** zu verstecken, setzen Sie die folgenden Eigenschaften:

Die Drag & Drop-Option ist nutzbar für die Benutzer mit der Rolle WORKPLACE_USERS.

3.9.3.4 Erstellen der Funktion-Detailseiten

Implementiere die JSP mit einem Formular, welches auf mehreren Containerseiten der Website genutzt werden soll. Benutze folgenden Wert `${cms.functionDetail['Name of the function detail page']}` für das **action-Attribut** des `<form>`-Tags.

```
<%@page buffer="none" session="false" taglibs="c,fn,cms" %>
<div class="box box_schema1">
   <form action="${cms.functionDetail['simplecalculator']}" method="post">  …
   </form>
</div>
```

- Implementieren Sie eine andere JSP, die die Formulardaten auswertet und die HTML-Ausgabe für die Detailansicht generiert.
- In der Explorer-Ansicht erstellen Sie zwei dynamische Funktionselemente im Ordner `/system/modules/[module name]/functions/` des Moduls. Eine der dynamischen Funktionen sollte auf die Formular-JSP zeigen und eine andere auf die Detailseiten-JSP.
- Editieren Sie die Modul-Konfigurationsdatei `system/modules/[module name]/.config` und konfigurieren Sie die Funktions-Detailseite imTab **Funktionen**. Fügen Sie ein neues Feld **Benannte Funktion** hinzu. Definieren Sie einen eindeutigen Namen für die Funktion. Dies sollte der gleiche Namen sein, welcher in der Formular-JSP als Action-Parameter verwendet wird. Nach Abschluss der Konfiguration der Funktions-Detailseite erscheint in der Registerkarte **Funktions-Seiten** der **Seite erstellen** Dialog im Sitemap-Editor.
- Bearbeiten Sie die Eigenschaft **container.info** auf der Template-JSP. Setzen Sie den Eigenschaftswert `functionDetail=[container name attribute]`. Verwenden Sie als Wert das Name-Attribut des Containers, in dem die Funktionergebnisse angezeigt werden sollten.

3.9.3.5 Verwendung

- Ziehen Sie das dynamische Funktions-Element mit dem Formular auf die Containerseite.
- Öffnen Sie den "Seite erstellen" Dialog des Sitemap-Editors.
- Verwenden Sie die konfigurierte Funktions-Detailseite im Tab "Funktions-Seiten" als Modellseite, um die neue Seite zur Navigation hinzuzufügen.

3.9.4 Navigationsebene

Die Navigationsebene ist ein spezieller Navigationsordner, der zur ersten Containerseite innerhalb des Ordners weiterleitet. Verwenden Sie diese Option, wenn Sie mehrere Sub-Seiten auf einer Navigationsebene ohne Übersichtseite haben wollen.

- Klicken Sie hier um die Navigation-Level-Demo zu öffnen

3.9.4.1 Anwendungsfall

Auf der Website gibt es eine Liste der Produktbeschreibungen, die in der Navigation unter Produkte aufgelistet werden sollen. Wenn der Benutzer die Ordner in der Navigation wählt, ist das erste Produkt, das direkt ausgewählte Element. Es gibt keine Übersichtsseite. Der erste Eintrag in der Liste wird auch angezeigt, nachdem die Reihenfolge der Elemente geändert wurde. OpenCms 8.5 führt eine neue Seite des Typs **Navigationsebene** hinzu, um einen Navigations-Einstiegspunkt zu definieren, welcher automatisch umleitet zur ersten Sub-Seite der Navigation.

3.9.4.2 Verwendung der Navigationsebene

- Öffnen Sie den Tab Funktions-Seiten im Sitemap-Editor.
- Ziehen Sie eine **Navigationsebene** per Drag & Drop in die Sitemap.
- Fügen Sie per Drag & Drop mehrere Unterseiten unter dem neuen Navigationspunkt hinzu.

3.9.4.3 Details zur Implementierung

Der **Navigations-Tag** `<cms:navigation>` bietet volle Unterstützung für die Navigationsebenen. Bauen Sie den Link zur **Navigationsebene** wie üblich. Die Klassen `CmsJspNavigationElement` und `CmsJspNavBuilder` erkennen automatisch die **Navigationsebenen** und erzeugen direkt den Link auf das erste Element in der Liste. Mehr ist nicht notwendig.

Styling – Die Klasse `CmsJspNavElement` bietet eine neue Methode, um die **Navigationsebene** in der Navigation zu erkennen. Diese Informationen können z. B. für das Styling nützlich sein. Im folgenden Beispiel wird die aktuelle CSS-Klasse verwendet, um das ausgewählte Element in der Navigation zu markieren. Das **Navigationsebenen-Element** ist nicht markiert.

Beispiel:

```
<cms:navigation type="treeForFolder" startLevel="1" endLevel="4" var="nav"/>
   <c:forEach items="${nav.items}" var="elem"> […]
   <c:set var="link"><cms:link>${elem.resourceName}</cms:link></c:set>
   <a href="${link}">
      <c:if test="${nav.isActive[elem.resourceName] and !elem.navigationLevel }">
            class="current"
      </c:if> >${elem.navText}</a>       […]
</c:forEach>
```

Bitte schauen Sie sich die folgende JSP an, um die komplette Navigation zu sehen.

- Beispiel auf GitHub.

3.9.5 Versteckte Einträge

Versteckte Einträge sind Containerseiten oder andere Seiten, die nicht im Navigationsmenü sichtbar sind, wenn diese mit dem `CmsJspNavBuilder` gebaut sind.

3.9.5.1 Beschreibung

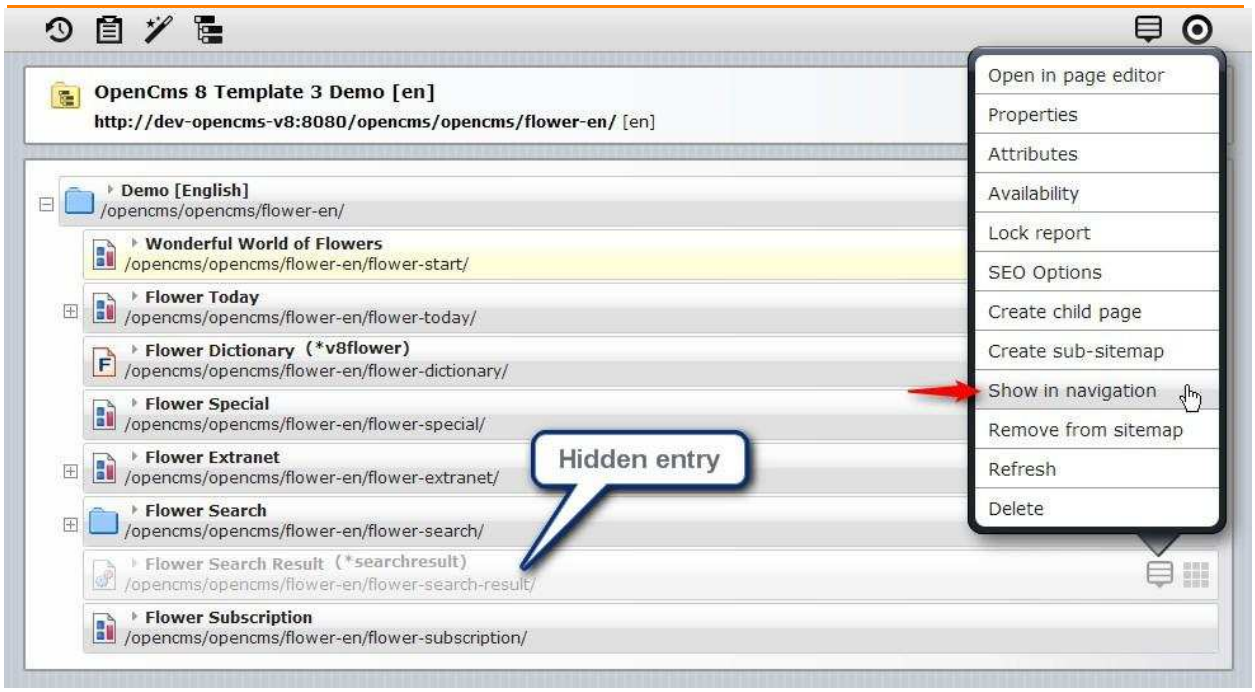
Seit OpenCms 8.5 können Ressourcen, welche nicht in der Navigation auf der Website enthalten sein sollen, direkt im Sitemap-Editor ausgeblendet werden. Die verborgenen Seiten sind noch im Sitemap-Editor sichtbar und können bearbeitet werden.

Diese neue Funktion soll für Seiten nützlich sein, die in der Navigation ausgeblendet sind, aber

- unter den vorgegebenen URLs direkt erscheinen: wie Kontakte, Suche, Standortseiten usw.
- Detailseiten für die Ressourcen anzeigen.

3.9.5.2 Verwendung

1. Öffnen Sie den Sitemap-Editor.
2. Öffnen Sie das Kontextmenü und wählen Sie **Verstecke in Navigation**.
3. Verwenden Sie im Kontextmenü **Zeige in Navigation**, um die Navigations-Eigenschaften für die Seite zu aktivieren.

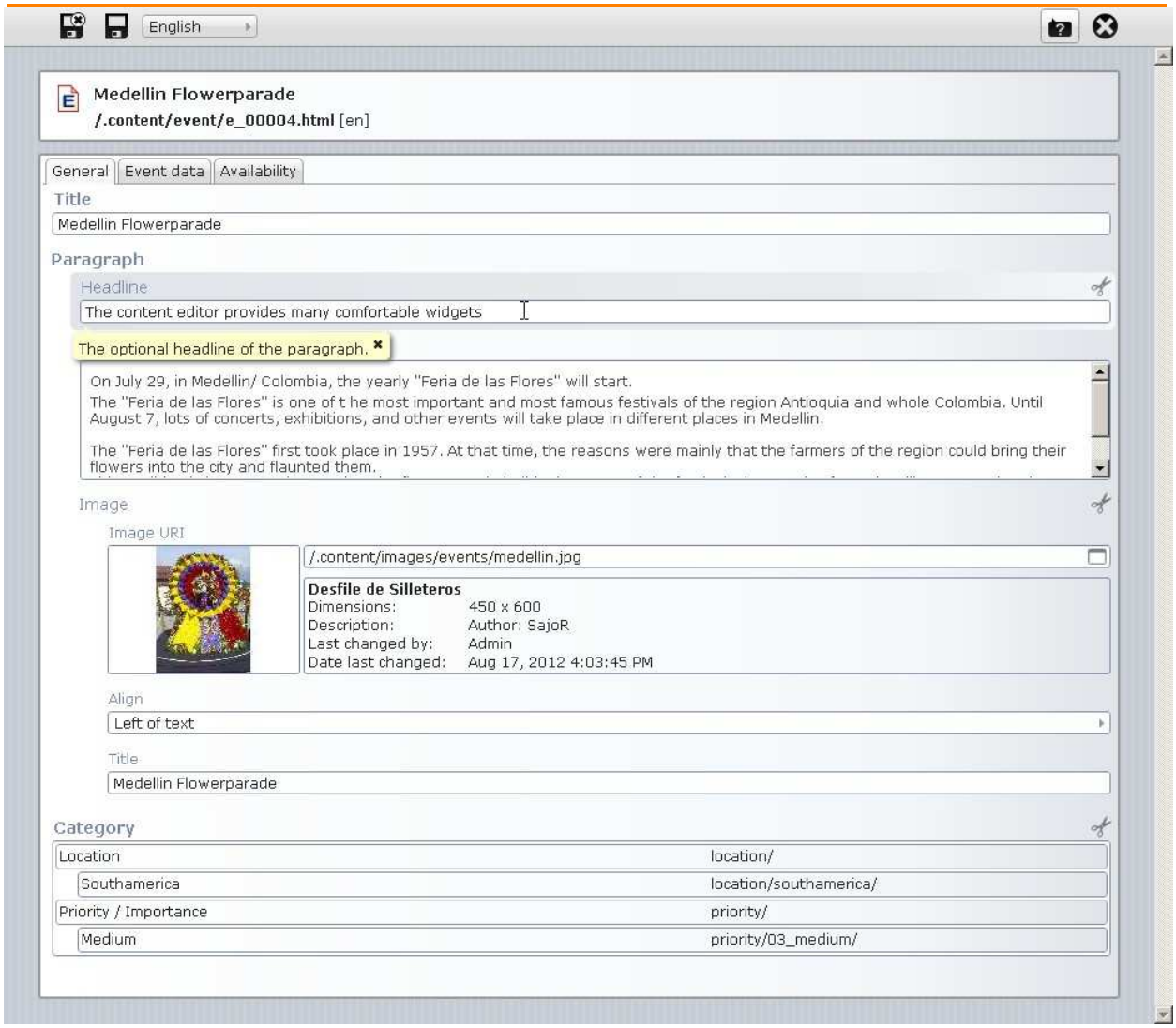


4 Inhaltseditor

Der Inhaltseditor wurde in OpenCms 8.5 komplett in HTML 5 umgeschrieben um eine einfachere Nutzung und bessere Performance zu ermöglichen. Der Editor wird automatisch erzeugt, basierend auf XML-Schemata, die die Struktur von Inhaltstypen beschreiben. Er stellt eine mächtige Benutzerschnittstelle für die Redakteure zur Verfügung, die verwendet wird, um auf XML-Schemata gestützte XML-Dateien zu erzeugen. Die erzeugte XML-Datei wird dann im OpenCms Repository gespeichert.

Hauptfunktionen des Inhaltseditors sind:

- Läuft im Client und stellt eine sehr schnell reagierende Benutzeroberfläche zur Verfügung.
- Automatisch aus XML-Schemas generiert, (Programmierung erforderlich, um diese Schemas zu erstellen).
- Bietet eine große Auswahl an reichhaltigen Benutzeroberflächen-Widgets.
- Kann mit benutzerdefinierten Benutzeroberflächen, mit Widgets für spezielle Anwendungen erweitert werden.
- Generiert XML-Dateien, die gegen das XML-Schema validiert und in der Datenbank gespeichert werden.



4.1 Widgets

Widgets werden verwendet, um einen passenden Inhaltseditor für den XML-Inhalt zu erzeugen. Die zu verwendenden Widgets werden in der XSD (Schema-Definition) des Inhaltstyps definiert. Im Vergleich zum alten XML-Inhaltseditor sind einige Konfigurationsparameter hinzugefügt, geändert oder entfernt worden. In den folgenden Abschnitten wird jedes Widget und seine Parameter mit einem kurzen Beispiel beschrieben.

Achtung: Die Widget-Konfiguration bestimmt im Inhaltseditor nur welcher GUI-Bestandteil für die Werteauswahl genutzt wird – nicht aber welcher Datentyp zu verwenden ist. Z.B. können Sie ein Datum mit dem Datums-Picker auswählen. Wenn Sie aber OpenCms nicht mitteilen, dass der Typ `OpenCmsDateTime` zu verwenden ist, dann wird er nicht als solcher behandelt. Der Datentyp eines im XSD definierten Elements ist für das interne Handling der Werte verantwortlich. Der Gebrauch von Datentypen vergrößert die Datenqualität, was nicht nur Inhaltssuche und Performance betrifft, sondern auch die gesamte Datenintegrität z.B. das Funktionieren interner Links.

In den folgenden Abschnitten werden Sie Beispielcodes finden, die zeigen, wie die zur Verfügung gestellten Widgets verwendet und konfiguriert werden. Wenn ein bestimmter Datentyp für den Gebrauch eines bestimmten Widgets sinnvoll ist, dann enthält der Beispielcode den `<xsd:element>`-Knoten. Sonst zeigen die Beispiele nur die Widget-

Konfiguration, die im `<layout>`-Knoten gemacht wird. Hierbei wird dann angenommen, dass das entsprechende Element `OpenCmsString` als Attributtyp verwendet.

Verfügbare XML Feldtypen

`OpenCmsBoolean` – Repräsentiert einen booleschen Wert (WAHR/FALSCH)
`OpenCmsCategory` – Kann ein oder mehr ausgewählte Kategorien behandeln
`OpenCmsColor` – Speichert einen hexadezimalen Farbwert
`OpenCmsDateTime` – Behandelt ein Datumswert
`OpenCmsHtml` – Wird benutzt für Rich-Text-Editorfelder (interne Linkrelationen bleiben bestehen; z.B. Links im Fließtext)
`OpenCmsLocale` – Speichert den Locale-Wert
`OpenCmsPlainTextString` – Speichert den extrahierten Text für ein Feld, das HTML enthält
`OpenCmsString` – Speichert einen einfachen String 1:1
`OpenCmsVarLink` – Kann interne und externe Links behandeln
`OpenCmsVfsFile` – Vorgesehen um interne Datei-Referenzen zu speichern
`OpenCmsVfsImage` – Verweist auf interne Bilder (behandelt Bildformate, Skalierung, Beschreibung …)

4.1.1 String-Widget

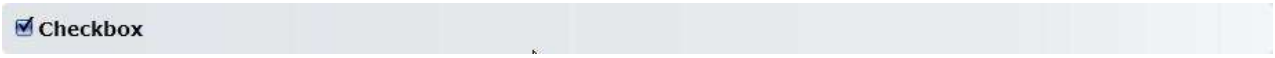
Das **String-Widget** bietet ein einfaches Text-Eingabefeld, um Klartext einzugeben, der als String-Wert intern gespeichert wird.

Dieses Widget benötigt keine Konfiguration und kann wie folgt definiert werden:

```
<xsd:element name="Input" type="OpenCmsString" />
```

4.1.2 Boolean-Widget

Das **Boolean-Widget** unterstützt eine Checkbox und speichert intern einen booleschen Wert.



Dies Widget benötigt keine Konfiguration und kann wie folgt definiert werden:

```
<xsd:element name="Checkbox" type="OpenCmsBoolean" />
```

4.1.3 Display-Widget

Das **Display-Widget** zeigt nur den Wert an, ohne die Möglichkeit diesen zu modifizieren.

Der Wert des Display-Widgets wird als ein Attribut im Layoutknoten konfiguriert.

Eine Konfiguration kann in etwa so aussehen:

```
<layout element="Di" widget="DisplayWidget" configuration="This is no default string"/>
```

4.1.4 Select-Widget

Das **Select-Widget** stellt eine Dropdown-Auswahl zur Verfügung.

Das Layout für dieses Widget kann in etwa so aussehen:

```
<layout element="Select" widget="SelectorWidget" configuration="1*|2|3|4|5|6|7" />
```

Lese Select-Widget-Konfiguration

4.1.5 Textarea-Widget

Das **Textarea-Widget** stellt einen HTML-Eingabebereich zur Verfügung, welcher gescrollt werden kann, wenn der Text zu lang ist, um komplett angezeigt zu werden. Das Feld kann in der Größe angepasst werden.

Im Konfigurationsparameter können Sie die Anzahl von Zeilen definieren, die das Textfeld haben soll, wenn es initial geöffnet wird. Wenn keine Konfiguration angegeben wird, wird als Standardwert fünf Zeilen angenommen.

```
<layout element="Textarea" widget="TextareaWidget" configuration="7" />
```

4.1.6 Radio-Button-Widget

Das **Radio-Button-Widget** stellt eine Gruppe von Radio-Buttons mit einer Wertzuweisung zur Verfügung.

Die Konfiguration könnte wie folgt aussehen:

```
<layout element="Radiobutton" widget="RadioSelectWidget"
       configuration="Radiobutton1*|Radiobutton2|Radiobutton3|
                      Radiobutton4|Radiobutton5|Radiobutton6" />
```

Die Konfigurationssyntax ist identisch mit der Select-Widget-Konfiguration.

4.1.7 Multiselect-Widget

Das **Multiselect-Widget** stellt eine Gruppe von Kontrollkästchen zur Verfügung, um ein oder mehrere Werte auszuwählen.

Die Konfiguration könnte wie folgt aussehen:

```
<layout element="Multi" widget="MultiSelectWidget" configuration="1*|2*|3*|4|5|6|7"/>
```

Die Konfigurationssyntax ist identisch mit der Select-Widget-Konfiguration.

4.1.8 Combo-Widget

Ein **Combo-Widget** bietet eine vordefinierte Anzahl an Auswahloptionen mit der Möglichkeit zur individuellen Texteingabe an.

Die Konfiguration könnte wie folgt aussehen:

```
<layout element="Combobox" widget="ComboWidget" configuration="1|2|3|4|5|6|7" />
```

Die Syntax ist identisch mit der Select-Widget-Konfiguration.

4.1.9 Ressourcentyp-Combo-Widget

Dieses spezielle **Ressourcentyp-Combo-Widget** stellt eine Liste aller in OpenCms konfigurierten Ressourcentypen zur Verfügung.

Dieses Widget benötigt keinen Konfigurations-String. Der Layout-Knoten könnte wie folgt aussehen:

```
<layout element="TypeCombo" widget="TypeComboWidget" />
```

4.1.10 HTML-Widget

Das **HTML-Widget** verwendet TinyMCE, der für viele spezielle CMS-Funktionalitäten erweitert worden ist.

Mit dem Widget **HtmlWidget** können Optionen für jedes Element des Typs **OpenCmsHtml** definiert werden. Sie müssen im Annotationsbereich der XSD hinterlegt werden. Das Konfigurationsattribut im Layoutbereich muss die aktivierten Optionen als kommaseparierten String-Wert enthalten:

```
<xsd:element name="Text" type="OpenCmsHtml" />

<layout element="Text" widget="HtmlWidget"
        configuration=" link,anchor,imagegallery,downloadgallery,formatselect"/>
```

Verfügbare Optionen sind:

- **anchor**: der Linkdialog-Button.
- **buttonbar**: ${Buttons, durch ';' getrennt}: eine individuelle -Symbolleisten-Konfiguration.
- **css**:/vfs/pfad/zur/cssfile.css: der absolute Pfad im OpenCms VFS zur CSS-Formatvorlage, um die Inhalte im Editor zu rendern (die Verfügbarkeit hängt vom integrierten Editor ab).
- **formatselect**: die Formatauswahl, um Textformate wie Absätze oder Überschriften auszuwählen.
- **formatselect.options**: ${Liste von Optionen, durch ';' getrennt}: die Optionen, die in der Format-Auswahl verfügbar sein sollen, z.B. `formatselect.options: p; h1; h2`.
- **fullpage**: der Editor wird im Fullscreen-Modus angezeigt.

${gallerytype}: Zeigt einen Galerie-Dialog-Button, z.B. zeigt `imagegallery` den Bildgalerie-Button, oder `downloadgallery` zeigt den Download-Galerie-Button.

- **height**:${editorheight}: die Editor-Höhe, die in 'px' oder '%', z.B. 400px, angegeben ist.
- **hidebuttons**: ${Liste von Buttons, getrennt durch ';', die nicht angezeigt werden sollen}: versteckt die Buttons, die gewöhnlich in der Default-Button-Bar erscheinen, z.B. `hidebuttons:bold; italic; underline; strikethrough` verbirgt einige Formatierungsknöpfe.
- **image**: der Bild-Dialog-Button (Verfügbarkeit hängt vom integrierten Editor ab).
- **link**: der Link-Dialog-Button.
- **source**: zeigt den Quellcode-Umschalter.
- **stylesxml**:/vfs/pfad/zur/stylefile.xml: der absolute Pfad im OpenCms VFS zu den benutzerdefinierten CSS-Styles, die im Style-Selektor angezeigt werden sollen (Verfügbarkeit hängt vom integrierten Editor ab).

- **stylesformat**:/vfs/pfad/zur/stylefile.xml: der absolute Pfad im OpenCms VFS zu den benutzerdefinierten CSS-Styles, die im Style-Selektor angezeigt werden sollen (Verfügbarkeit hängt vom integrierten Editor ab).
- **table**: der Tabellen-Dialog-Button (Verfügbarkeit hängt vom integrierten Editor ab).

Einige Konfigurationen, wie die Button-Bar, sollten in der globalen Widget-Konfiguration in der Datei opencms-vfs.xml definiert werden.

4.1.11 Lokalisierungs-Widget

Das **Lokalisierungs-Widget** bietet ein Standard-HTML-Formular-Eingabefeld für das Überschreiben lokalisierter Werte eines Ressource-Bundles. Das Ressource-Bundle wird durch das Widget-Konfigurations-Attribut konfiguriert. Es kann auch ein optionaler Name des Schlüssels zum Nachschlagen im Bundle angegeben werden, falls er sich vom Elementnamen key=mykey unterscheidet.

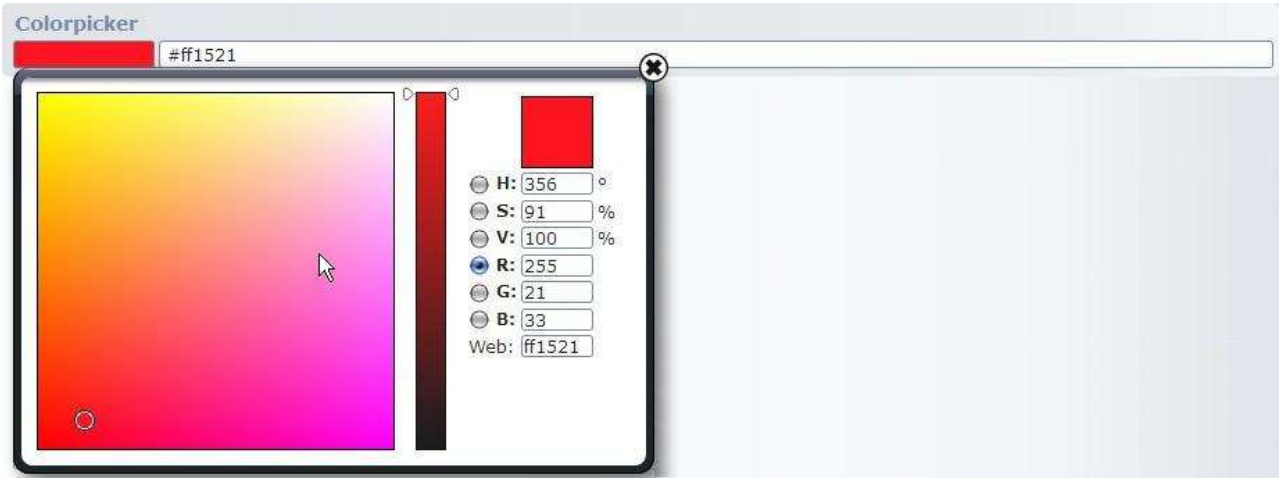
Um den Wert einer Locale zu erhalten, muss eine Konfigurationsanweisung im Layout-Tag angegeben werden: z.B. `locale=en`.

```
<layout element="Localization" widget="LocalizationWidget"
        configuration="org.opencms.workplace.messages|key=mykey|locale=en" />
```

Um die gespeicherten Lokalisierungswerte zu verwenden und die Werte des **Ressource-Bundles** als Fallback zu haben, verwenden Sie das `CmsXmlMessages`-Objekt.

4.1.12 Color-Picker-Widget

Mit dem **Color-Picker-Widget** können Sie eine Farbe wählen.



Dieses Widget braucht keine Konfiguration, es muss nur der Element-Typ **`OpenCmsColor`** verwendet werden.

```
<xsd:element name="Colorpicker" type="OpenCmsColor" />
```

4.1.13 Date-Picker-Widget

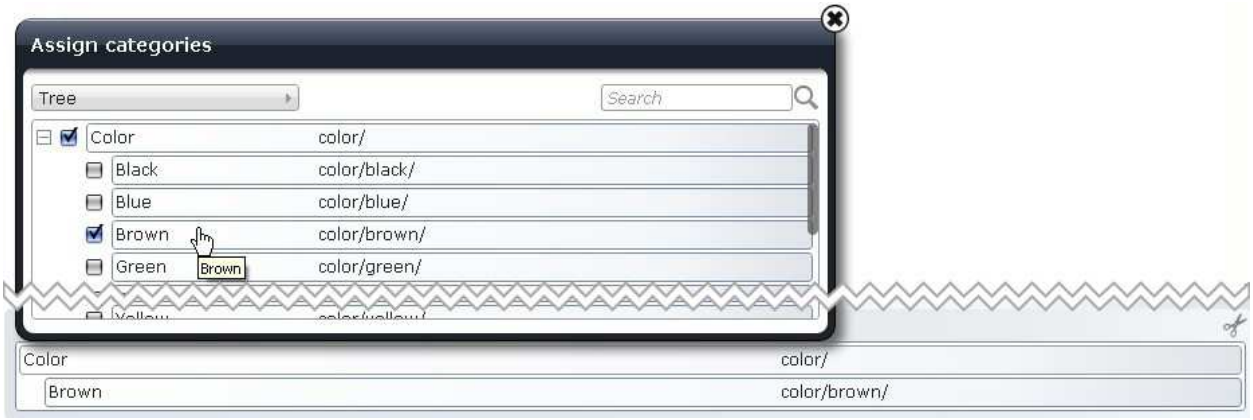
Das **Date-Picker-Widget** bietet eine einfache Möglichkeit um ein Datum zu spezifizieren.

Dieses Widget braucht keine Konfiguration, es muss nur der Element-Typ **`OpenCmsDateTime`** verwendet werden.

```
<xsd:element name="Calendar" type="OpenCmsDateTime" />
```

4.1.14 Kategorie-Widget

Das Kategorie-Widget stellt auf bequeme Weise eine Kategorie-Auswahl zur Verfügung.



Dieses Widget hat einige optionale Konfigurationsparameter:

- **category**: Pfad zur Root-Kategorie, um die gezeigten Kategorien auf seine Kinder einzuschränken.
- **onlyleafs**: Startet die Auswahl in einer Listenansicht
- **parentSelection**: Signalisiert, ob die Eltern der ausgewählten Kategorien auch zu speichern sind.

Das **Kategorie-Widget** sollte Elementen des Typs `OpenCmsCategory` zugeordnet werden, um die Speicherung von Mehrfach-Selektionen zu ermöglichen. Aus Gründen der Rückwärtskompatibilität ist es noch immer möglich, Elemente des Typs `OpenCmsVfsFile` mit dem **Kategorie-Widget** hinzuzufügen. Dann ist es aber nicht möglich, eine Mehrfachauswahl vorzunehmen. Eine Konfiguration könnte wie folgt aussehen:

```
<xsd:element name="Category" type="OpenCmsCategory" />

<layout element="Category" widget="CategoryWidget"
    configuration="category=onlyleafs=false|parentSelection"/>
```

4.1.15 Gruppen-Widget

Das **Gruppen-Widget** ist eine spezielle Auswahlbox, die eine Liste aller OpenCms-Benutzergruppen anbietet.

Dieses Widget braucht keine Konfiguration und kann wie folgt verwendet werden.

```
<layout element="Group" widget="GroupWidget" />
```

4.1.16 Multi-Gruppen-Widget

Mit dem **Multi-Gruppen-Widget** können Sie mehrere Gruppen auf einmal auswählen.

Dieses Widget ist mit den folgenden Optionen konfigurierbar:

- **groupfilter**: regulärer Ausdruck, um verfügbare Gruppen zu filtern.
- **groups**: eine Komma-separierte Liste von Gruppennamen die in der Auswahlbox angezeigt wird. Merken Sie sich bitte, wenn diese Konfigurationsauswahl genutzt wird, werden `groupfilter` und `includesubous` nicht mehr berücksichtigt.
- **includesubous**: Boolescher Flag, um anzuzeigen, ob Sub-OUs für auszuwählende Gruppen gescannt werden sollten.
- **oufqn**: der „voll qualifizierte Name“ der OU, um die Gruppen zu lesen.

Um die ausgewählte Gruppe zu einer Berechtigung zuzuordnen, verwenden Sie die folgende Zuordnungskonfiguration:

```
<mapping element="..."
mapto="permission:GROUP:+r+v|GROUP.ALL_OTHERS:|GROUP.Projectmanagers:+r+v+w+c" />
```

Das bedeutet, dass die +r+v Berechtigung für die Principal GROUP in die Quelle geschrieben wird. Zusätzlich werden zwei Berechtigungen als Default geschrieben: für ALL_OTHERS wird keine Berechtigung vergeben. Für Projektmanager wird, "+r+v+w+c" gesetzt.

Dieses Widget braucht keine Konfiguration und kann wie folgt aussehen:

```
<layout element="GroupMulti" widget="GroupMultiSelectorWidget" />
```

4.1.17 VFS-Datei-Widget

Das **VFS-Datei-Widget** stellt eine Auswahl einer Datei im VFS zur Verfügung. Es kann entweder direkt ein Link zu einer Datei eingetippt oder aus einer Galerie ausgewählt werden. Diese Galerie erscheint, wenn der Button auf der rechten Seite des Eingabefeldes angeklickt wird.

Dieses Widget bietet folgende Konfigurationsmöglichkeiten:

- **hidesiteselector, showsiteselector**: Die Seitenauswahl wird ausgeblendet oder angezeigt (Standard)
- **excludefiles, includefiles**: Dateien werden im Verzeichnisbaum-Popup ausgeblendet oder angezeigt (Standard)
- **notprojectaware, projectaware**: Zeigt oder versteckt (Standard) nur Dateien des aktuellen Projektes
- **startsite**: Die Seite mit der das Verzeichnisbaum-Popup geöffnet werden sollte

Konfigurationsbeispiel:

```
<xsd:element name="VfsFile" type="OpenCmsVfsFile" />

<layout element="VfsFile" widget="VfsFileWidget"
        configuration="startsite=/sites/www.domain.org/repository|hidesiteselector" />
```

4.1.18 VFS-Bilder-Widget

Das **VFS-Bilder-Widget** erlaubt es, Bilder auszuwählen und bietet eine Reihe von Funktionen an wie die Größe anzupassen, zuschneiden usw.

Konfiguration

Die Konfiguration muss als JSON Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class**: optionaler Klassenname, der eine dynamische Start-Konfiguration implementiert und spezielle Format-Werte durchführt. Muss ein voll qualifizierter Klassenname sein
- **formatnames**: Liste von Formatnamen, die zur Auswahl stehen, mit Paaren von auswählbaren Werten und auswählbaren Texten, z.B. value1:optiontext1|value2:optiontext2
- **formatvalues**: entsprechende Formatwerte zur Format-Namensliste. Kann durch die dynamische Konfigurationsklasse dynamisch erzeugt werden. Die Liste von Werten sollte Breiten- und Höhen-Informationen enthalten. Mit einem '?' als Kürzel für die dynamische Größe und mit einem 'x' als Separator für die Breite und Höhe. Beispiel: ['200x? ', '800x600']
- **scaleparams**: Standardskalierungs-Parameter (Breite, Höhe oder Zuschneiden sollte nicht angeboten werden!)
- **startup**: der Startup-Ordner, kann durch die zur Verfügung gestellte Klasse dynamisch erzeugt werden. In diesem Fall ist `dynamic` als Wert zu verwenden
- **type**: der Startup-Ordner-Typ, kann `gallery` oder `category` sein. Kann durch die zur Verfügung gestellte Klasse dynamisch erzeugt werden. In diesem Fall ist `dynamic` als Wert zu verwenden
- **usedescription**: zeigt an, ob die Beschreibung des Eingabefeldes für das Bild gezeigt werden soll oder nicht
- **useformat**: zeigt an, ob die Format-Auswahlbox für das Bild gezeigt werden soll oder nicht

Eine JSON-Beispiel-Konfiguration:

```
{scaleparams: 'q:70,r:2,c:CCCC00',
    type: 'gallery',startup: '/demo_en/images/',usedescription: true,useformat: true,
    formatnames: 'imageleft:Image left|imageright:Image right|imagetop:Image top',
```

```
    formatvalues: ['150x?', '250x300', '?x250']}
```

Komplettes Beispiel um das VFS-Bild-Widget in einem XSD zu konfigurieren:

```
<xsd:element name="VfsImage" type="OpenCmsVfsImage" />
<layout element="Widget" widget="VfsImageWidget" configuration="
    {useformat:true,usedescription:true,formatnames:'left:Left|right:Right|top:Top',
     ['150x?','250x300','?x250']}"/>
```

4.1.19 Bildergalerie-Widget

Das **Bildergalerie-Widget** stellt eine Auswahl von Dateien aus der Bildgalerie zur Verfügung. Diese Galerie erscheint durch das Anklicken des Buttons auf der rechten Seite des Eingabefeldes.

Die Konfigurationseinstellungen werden von dem Konfigurations-String des Widgets gelesen. Für verschachtelte XML-Schemas muss der Konfigurations-String innerhalb des verschachtelten Inhalts definiert werden. Der Konfigurations-String muss als JSON-Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class:** optionale Klasse, die dynamische Startup-Konfigurationen und Formatwerte implementiert.
- **startup:** Den Start-Ordner, auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.
- **type:** kann `gallery` oder `category` sein. Auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.
- XSD Konfigurationsbeispiel:

```
<xsd:element name="ImageGallery" type="OpenCmsVfsFile" minOccurs="1"/>

<layout element="ImageGallery" widget="ImageGalleryWidget"
         configuration="{type: 'gallery', startup: '/demo_en/images/'}"/>
```

4.1.20 Download-Galerie-Widget

Das **Download-Galerie-Widget** stellt eine Auswahl an VFS-Ressourcen zur Verfügung.

Die Konfigurationseinstellungen werden von dem Konfigurations-String des Widgets gelesen. Für verschachtelte XML-Schemas muss der Konfigurations-String innerhalb des verschachtelten Inhalts definiert werden. Der Konfigurations-String muss als JSON-Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class:** optionale Klasse, die dynamische Startup-Konfigurationen und Formatwerte implementiert.

- **startup:** Den Start-Ordner, auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.
- **type:** kann `gallery` oder `category` sein. Auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.

Konfigurationsbeispiel:

```
<xsd:element name="DownloadGallery" type="OpenCmsVsfFile" minOccurs="1"/>

<layout element="DownloadGallery" widget="LegacyDownloadGalleryWidget"
        configuration="{type: 'gallery', startup: '/demo_en/images/'}"/>
```

4.1.21 HTML-Galerie-Widget

Das **HTML-Galerie-Widget** stellt eine Auswahl an Dateien aus der HTML-Galerie zur Verfügung. Diese Galerie erscheint durch das Anklicken des Buttons auf der rechten Seite des Eingabefeldes.

Die Konfigurationseinstellungen werden von dem Konfigurations-String des Widgets gelesen. Für verschachtelte XML-Schemas muss der Konfigurations-String innerhalb des verschachtelten Inhalts definiert werden. Der Konfigurations-String muss als JSON-Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class:** optionale Klasse, die dynamische Startup-Konfigurationen und Formatwerte implementiert.
- **startup:** Den Start-Ordner, auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.
- **type:** kann `gallery` oder `category` sein. Auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.

Konfigurationsbeispiel:

```
<layout element="HtmlGallery" widget="HtmlGalleryWidget"
        configuration="{type: 'gallery', startup: '/demo_en/html/'}"/>
```

4.1.22 Tabellen-Galerie-Widget

Das **Tabellen-Galerie-Widge**t stellt eine Auswahl der Dateien aus der Tabellen-Galerie zur Verfügung. Diese Galerie erscheint durch das Anklicken des Buttons auf der rechten Seite des Eingabefeldes.

Die Konfigurationseinstellungen werden von dem Konfigurations-String des Widgets gelesen. Für verschachtelte XML-Schemas muss der Konfigurations-String innerhalb des verschachtelten Inhalts definiert werden. Der Konfigurations-String muss als JSON-Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class:** optionale Klasse, die dynamische Startup-Konfigurationen und Formatwerte implementiert.

- **startup:** Den Start-Ordner, auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.

type: kann `gallery` oder `category` sein. Auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.

Konfigurationsbeispiel:

```
<layout element="TableGallery" widget="TableGalleryWidget"
        configuration="{type: 'gallery', startup: '/demo_en/table/'}"/>
```

4.1.23 Link-Galerie-Widget

Das **Link-Galerie-Widget** stellt eine Auswahl von Dateien in der Link-Galerie zur Verfügung. Diese Galerie erscheint durch das Anklicken des Buttons auf der rechten Seite des Eingabefeldes.

Die Konfigurationseinstellungen werden von dem Konfigurations-String des Widgets gelesen. Für verschachtelte XML-Schemas muss der Konfigurations-String innerhalb des verschachtelten Inhalts definiert werden. Der Konfigurations-String muss als JSON-Objekt mit den folgenden möglichen Schlüsseln formatiert werden:

- **class:** optionale Klasse, die dynamische Startup-Konfigurationen und Formatwerte implementiert.
- **startup:** Den Start-Ordner, auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.
- **type:** kann `gallery` oder `category` sein. Auf `dynamic` setzen, wenn Sie die im Parameter `class` gesetzte Klasse verwenden wollen.

```
{type: 'gallery', startup: '/demo_en/link/'}
```

Beispiel der XSD-Deklaration:

```
<xsd:sequence>
    <xsd:element name="LinkGallery" type="OpenCmsVfsFile" minOccurs="1"/>
    […]
</xsd:sequence>

[…]
<layouts>
    <layout element="LinkGallery" widget="LinkGalleryWidget"
          configuration="{type: 'gallery',
          startup: '/demo_en/link/'}"/>
</layouts>
```

4.2 Select-Widget-Konfiguration

Wenn Optionen von XML-Content-Schema-Definitionen als Widget-Konfiguration-Optionen übergeben werden, dann wird die folgende Syntax verwendet, um die Options-Werte zu definieren:

```
value='{text}' default='{true|false}' option='{text}'
```

```
help='{text}|{more option definitions}
```

Zum Beispiel:

```
value='value1' default='true' option='option1' help='help1'|value='value2'
option='option2' help='help2'
```

Die Elemente `default`, `option` und `help` sind alle optional. Nur der Parameter `value` muss angegeben werden. Es sollte nur für eine Option ein Defaultwert auf ‚true' gesetzt sein. Wenn mehr als einer erkannt wird, wird nur der erste gefundene Defaultwert verwendet. Wenn der Parameter `option` nicht gesetzt ist, wird der Parameter `option` auf den Wert des Parameters `value` gesetzt. Der Standardwert für den Parameter `help` ist null.

Abkürzungssyntax-Optionen:

Wenn Sie den Parameterschlüssel `value` nicht angeben, dann wird angenommen, dass der Wert an der ersten Stelle einer Options-Definition steht. In diesem Fall muss der Wert nicht durch die Zeichen `'` eingeschlossen werden. Beispiel: `value='some value' default='true'` kann ebenso geschrieben werden als: `default='true'`.

Nur wenn Sie die kurze Wertdefinition, wie oben beschrieben, verwenden, kann ein Standard-Wert mit * am Ende der Wertdefinition gekennzeichnet werden. Beispiel:

`value='some value' default='true'` kann ebenso geschrieben werden als `some value*`.

Nur wenn Sie die kurze Wertdefinition, wie oben beschrieben, verwenden, können Sie auch an den Wert ein „:“ anhängen. In diesem Fall muss kein „`'`“ die Option umgeben. Beachten Sie bitte, dass in diesem Fall der Wert selber kein „:“-Zeichen enthalten kann, weil es sonst als ein Begrenzungszeichen interpretiert werden würde. Beispiel: `value='some value' option='some option'` kann ebenso geschrieben werden als `some value:some option`.

Jede Kombination der oben beschriebenen Abkürzungen ist im Konfigurations-String erlaubt. Hier noch weitere Beispiele von gültigen Konfigurations-Strings:

```
1*|2|3|4|5|6|7
1 default='true'|2|3|4|5|6|7
value='1' default='true'|value='2'|value='3'
value='1'|2*|value='3'
1*:option text|2|3|4
1* option='option text' help='some'|2|3|4
```

Bitte beachten Sie: Wenn ein Eintrag im Konfigurations-String fehlerhaft ist, wird dieser Fehler stillschweigend ignoriert (aber trotzdem im Log-Channel dieser Klasse ins INFO-Level geschrieben).

4.3 Implementierung von Custom-Widgets

Wenn die zur Verfügung gestellten Widgets nicht Ihre Anforderungen erfüllen, ist es möglich, kundenspezifische Widgets für den Inhaltseditor zu implementieren. Das Modul `'org.opencms.dev.demo.customwidget'`, das mit dem Standard OpenCms ausgeliefert wird, zeigt wie kundenspezifische Widgets für den GWT-gestützten Inhaltseditor implementiert werden, ohne die Notwendigkeit auch nur eine Zeile von GWT-Code zu schreiben bzw. zu kompilieren.

Die serverseitige Implementierung ist abgelegt unter:

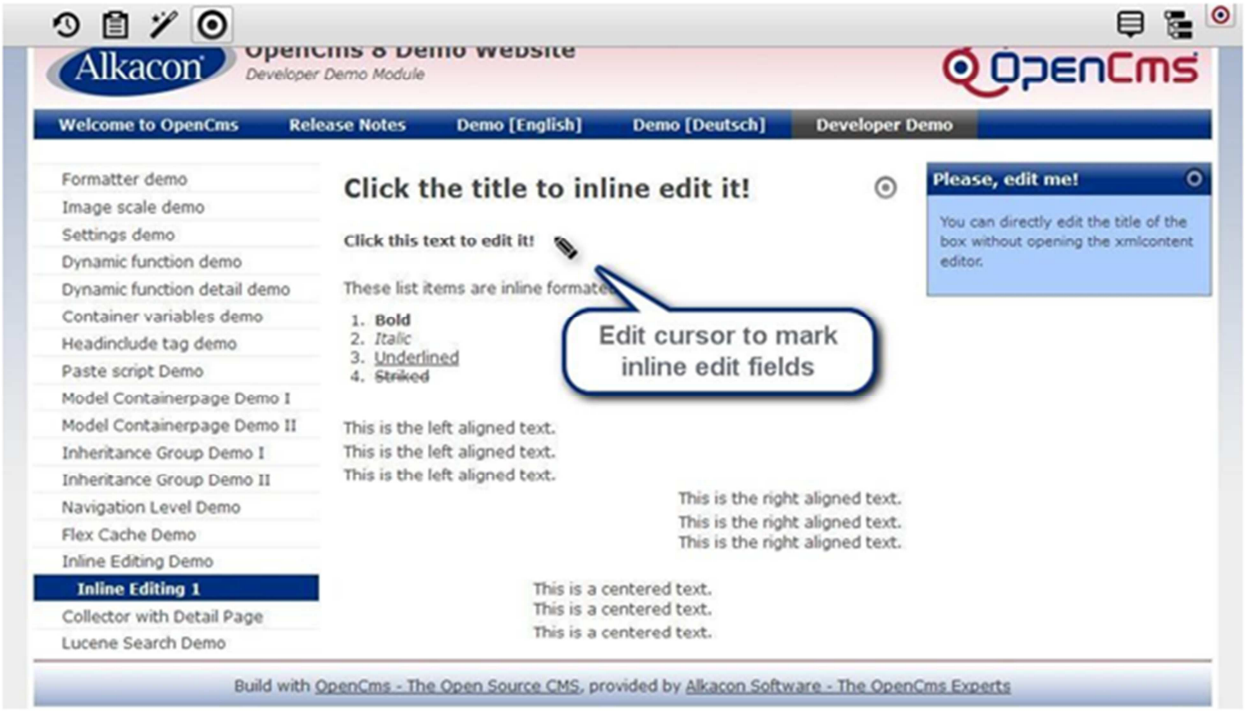
`org.opencms.dev.demo.customwidget.CustomWidget.`

Und die clientseitige Implementierung finden Sie im VFS unter:

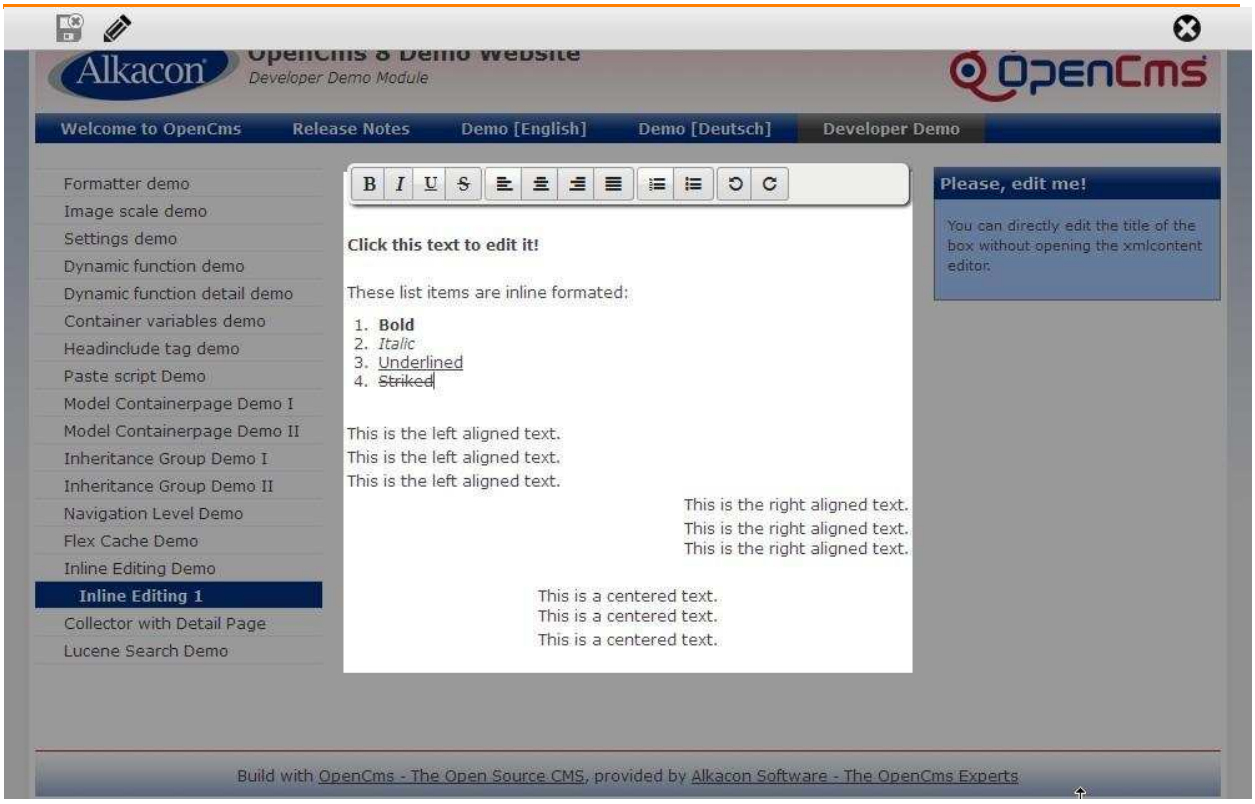
`/system/modules/org.opencms.dev.demo.customwidget/resources/mywidget.js`

5 Inline-Editor

Der Inline-Editor ist ein neues Feature, das mit OpenCms 8.5 eingeführt wurde. Er verbessert die User-Experience während der Bearbeitung von Seiteninhalten. Das Inline-Editing ermöglicht dem Redakteur, Inhalte direkt auf der Vorschau-Seite zu ändern, ohne Inhaltselemente zu verschieben. Der formbasierte Inhaltseditor kann wie gewohnt für die Bearbeitung des kompletten Inhalts der Ressource verwendet werden. Bewegt man den Cursor über einen Text, der im Inline-Editor bearbeitet werden kann, erscheint ein Icon = Stift = zur Bearbeitung:



Durch Klicken auf das Inline-editierbare Feld wird der Text markiert und eine Format-Symbolleiste erscheint (Format-Symbolleiste ist optional):



- Klicken Sie hier, um die Inline-Editor-Demo zu öffnen

5.1 Felder, die das Inline-Editing unterstützen

Inline editieren kann für Inhaltsfelder des Typs `OpenCmsString` und `HtmlWidget` verwendet werden. Wir beabsichtigen, diese Liste für die zukünftigen Versionen zu erweitern.

5.2 Konfiguration

Inline-Editing wird in einer Formatter-JSP konfiguriert. Jeder Formatter eines Ressourcentyps legt sein eigenes Inline-Editierungs-Verhalten fest. Z. B. kann ein Artikel sowohl auf die mittlere Spalte als auch auf die Seitenspalte gezogen werden, aber der Editor kann nur den Artikel in der mittleren Spalte bearbeiten. Um dies zu erreichen sollte der Inline-Editor nur im Formatter der mittleren Spalte aktiviert werden.

5.2.1 Formatters

Sie haben ein XML-Content mit einem Feld `Title`. Die Inline-Editing-Funktion für dieses XML-Content-Feld kann in der Formatter-JSP in nur zwei Schritten konfiguriert werden:

1. Damit das Inline-Editing zur Verfügung steht, fügen Sie dem `<cms:formatter>` -Tag das Attribut `rdfa="rdfa"` hinzu

```
<cms:formatter var="content" val="value" rdfa="rdfa">
```

2. Fügen Sie `${rdfa.fieldName}` dem HTML-Element als Attribut hinzu, welches den Wert umgibt.

```
<span ${rdfa.Text}>${value.Text}</span>
```

5.2.2 Verschachtelte Inhalte

Mit EL kann auf verschachtelte Inhaltsfelder zugegriffen werden. Dazu folgendes Beispiel:

```
<cms:formatter var="content" val="value" rdfa="rdfa">
   <c:forEach items="${content.valueList.Paragraph}" var="paragraph">
       <h2>${paragraph.value.Headline}</h2>
       <span>${paragraph.value.Text}</span>
   </c:forEach>
</cms:formatter>
```

`${content.valueList.Paragraph}` liefert eine Liste von `CmsJspContentAccessValueWrapper`-Objekten.

Über `${paragraph.value.Headline}` und `${paragraph.value.Text}` erhält man Zugriff auf die Werte der verschachtelten Felder. Ersetzen Sie einfach `.value` durch `.rdfa` um die für das Inline-Editing erforderlichen CSS-Attribute zu erhalten. Um das Inline-Editing zu aktivieren fügen Sie dem umgebenden HTML-Tag `${paragraph.rdfa.Headline}` und `${paragraph.rdfa.Text}` hinzu:

```
<cms:formatter var="content" val="value" rdfa="rdfa">
   <c:forEach items="${content.valueList.Paragraph}" var="paragraph">
       <h2 ${paragraph.rdfa.Headline}>${paragraph.value.Headline}</h2>
       <span ${paragraph.rdfa.Text}>${paragraph.value.Text}</span>
   </c:forEach>
</cms:formatter>
```

5.2.3 Details

Das `rdfa`-Attribut des <cms:formatter>-Tags wird in ähnlicher Weise wie das `val`-Attribut verwendet. Angenommen, der Inhalt hat ein Feld mit dem Namen `Title`. In EL erhalten Sie mit `${rdfa.Title}` die spezifischen CSS-Attribute, die erforderlich sind, um das Inline-Editing für diese Feldinhalte zu ermöglichen. Diese automatisch generierten Attribute müssen als Attribut im HTML-Element, das den Inhalt des Feldes umgibt eingestellt werden. Beachten Sie, dass die Inline-Editing-Funktion nicht auf Feldinhalte angewendet werden kann, die in einer JSP durch die Methoden `cms:stripHtml()` oder `cms:trimToSite()`manipuliert wurden.

6 Elementgruppe

Eine **Elementgruppe** ist ein neues Inhaltselement in OpenCms 8, welches auf eine Gruppe von anderen Inhaltselementen verweist.

- Klicken Sie hier, um die Elementgruppe-Demo zu öffnen

6.1 Beschreibung

Eine **Elementgruppe** ist ein Inhaltselement, ähnlich wie die Vererbungsgruppe, die der Benutzer auf andere Elemente mit der Drag & Drop Funktion ziehen kann. Der Benutzer kann die Elemente innerhalb der Gruppe bearbeiten, löschen oder ersetzen sowie ihre Reihenfolge ändern. Alle Änderungen an der Elementgruppe betreffen alle Containerseiten, zu denen die Elementgruppe gehört. Die Elementgruppe ermöglicht die Aufrechterhaltung der referenzierten Elemente an einem Ort, mit dem Effekt diese auf viele Seiten zu übernehmen.

6.2 Kombination mit Vorlagenseite

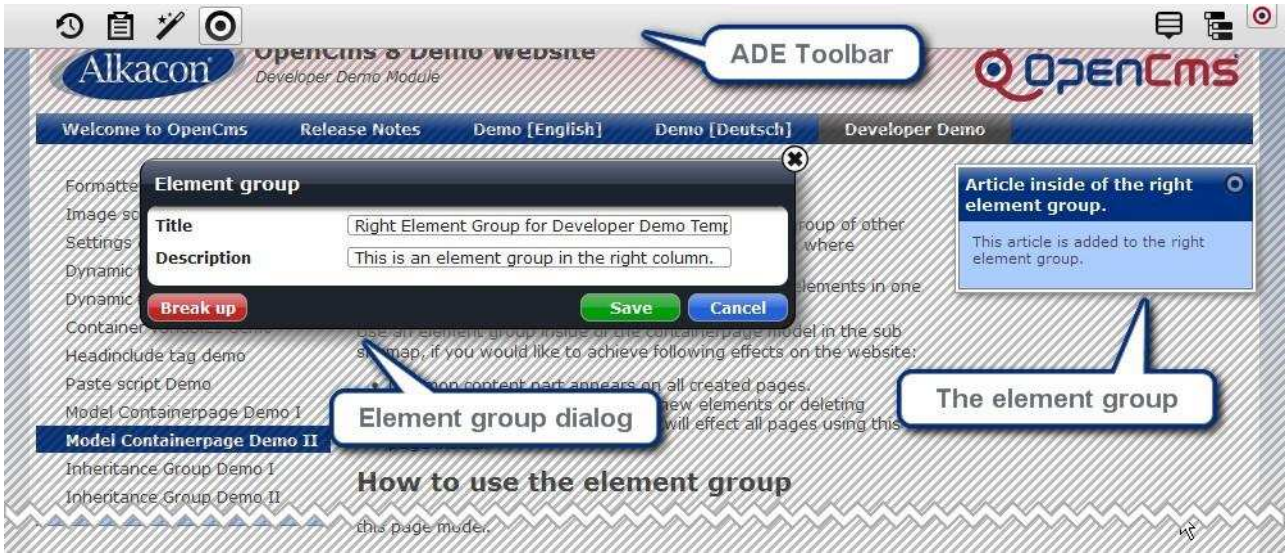
Der beliebteste Anwendungsfall für Elementgruppen ist die Kombination mit einer Modellseite. Ziehen Sie mit Drag & Drop die Elementgruppe auf den entsprechenden Container der

Modellseite und organisieren Sie alle Elemente innerhalb der Elementgruppe, die Sie möchten, um folgende Auswirkungen auf die Website haben:

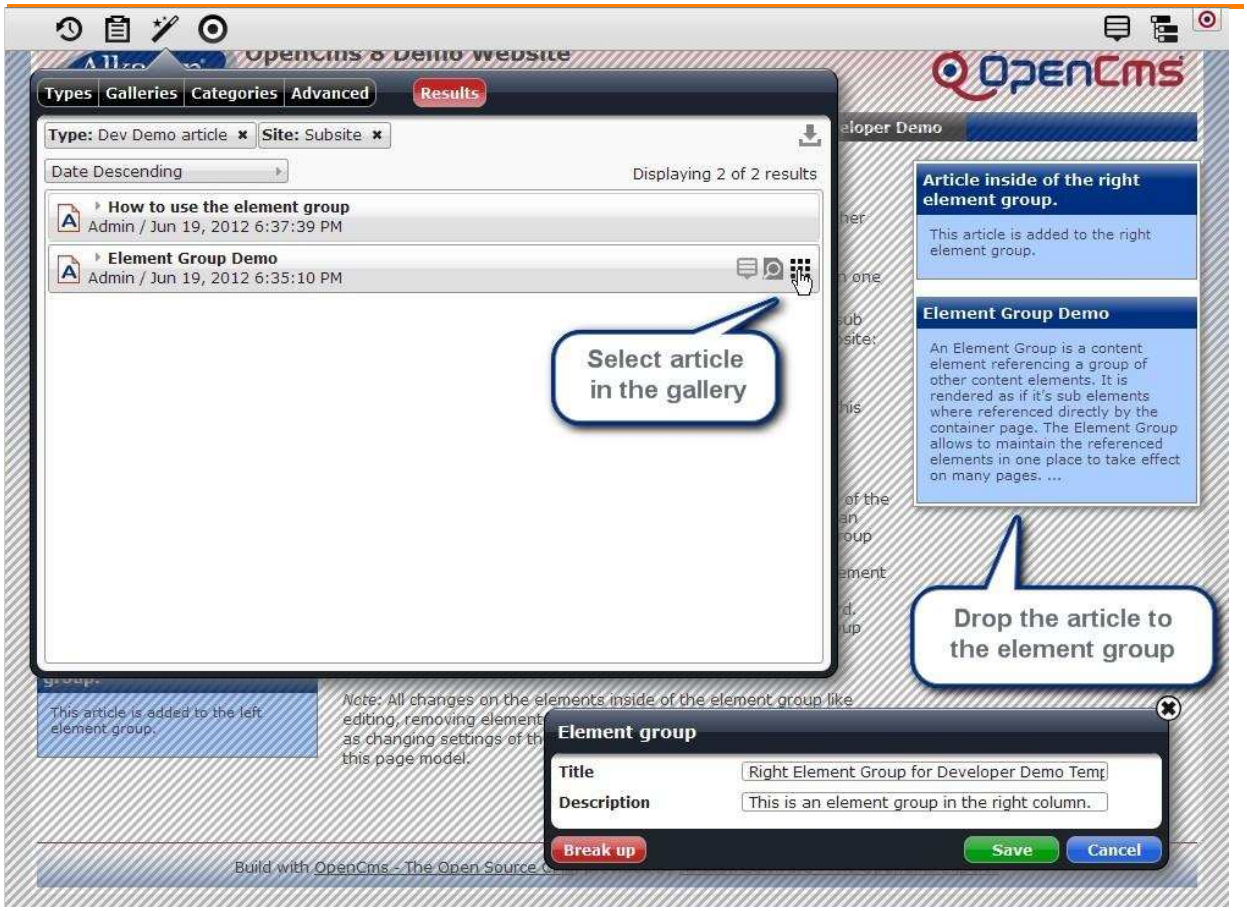
- Gemeinsame Inhaltselemente erscheinen auf allen erstellten Seiten, z. B. Header, Footer, Randspalten.
- Alle nachträglichen Änderungen in diesem Container wirken sich auf alle Seiten aus, die diese Modellseite nutzen – zum Beispiel:
    - Bearbeitung
    - Neue Elemente auf die Seite ziehen
    - Löschen von Elementen
    - Änderung der Elementreihenfolge

6.3 Gebrauch der Elementgruppe

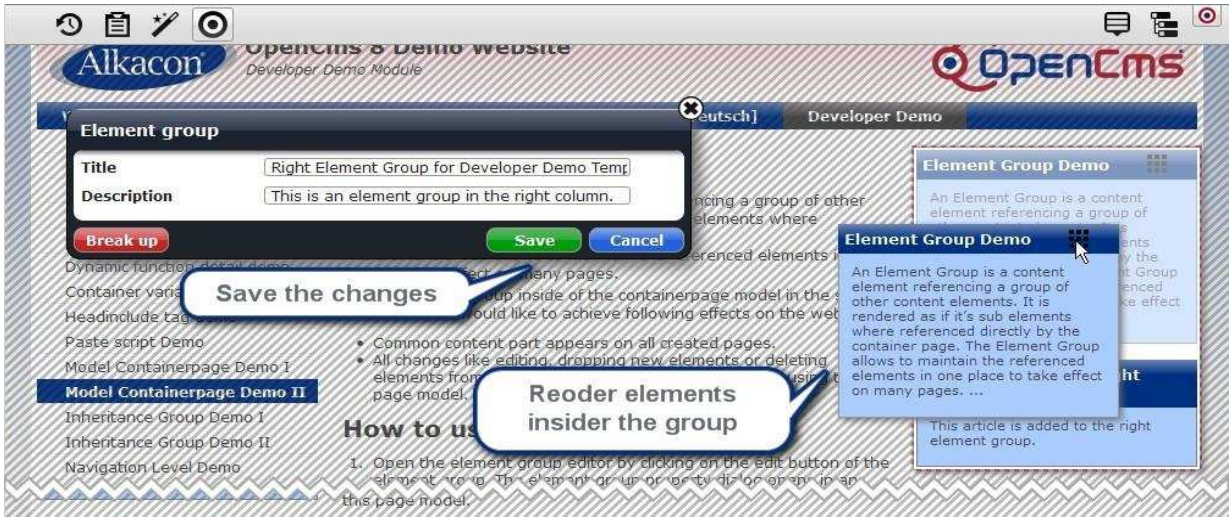
- Ziehen Sie per Drag & Drop eine Elementgruppe auf die Containerseite wie jeden anderen Inhalt aus dem "Inhalte hinzufügen"-Dialog.
- Öffnen Sie den Elementgruppen-Editor, indem Sie auf das Bearbeiten-Symbol klicken. Die Elementgruppe öffnet sich in einem Overlay, wo der Titel und die Beschreibung der Elementgruppe bearbeitet werden kann.
- Während die Elementgruppe bearbeitet wird, ist die ADE-Symbolleiste aktiv und kann wie gewohnt verwendet werden.



- per Drag & Drop neue Elemente aus dem **Inhalt-hinzufügen-**Dialog zu der Elementgruppe ziehen



- Bearbeiten oder löschen Sie die vorhandenen Elemente aus der Elementgruppe.
- Ordnen Sie die Elemente innerhalb der Gruppe.
- Speichern Sie alle Änderungen, indem Sie auf die OK-Taste des Elementgruppen-Editors klicken.



Änderungen an den Elementen innerhalb der Elementgruppe wie das Bearbeiten, Entfernen, Bewegen von Elementen, sowie das Ändern von Einstellungen der Elemente werden auf allen Seiten, die diese Modellseite verwenden, übernommen.

7 Vererbungsgruppe

Vererbungsgruppen ist ein neuer Inhaltstyp in OpenCms 8.5 und kann in Containerseiten genutzt werden.

Klicken Sie hier, um die Vererbungsgruppendemo zu öffnen

7.1 Beschreibung

Eine Vererbungsgruppe ist einer Elementgruppe in dem Sinne ähnlich, dass es erlaubt, eine Einheit von Elementen per Drag & Drop als ein einzelnes Element auf der Seite zu platzieren. Allerdings erlauben es Elementgruppen nicht, für eine bestimmte Untermenge von Seiten eine Einheit von Elementen anzupassen. Wenn eine Änderung an einer Elementgruppe gemacht wird, ist diese Änderung auf allen Seiten sichtbar, die diese Elementgruppe verwenden. Vererbungsgruppen sind dafür gedacht, um genau dieses Problem zu lösen.

Wenn Sie die Vererbungsgruppe einer Seite ändern, wird der Inhalt dieser Gruppe nur auf dieser Seite und auf den Seiten geändert, die Kindelemente vom Elternordner der aktuellen Seite sind, mit anderen Worten: Änderungen werden durch die Gruppe an alle Kinderseiten vererbt – sonst nirgendwo hin. So können Sie beispielsweise eine Vererbungsgruppe mit gemeinschaftlichem Inhalt für die rechte Spalte Ihres Templates definieren, welcher überall auf Ihrer Website sichtbar ist und dann aber jedem Unterbereich weitere spezifische Inhalte hinzufügen.

7.2 Basisinformationen

Nennen wir eine Containerseite a.html und eine **Kindseite,** einer anderen Containerseite b.html, dann ist der Elternordner von a.html ein direkter oder indirekter Unterordner des Elternordners von b.html ist. Andersherum nennen wir b.html eine **Elternseite** von a.html. Wenn wir eine Containerseite bearbeiten, nennen wir den Elternordner dieser Seite den *aktuellen Ordner*. Das ist wichtig, weil Vererbungsgruppendaten Verzeichnisse und nicht einzelnen Containerseiten zugewiesen sind. Darum können Containerseiten im selben Verzeichnis auch keine unterschiedlichen Inhalte für dieselbe Vererbungsgruppe haben.

7.3 Verwendung

7.3.1 Erstellung von Vererbungsgruppen

Durch Öffnen des Galeriemenüs und dem Ziehen eines „Vererbungsgruppen-Elements“ aus dem Typen-Tab auf Ihre Seite, können Sie eine neue Vererbungsgruppe erstellen. Genau wie bei anderen Ressourcen wird die Vererbungsgruppe erst im VFS erstellt, wenn Sie diese bearbeiten.

7.3.2 Verwendung vorhandener Vererbungsgruppen

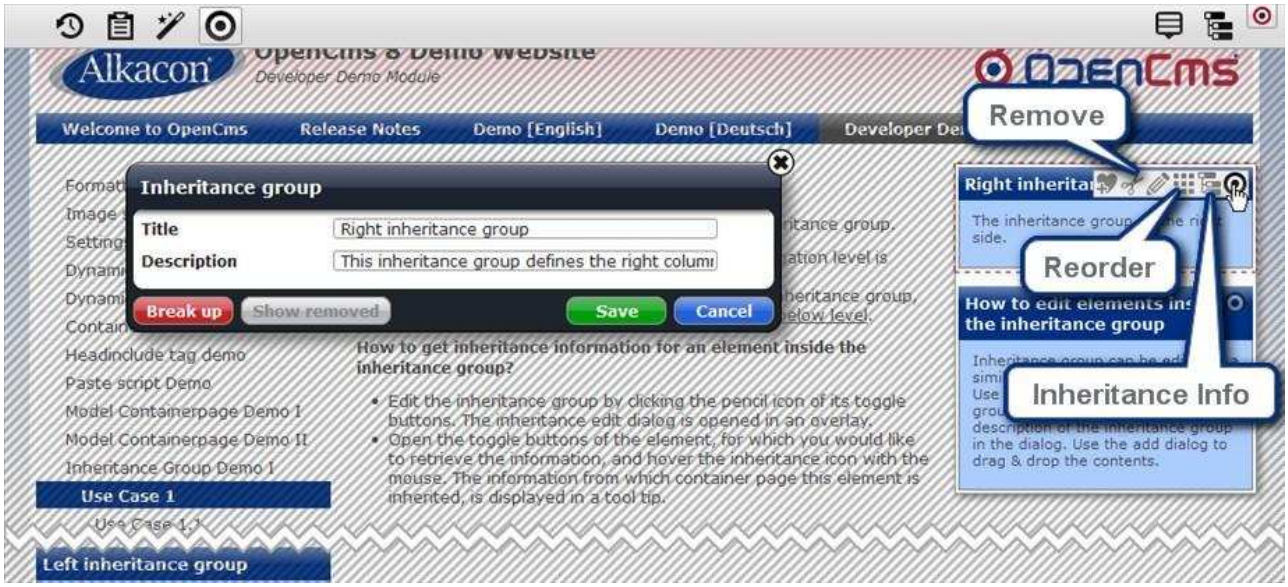
Bereits existierende Vererbungsgruppen können Sie auf Ihrer Seite verwenden, in dem Sie das Galeriemenü öffnen, den Typ Vererbungsgruppe aus dem Typen-Tab auswählen (Lupen-Symbol) und ein existierendes Vererbungsgruppen-Element auf Ihre Seite ziehen.

Ist eine Vererbungsgruppe erst mal erstellt, kann man sie überall auf Containerseiten einfügen.

7.3.3 Änderung von Vererbungsgruppen

Um eine Vererbungsgruppe zu bearbeiten benötigen Sie zunächst eine Containerseite, die eine Vererbungsgruppe enthält. Fahren Sie mit der Maus über den **Editierpunkt** der

Vererbungsgruppe und klicken Sie auf das Bearbeitungssymbol. Dadurch erscheint der Vererbungsgruppen-Editor. Sie können jetzt verschiedene Aktionen durchführen:

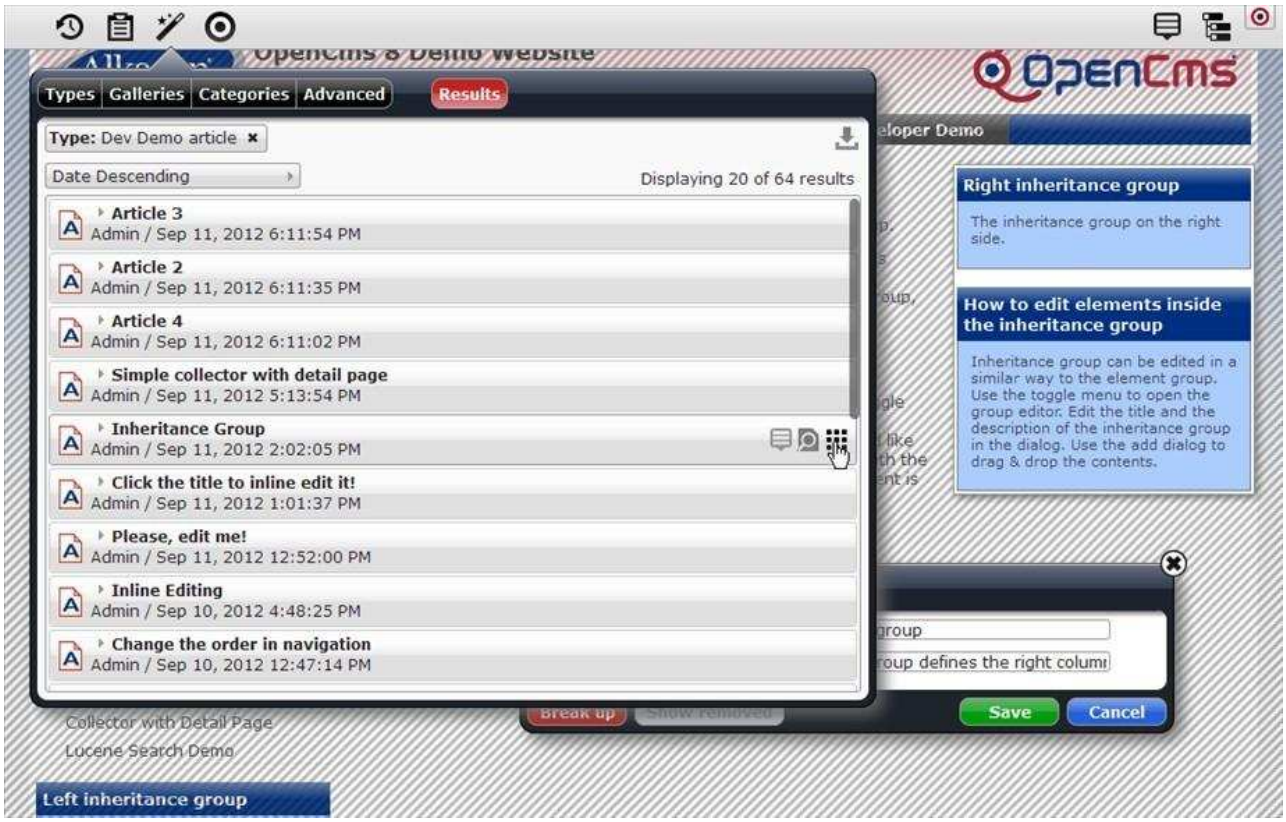


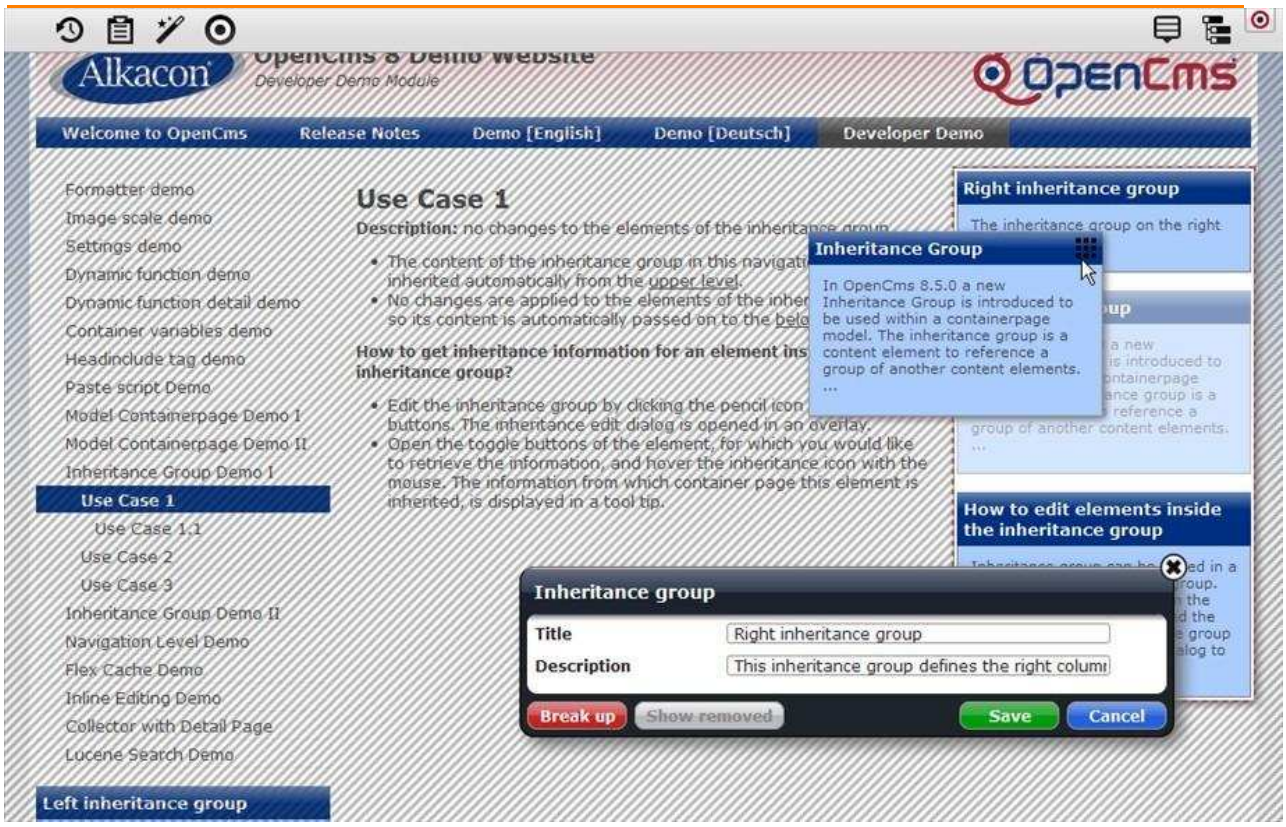
7.3.3.1 Titel und Beschreibung ändern

In den beiden Textfeldern können Sie den Titel und die Beschreibung der Vererbungsgruppe ändern. Der Titel und die Beschreibung werden nicht angezeigt, wenn die Vererbungsgruppe gerendert wird. Sie werden hauptsächlich genutzt, um vorhandene Vererbungsgruppen im Galeriedialog zu finden.

7.3.3.2 Inhaltselemente hinzufügen

Sie können der Vererbungsgruppe neue Inhaltselemente hinzufügen, in dem Sie den **Inhalte hinzufügen-Dialog** öffnen und von dort ein Inhaltselement in die Vererbungsgruppe ziehen.





Beachten Sie, dass beim Hinzufügen neuer Elemente zu den Vererbungsgruppen einer Seite, diese Elemente von der gleichen Vererbungsgruppe auf jeder nachkommenden Seite vererbt werden. Standardmäßig werden neue Elemente am unteren Teil der Vererbungsgruppe, der nachkommenden Seiten erstellt.

7.3.3.3 Inhaltselemente entfernen

Durch den Klick auf das **Entfernen-Symbol** im Vererbungsgruppen-Editor wird, abhängig davon wo das Inhaltselement der Vererbungsgruppe hinzugefügt wurde, eine von zwei Aktionen durchgeführt.

Wenn der Vererbungsgruppe das Inhaltselement in der gerade bearbeiteten Seite oder einer Seite im selben Verzeichnis hinzugefügt wurde, wird das Element einfach entfernt und es taucht in der Vererbungsgruppe keiner Kindseite auf. Wenn das Inhaltselement aber einer Vererbungsgruppe in einem Elternverzeichnis hinzugefügt wurde, wird das Element auf der aktuellen Seite und auf allen Unterseiten versteckt. Der Unterschied ist, dass ein verstecktes Element in einer Unterseite wieder auf „sichtbar“ gesetzt werden kann.

7.3.3.4 Umsortierung von Inhaltselementen

Vererbungsgruppen können per Drag & Drop umsortiert werden. Ziehen Sie die Elemente einfach durch mit Hilfe des **Verschieben-Symbols** innerhalb der Vererbungsgruppe an eine andere Stelle.

7.3.3.5 Vererbungsstatus

Wurde eine Vererbungsgruppe nicht dem aktuellen Verzeichnis hinzugefügt, erscheint im Menü des Elements ein baumähnliches “Vererbungsinformationssymbol“. Fährt man mit der Maus über das Symbol, wird angezeigt in welchem Verzeichnis dieses Element der Vererbungsgruppe hinzugefügt wurde.

7.3.3.6 Zuvor versteckte Elemente anzeigen

Der “Entfernte anzeigen/ausblenden”-Button des Vererbungsgruppen-Editors blendet die Elemente, die in der Vererbungsgruppe eines Elternordners versteckt wurden, ein bzw. aus. Um ein verstecktes Element auf der aktuellen Seite anzuzeigen, klicken Sie zunächst auf den „Entfernte anzeigen“-Button. Um die versteckten Elemente von den normalen Elementen zu unterscheiden, werden diese durch diagonale Streifen auf den versteckten Elementen markiert. Fahren Sie mit der Maus über den **Editierpunkt** des Elements, welches Sie sichtbar machen möchten und klicken dann auf das „Inhalte hinzufügen“-Symbol (+) des erscheinenden Elementmenüs.

7.3.3.7 Speichern

Die Änderungen an einer Vererbungsgruppe werden nur durchgeführt, wenn Sie den “Speichern”-Button klicken. Dadurch wird der Vererbungsgruppen-Editor geschlossen.

7.3.3.8 Auflösen

Durch Verwendung dieses Buttons wird die Vererbungsgruppe aufgelöst. Sie wird ersetzt durch die aktuell vorhandenen, individuellen Inhaltselemente der Containerseite – genauso wie bei Gruppencontainern. Bitte beachten Sie, dass diese Aktion sofort, ohne Klicken des “Speichern-Buttons“, ausgeführt wird.

7.3.3.9 Elementeinstellungen bearbeiten

Elementeinstellungen von Vererbungsgruppen können genauso bearbeitet werden, wie die Elemente von Containerseiten. Beachten Sie aber, dass die Elementeinstellungen eines Elements nur geändert werden können, wenn das Element der Vererbungsgruppe im aktuellen Verzeichnis hinzugefügt wurde. Wurde das Element der Vererbungsgruppe in einem anderen Verzeichnis hinzugefügt, erscheint die Bearbeitungsfunktion für Elementeinstellungen nicht.

7.4 Internas

Für die Implementierung von Vererbungsgruppen werden zwei Ressourcentypen verwendet:

`inheritance_group` und `inheritance_config`. `inheritance_group` ist der Ressourcentyp für Inhalte, die tatsächlich in die Containerseite eingefügt werden, wenn eine neue Vererbungsgruppe erstellt oder eine vorhandene Vererbungsgruppe über den Galeriedialog in eine Seite eingefügt werden. Sie enthalten nur die interne ID der Vererbungsgruppe, in der sie verwendet werden.

Ressourcen vom Typ `inheritance_config` enthalten die tatsächlichen Inhalte der Vererbungsgruppen. Die Änderungen einer Vererbungsgruppe in einem Verzeichnis werden in einer Datei (vom Typ `inheritance_config`) im selben Verzeichnis mit dem Namen `.inherited` gespeichert. Da es nur eine Datei pro Verzeichnis gibt, werden mehrere Vererbungsgruppen in derselben Datei gespeichert. Der Inhalt dieser Konfigurationsdatei sollte nicht manuell bearbeitet werden.

8 Kollektoren

Klick hier, um eine einfache Kollektor-Demo zu öffnen

8.1 Implementierung

Um einen Kollektor zu entwickeln, der den `<cms:collector>`-Tag verwendet, sollte der neue Kollektor das Interface `I_CmsResourceCollector` implementieren. Das Paket `org.opencms.file.collectors` bietet mit der Klasse `A_CmsResourceCollector` bereits eine Standard-Implementierung dieses Interfaces. Erweitern Sie diese Klasse, wenn Sie Ihre eigenen Kollektoren entwickeln. Folgende Methoden müssen implementiert werden:

```
List<String> getCollectorNames();
```

`getCollectorNames()`gibt einen oder mehrere Kollektoren als eine Liste von Strings zurück.

```
String getCreateLink(CmsObject cms, String collectorName, String param)
    throws CmsException, CmsDataAccessException;
```

`getCreateLink(CmsObject, String, String)` gibt den Link zurück, der ausgeführt werden muss, wenn ein Benutzer den **Direct Edit Neu-Button** in einer Liste klickt, die durch den Kollektor erstellt wurde. Wenn diese Methode `null` zurückgibt, bedeutet das, dass die gewählte Kollektorimplementierung keine **Erstelle Link-Funktion** unterstützt und somit der **Neu-Button** in der mit diesem Kollektor generierten Liste nicht erscheint.

```
String getCreateParam(CmsObject cms, String collectorName, String param)
    throws CmsDataAccessException;
```

Die Methode `getCreateParam(CmsObject, String, String)`gibt die Parameter zurück, welche an `getCreateLink(CmsObject, String, String)` übergeben werden müssen. Wenn diese Methode `null` zurückgibt, bedeutet das, dass die gewählte Kollektorimplementierung keine **Erstelle Link-Funktion** unterstützt und somit der **Neu-Button** in der mit diesem Kollektor generierten Liste nicht erscheint.

```
List<CmsResource> getResults(CmsObject cms, String collectorName, String param)
    throws CmsDataAccessException, CmsException;
```

`getResults(CmsObject, String, String)` gibt eine Liste von `org.opencms.file.CmsResource`-Objekten zurück, welche im VFS durch den angegebenen Kollektor gesammelt wurden.

8.2 Konfiguration

Editiere `opencms-vfs.xml` und füge dem `<collectors>`-Knoten folgende Zeile hinzu und starten Sie anschließend den Servlet-Container neu.

```
<collector class="org.opencms.dev.demo.CmsSimpleResourceCollector" order="180" />
```

8.3 Anwenden von Kollektoren in JSP-Dateien

Verwenden Sie den <cms:contentload>-Tag, um die Ressourcen zu sammeln.

```
<cms:contentload collector="..." param="..." editable="true">
   <cms:contentaccess var="content" /> ...
</cms:contentload>
```

Dieser Tag benötigt folgende Parameter:

collector – den Namen des Kollektors.

param – Kollektor-Parameter. Die Standard-Parameter Syntax ist: `[path]|[resource type]|[count]`

editable – Setzen Sie dieses Attribut auf `true`, um das direkte Editieren für diese Liste zu aktivieren.

Mit `<cms:contentaccess var="content"/>` erhalten Sie Zugriff zum `CmsXmlContent` des aktuellen Elements in der Iteration.

```
<%@page buffer="none" session="false" taglibs="c,cms" %>
<%-- The JSP HTML should be surrounded by block element --%>
<div>
  <%-- Read collector parameter, e.g. from request --%>
  <c:set var="folder" value="${param.folder}"/>
  <c:set var="type" value="${param.type}"/>
  <c:set var="count" value="${param.count}"/>
    <ul>
    <%-- Use <cms:contentload> with new collector--%>
    <cms:contentload collector="myCollector" param="${folder}|${type}|${count}">
      <%-- Access the content --%>
      <cms:contentaccess var="content" />
      <c:set var="link"><cms:link>${content.filename}</cms:link></c:set>
      <li><a href="${link}">${content.value.Title}</a></li>
    </cms:contentload>
  </ul>
</div>
```

8.4 Erstellen von Listen mit Drag & Drop Funktion

Ein üblicher Weg, um eine JSP auf eine Containerseite zu ziehen ist es, die dynamische Funktion zu nutzen.

- Erstellen Sie eine neue dynamische Funktion. Normalerweise ist dieser Ressourcentyp ein Teil des Moduls und sollte im Ordner `/system/modules/mymodule/functions/`, erstellt werden. Die Kollektor-JSP des Dev-Demo-Moduls ist im Ordner: `/system/modules/org.opencms.dev.demo/functions/`
- Wählen Sie die JSP, die **myCollector** enthält. In der Dev-Demo ist das die folgende JSP: `/system/modules/org.opencms.dev.demo/pages/collector.jsp`
- Um die Kollektor-Parameter zu definieren, setzen Sie die Anfangswerte des Request-Parameters in der dynamischen Funktion. Diese Parameter werden in der `collector.jsp` verwendet:
    - folder=`/dev-demo/collector-with-detail-page/.content/article/`
    - type = ddarticle
    - count = 5

Jetzt können Sie eine Containerseite im ADE öffnen. Ziehen Sie die neue dynamische Funktion zur Ansicht in den Container.

8.5 Verwendung von Kollektoren in Detailseiten

- Öffnen Sie im Sitemap-Editor den **Seite erstellen**- Dialog
- ziehen Sie eine Seite des Ressourcentyps aus dem **Typen-Tab,** für den der Kollektor konfiguriert ist.
- OpenCms konfiguriert die neue Detailseite für die Sub-Sitemap automatisch.

Wenn Sie nun auf den Link klicken, der die Ressource öffnet, wird die Detailseite der Ressource geöffnet. In der Liste wird die Detailseiten-URL verwendet.

8.6 Code-Beispiel

Beispiel einer einfachen Kollektor-Klasse und JSP:
CmsSimpleResourceCollector.java

simple-collector.jsp

9 XSD Choice-Element

Das **XSD Choice-Element** erweitert die XML-Schema-Definition von OpenCms. Das **XSD Choice-Element** bietet eine oder mehrere Auswahlmöglichkeiten von OpenCms-Typen in beliebiger Reihenfolge. Dieser Teil der Dokumentation beschreibt, wie das Choice-Element innerhalb XSD verwendet wird und wie auf die Werte in einer JSP mit EL zugreifen kann.

9.1 Definition

Um einen OpenCms-Typen zu beschreiben, kann der `<xsd:choice>`-Knoten in der gleichen Weise wie ein `<xsd:sequence>`-Knoten verwendet werden. Es muss in der XML-Schema-Definition eines *verschachtelten XML-Inhaltes* definiert werden. Das Element im Root-Schema muss *optional* sein. Fügen Sie das Attribut `minOccurs="0"` hinzu, um es optional zu machen.

Root XML Schema Definition:

```
<%@page buffer="none" session="false" taglibs="c,cms" %>
<%-- The JSP HTML should be surround by block element --%>
<div>
      <%-- Read collector parameter, e.g. from request --%>
        <c:set var="folder" value="${param.folder}"/>
        <c:set var="type" value="${param.type}"/>
        <c:set var="count" value="${param.count}"/>
        <ul>
      <%-- Use <cms:contentload> with new collector--%>
        <cms:contentload collector="myCollector"
                         param="${folder}|${type}|${count}" editable="true">
              <%-- Access the content --%>
                <cms:contentaccess var="content" />
              <%-- Set the link to the content in the list and
                   do not forget to use <cms:link> tag --%>
                <li>
                  <c:set var="link"><cms:link>${content.filename}</cms:link></c:set>
                  <a href="${link}">
                      ${content.value.Title}
                  </a>
              </li>
        </cms:contentload>
        </ul>
</div>
```

Verwenden Sie den <xsd:choice>-Knoten, um das Auswahlelement zu definieren. Die Elemente, die die Auswahlmöglichkeiten der <xsd:choice> definieren, müssen auch *optional* sein.

Verschachtelte XML-Schema-Definition mit <xsd:choice>:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified">
       <xsd:include schemaLocation="opencms://opencms-xmlcontent.xsd"/>
       …
       <xsd:complexType name="OpenCmsDevDemoTextOption">
              <xsd:choice minOccurs="0" maxOccurs="3">
               <xsd:element name="Text" type="OpenCmsString" minOccurs="0" />
               <xsd:element name="Html" type="OpenCmsHtml" minOccurs="0" />
               <xsd:element name="Link" type="OpenCmsVarLink" minOccurs="0" />
              </xsd:choice>
              <xsd:attribute name="language" type="OpenCmsLocale" use="optional"/>
       </xsd:complexType>
</xsd:schema>
```

9.2 Einzel- und Mehrfachauswahl

Das <xsd:choice>-Element kann als Einzel-oder Multiple-Choice-Element verwendet werden. Um die Einzelauswahl zu verwenden, benutzen Sie keine Attribute für den <xsd:choice>-Knoten. Dies ist die Standardeinstellung.
Um eine Mehrfachauswahl zu verwenden, setzen Sie folgende Attribute:

- `minOccurs="0" und`
- `maxOccurs="[max number of elements]"`, z. B. maxOccurs="5"

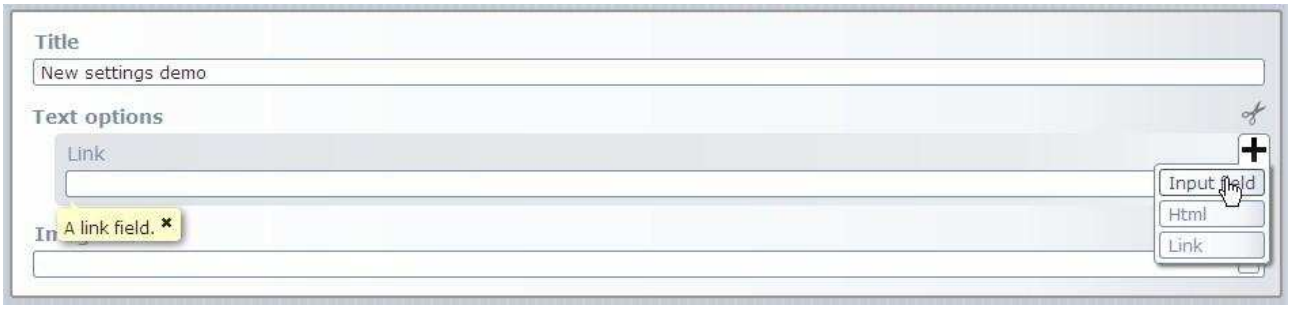
Einzelauswahl:

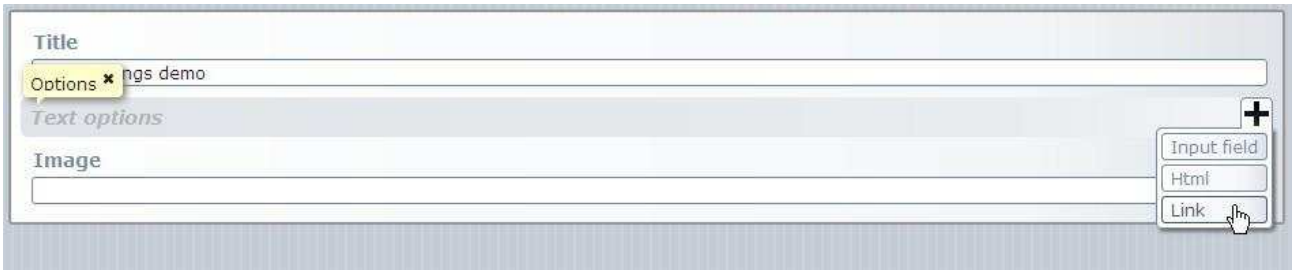
```
<xsd:choice>
   <xsd:element name="VariableLink" type="OpenCmsVarLink" minOccurs="0" />
   <xsd:element name="LinkGallery" type="OpenCmsVfsFile" minOccurs="0" />
   <xsd:element name="DownloadGallery" type="OpenCmsVfsFile" minOccurs="0" />
</xsd:choice>
```

Mehrfachauswahl:

```
<xsd:choice minOccurs="0" maxOccurs="5">
   <xsd:element name="VariableLink" type="OpenCmsVarLink" minOccurs="0" />
   <xsd:element name="LinkGallery" type="OpenCmsVfsFile" minOccurs="0" />
   <xsd:element name="DownloadGallery" type="OpenCmsVfsFile" minOccurs="0" />
</xsd:choice>
```

9.3 Inhaltseditor





9.4 Zugriff auf Werte in einer JSP

Auf Elemente eines `<xsd:choice>`-Knoten kann in gleicher Weise auf Elemente zugegriffen werden, wie bei verschachtelten Inhalten. Bevor der Inhalt gelesen wird, testen Sie das Root-Element und die Choice-Elemente, ob sie existieren. Überprüfen Sie die Existenz der verschachtelten Root-Elemente.

```
<c:if test="${value.Options.exists}"> … </c:if>
```

Überprüfen Sie den Wert des Choice-Elements. Verwenden Sie für die Prüfung `isSet`. `isSet` liefert `true` zurück, wenn das Element vorhanden ist und sein Wert kein leerer String ist.

Wir erweitern das obige Beispiel:

```
<c:if test="${value.Options.exists && value.Options.value.Text.isSet}">
   <div>${value.Options.value.Text}</div>
</c:if>
```

In der gleichen Weise greifen wir auf die anderen Choice-Elemente zu:

```
${value.Options.value.Html.exists}
${value.Options.value.Html.isSet}
${value.Options.value.Html}
${value.Options.value.Link.exists}
${value.Options.value.Link.isSet}
${value.Options.value.Link}
```

Komplettes Beispiel mit einem Auswahlelement innerhalb eines verschachtelten Inhalts (v8InfoBox):

```
<c:if test="${value.FurtherInfo.value.Link.exists
            && (value.FurtherInfo.value.Link.value.VariableLink.isSet
            || value.FurtherInfo.value.Link.value.LinkGallery.isSet
            || value.FurtherInfo.value.Link.value.DownloadGallery.isSet)}">
   <c:choose>
       <c:when test="${value.FurtherInfo.value.Link.value.VariableLink.isSet}">
```

```
            <c:set var="infolink">
                ${value.FurtherInfo.value.Link.value.VariableLink}
            </c:set>
        </c:when>
        <c:when test="${value.FurtherInfo.value.Link.value.LinkGallery.isSet}">
            <c:set var="infolink">
                ${value.FurtherInfo.value.Link.value.LinkGallery}
            </c:set>
        </c:when>
        <c:when test="${value.FurtherInfo.value.Link.value.DownloadGallery.isSet}">
            <c:set var="infolink">
                 ${value.FurtherInfo.value.Link.value.DownloadGallery}
            </c:set>
        </c:when>
    </c:choose>
    <c:set var="infotext">${infolink}</c:set> ...
    <div class="boxbody_listentry">
        <c:set var="link"><cms:link>${infolink}</cms:link></c:set>
        <a href="${link}">${infotext}</a><br/>
    </div>
</c:if>
```

`${value.FurtherInfo.value.Link.exists}` Prüfe die Existenz der Auswahlelemente.

`${value.FurtherInfo.value.Link.value.VariableLink.isSet}` prüfe den Wert eines Auswahlelements.

9.5 Beispiel

Die vollständigen Beispiele für die Nutzung der Auswahlelemente sind in der Entwickler-Demo und im Template III zu finden.

1. The root XSD of the info box (v8-modules)
2. The nested XSD containing further info (v8-modules)
3. The nested XSD containing choice element (v8-modules)
4. The formatter JSP (v8-modules)
5. The root XSD of the article with settings (dev-demo)
6. The nested XSD containing choice element (dev-demo)

10 ADE-Konfiguration

10.1 Sitemap-Konfiguration

Der zentrale Ort, an dem die Ressourcentypen die im ADE **Inhalt hinzufügen**-Dialog zur Verfügung stehen und andere globale Einstellungen konfiguriert werden, ist die `.content/.config`-Datei. Einstellungen für Sub-Sitemaps folgen dem gleichen Muster mit einer `.content/.config`-Datei in der Sub-Sitemap.

Die Sitemap-Konfigurationsdatei bietet vier Tabs:

- **Ressourcentypen:** Definiert einen Ressourcentypen Namen und Musternamen, Standard-Ordner, Formatter (optional) und vieles mehr.
- **Modellseiten:** Definiert Modellseiten, die vom Sitemap-Editor verwenden werden, um neue Containerseiten zu erstellen.
- **Eigenschaftskonfiguration** Definiert, welche Eigenschaften im **Standardeigenschaften-Tab** des ADE angezeigt werden.
- **Detailseiten:** Definiert die Detailseiten, die für diesen Ressourcentypen genutzt werden können.

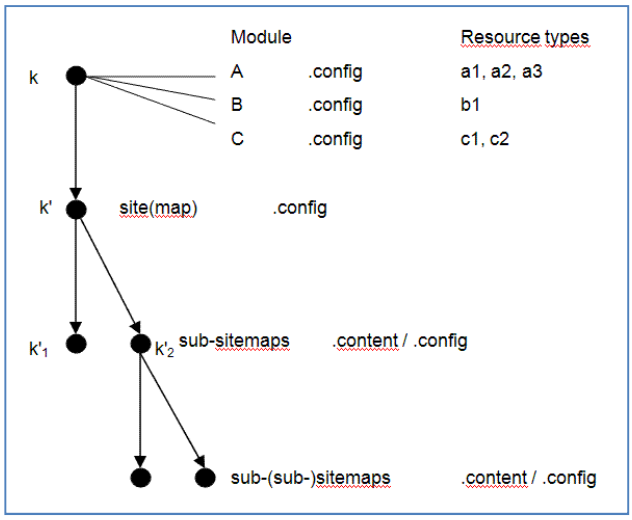
10.2 Modul-Konfiguration

Falls ein Modul einen neuen Ressourcen-Typ (wie in der Modul-Manifest definiert) in OpenCms hinzufügt und Sie möchten diesen automatisch ADE zur Verfügung stellen nachdem das Modul installiert ist, können Sie eine `.config`-Datei des Typs `module_config` im Root-Ordner Ihres Moduls platzieren.

Die Modul-Konfigurationsdatei bietet drei Optionen:

- **Ressourcentypen:** Definiert den Namen des Ressourcentypen und ein Namensmuster, Standard-Ordner, Formatter und vieles mehr.
- **Eigenschaftskonfiguration:** Definiert, welche Eigenschaften im **Standardeigenschaften-Tab** des ADE angezeigt werden.
- **Funktionen:** Definiert eine dynamische Funktion, welche mit dem Modul mitgeliefert wird Detailseiten (Es ist mindestens ein Name zur Identifikation und ein JSP-Provider angegeben werden).

10.3 Vererbung der Konfiguration



11 Integration der Solr Suche

11.1 Allgemeines

Schon viele Jahre kommt für die Suche die Apache Lucene zum Einsatz und Apache Solr wuchs und wurde mächtiger. Solr ist heute eine Enterprise-Search-Plattform, die auf Lucene basiert. Es ist ein Standalone-Server für die Enterprise-Search mit einer REST-API. Man fügt Solr Dokumente via XML, JSON oder binär über HTTP hinzu (dies wird **Indizierung** genannt). Man sucht via HTTP GET und erhält XML, JSON oder binäre Ergebnisse. Um detailliertere Informationen über Solr zu erfahren und wie sie funktioniert, besuchen Sie bitte die Website des Projekts Apache Solr. Wenn Sie mit der leistungsfähigen und flexiblen Apache Solr REST-ähnlichen Schnittstelle suchen, vereinfacht dies die Entwicklungskomplexität. Darüber hinaus können Sie sich auf bestehende grafische Schnittstellen verlassen. Diese bieten komfortable AJAX-basierte Such-Funktionalitäten für den Endanwender Ihrer Internet-/Intranet-Anwendung.

11.2 Suche nach Inhalten in OpenCms

OpenCms 8.5 integriert Apache Solr. Und nicht nur für die Volltextsuche, sondern auch als leistungsfähige Enterprise-Search-Plattform .

11.2.1 Demo

Die OpenCms v8-Module bietet eine Solr basierte OpenCms-Such-Demo mit dem Open Source UI **Ajax Solr** (Ajax Solr Project on github.com):

- Klick hier, um auf die OpenCms-Solr-Such-DEMO zu kommen

11.2.2 Schnellstart Beispiel

11.2.2.1 Senden einer REST-ähnlichen Abfrage

Angenommen Sie möchten eine Liste von "allen Artikeln, die sich seit gestern geändert haben und wo die Eigenschaft 'X' den Wert 'Y' hat“ anzeigen.

```
http://localhost:8080/opencms/opencms/handleSolrSelect?
    fq=type:v8article
```

```
    &fq=lastmodified:[NOW-1DAY TO NOW]
    &fq=Title_prop:Flower
```

Parametererklärung:

```
http://localhost:8080/opencms/opencms/handleSolrSelect
   // The URI of the OpenCms Solr Select Handler configured in 'opencms-system.xml'
    ?fq=type:v8article                  // Filter query on the field type
                                        // with the value 'v8article'
    &fq=lastmodified:[NOW-1DAY TO NOW]  // Filter query on the field lastmodified
                                        // with a range query from 'NOW-1DAY TO NOW'
    &fq=Title_prop:Flower               // Filter query on the field Title_prop
                                        // with the value 'v8article'
```

NOTE: Solr Syntax Abfrage

Wenn Sie sich mit der Solr-Abfrage-Syntax vertraut machen möchten, erhalten Sie einen allgemeinen Überblick unter Solr query Syntax. Für erweiterte Funktionen hilft Ihnen Searching - Solr Reference Guide - Lucid Imagination weiter.
Bitte beachten Sie, dass viele Zeichen in der Solr Abfragesyntax (dabei ist das Pluszeichen besonders wichtig: "+") Sonderzeichen in URLs sind, so das Sie beim manuellen Erstellen von Anforderungs-URLs darauf achten müssen, dass die Zeichen richtig URL encodiert sind.

```
                                q=  +popularity:[10   TO   *]     +section:0
http://localhost:8983/solr/select?q=%2Bpopularity:[10%20TO%20*]%20%2Bsection:0
```

Mehr Informationen finden Sie unter Yonik Seeleys Blog Nested Queries in Solr.

Sie können jede „Solr valide“ Eingabe in den neuen OpenCms Solr-Anfrage-Handler (handleSolrSelect) einfügen. Die Solr-Wiki-Seite Search and Indexing hilft Ihnen dabei, sich mit der Abfrage-Syntax vertraut zu machen

11.2.2.2 Abfragen der Antwort

Die Antwort (Response), die Solr erzeugt, kann standardmäßig XML oder JSON sein. Mit einem zusätzlichen Parameter 'wt' können Sie den QueryResponseWriter spezifizieren, den Solr verwenden soll. Hier zeigen wir für das obige Beispiel wie ein Ergebnis der Abfrage aussehen kann:

```
<response>
 <lst name="responseHeader">
   <int name="status">0</int>
   <int name="QTime">7</int>
   <lst name="params">
     <str name="qt">dismax</str>
     <str name="fl">*,score</str>
     <int name="rows">50</int>
     <str name="q">*:*</str>
     <arr name="fq">
       <str>type:v8article</str>
       <str>contentdate:[NOW-1DAY TO NOW]</str>
       <str>Title_prop:Flower</str>
     </arr>
     <long name="start">0</long>
   </lst>
 </lst>
```

```
<result name="response" numFound="2" start="0">
  <doc>
    <str name="id">51041618-77f5-11e0-be13-000c2972a6a4</str>
    <str name="contentblob">[B:[B@6c1cb5</str>
    <str name="path">/sites/default/.content/article/a_00003.html</str>
    <str name="type">v8article</str>
    <str name="suffix">.html</str>
    <date name="created">2011-05-06T15:27:13Z</date>
    <date name="lastmodified">2011-08-17T13:58:29Z</date>
    <date name="contentdate">2012-09-03T10:41:13.56Z</date>
    <date name="relased">1970-01-01T00:00:00Z</date>
    <date name="expired">292278994-08-17T07:12:55.807Z</date>
    <arr name="res_locales">
      <str>en</str>
      <str>de</str>
    </arr>
    <arr name="con_locales">
      <str>en</str>
   </arr>
    <str name="template_prop">
      /system/modules/com.alkacon.opencms.v8.template3/templates/main.jsp</str>
    <str name="style.layout_prop">/.content/style</str>
    <str name="NavText_prop">OpenCms 8 Demo</str>
    <str name="Title_prop">Flower Today</str>
    <arr name="content_en">
      <str>News from the world of flowers Flower Today In this [...]</str>
    </arr>
    <date name="timestamp">2012-09-03T10:45:47.055Z</date>
    <float name="score">1.0</float>
  </doc>
  <doc>
    <str name="id">ac56418f-77fd-11e0-be13-000c2972a6a4</str>
    <str name="contentblob">[B:[B@1d0e4a2</str>
    <str name="path">/sites/default/.content/article/a_00030.html</str>
    <str name="type">v8article</str>
    <str name="suffix">.html</str>
    <date name="created">2011-05-06T16:27:02Z</date>
    <date name="lastmodified">2011-08-17T14:03:27Z</date>
    <date name="contentdate">2012-09-03T10:41:18.155Z</date>
    <date name="relased">1970-01-01T00:00:00Z</date>
    <date name="expired">292278994-08-17T07:12:55.807Z</date>
    <arr name="res_locales">
      <str>en</str>
      <str>de</str>
    </arr>
    <arr name="con_locales">
      <str>en</str>
    </arr>
    <str name="template_prop">
      /system/modules/com.alkacon.opencms.v8.template3/templates/main.jsp
    </str>
    <str name="style.layout_prop">/.content/style</str>
    <str name="NavText_prop">OpenCms 8 Demo</str>
    <str name="Title_prop">Flower Dictionary</str>
    <arr name="content_en">
      <str>The different types of flowers Flower Dictionary There are [...]</str>
    </arr>
    <date name="timestamp">2012-09-03T10:45:49.265Z</date>
    <float name="score">1.0</float>
  </doc>
 </result>
</response>
```

11.2.2.3 Senden einer Java-API-Abfrage

```
String query="fq=type:v8article&fq=lastmodified:[NOW-1DAY TO
NOW]&fq=Title_prop:Flower";
 CmsSolrResultList results = OpenCms.getSearchManager().getIndexSolr("Solr Online
Index").search(getCmsObject(), query);
 for (CmsSearchResource sResource : results) {
   String path = searchRes.getField(I_CmsSearchField.FIELD_PATH);
   Date date =searchRes.getMultivaluedField(I_CmsSearchField.FIELD_DATE_LASTMODIFIED);
   List<String> cats = searchRes.getMultivaluedField(I_CmsSearchField.FIELD_CATEGORY);
 }
```

Die Klasse `org.opencms.search.solr.CmsSolrResultList` kapselt eine Liste von **OpenCms-Ressource-Dokumenten** (`{@link CmsSearchResource}`).

Auf diese Liste kann wie auf eine {@link ArrayList} zugegriffen werden. Die Einträge sind {@link CmsSearchResource}-Elemente die {@link CmsResource} erweitern und halten die Solr Implementierung von {@link I_CmsSearchDocument} als Member.

Dies ermöglicht Ihnen, mit der Ergebnisliste genauso umzugehen, wie Sie es mit der gut bekannten {@link List} tun und mit ihren Einträgen, wie Sie es mit {@link CmsResource} gewohnt sind.

11.2.2.4 Solr Querys mit der CmsSolrQuery-Klasse

```
CmsSolrIndex index = OpenCms.getSearchManager().getIndexSolr("Solr Online Index");
CmsSolrQuery squery = new CmsSolrQuery(getCmsObject(),
    "path:/sites/default/xmlcontent/article_0001.html");
List<CmsResource> results = index.search(getCmsObject(), squery);
```

11.2.3 Erweiterte Suchfunktionen

Automatische Vorschläge/Vervollständigung/Korrekturen:
http://wiki.apache.org/solr/Suggester

Facettierte Suche: http://wiki.apache.org/solr/SimpleFacetParameters

Highlighting: http://wiki.apache.org/solr/HighlightingParameters

Range Abfragen: http://wiki.apache.org/solr/SolrQuerySyntax

Sortierung: http://wiki.apache.org/solr/CommonQueryParameters

Rechtschreibprüfung: http://wiki.apache.org/solr/SpellCheckComponent

Thesaurus/Synonyme: http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters

11.2.3.1 Abfragen von mehreren Cores (Indizes)

Core ist das Wort in der Solr Welt für das Denken in mehreren Indizes. Um die korrekte Sprache zu verwenden, sagen wir **Core** statt **Index**. Mehrere Cores sind nur erforderlich, wenn Sie völlig unterschiedliche Anwendungen auf einem Solr Server laufen lassen, der alle Daten verwaltet. Mehr Informationen unter: Solr Core Administration. Angenommen, Sie haben mehrere Solr Cores konfiguriert und Sie möchten in einen speziellen Cores suchen, dann müssen Sie Solr/OpenCms mitteilen, in welchem Core/Index Solr suchen soll. Dies wird durch einen speziellen Parameter gemacht:

```
http://localhost:8080/opencms/opencms/handleSolrSelect?
                              // The URI of the OpenCms Solr Select Handler
                              // configured in 'opencms-system.xml'
   &core=My Solr Index Name   // Searches on the core with the name 'My Solr Index Name'
   &q=content_en:Flower       // for the text 'Flower'
```

11.2.3.2 Den Standard-OpenCms-Solr-Kollektor verwenden

OpenCms Version 8.5 liefert einen Standard-Kollektor bei Gebrauch von `byQuery` als Name, um einfach einen Abfragestring zu übergeben und `byContext` als Name um einen Abfrage-String zu übergeben und OpenCms den Request-Kontext des Benutzers nutzen zu lassen.

Die implementierte Klasse für diesen Kollektor finden Sie unter:

`org.opencms.file.collectors.CmsSolrCollector`.

```
<cms:contentload collector="byQuery" preload="true"
  param="fq=parent-folders:/sites/default/&fq=type:ddarticle&sort=lastmodified desc">
  <cms:contentinfo var="info" />
  <c:if test='${info.resultSize != 0}'>
    <cms:contentinfo var="info" />
    <c:if test='${info.resultSize != 0}'>
      <h3>Solr Collector Demo</h3>
      <cms:contentload editable="false">
        <cms:contentaccess var="content" />
        <%-- Title of the article --%>
        <h6>${content.value.Title}</h6>
        <%-- The text field of the article with image --%>
        <div class="paragraph">
          <%-- Set the required variables for the image. --%>
          <c:if test="${content.value.Image.isSet}">
            <%-- Output of the image using cms:img tag --%>
            <c:set var="imgwidth">${(cms.container.width - 20) / 3}</c:set>
            <%-- Output the image. --%>
            <cms:img src="${content.value.Image}" />
          </c:if>
          ${cms:trimToSize(cms:stripHtml(content.value.Text), 300)}
        </div>
        <div class="clear"></div>
      </cms:contentload>
    </c:if>
  </c:if>
</cms:contentload>
```

11.3 Indizierung von OpenCms-Inhalten

11.3.1 Konfiguration der Suche

Im Allgemeinen wird die Konfiguration, der systemweiten Suche für OpenCms in der Datei '`opencms-search.xml`' (`<CATALINA_HOME>/webapps/<OPENCMS_WEBAPP>/WEB_INF/config/opencms-search.xml`) vorgenommen.

11.3.1.1 Embedded/HTTP Server Solr

Seit der Version 8.5 von OpenCms gibt es einen neuen optionalen Knoten mit dem XPath: `opencms/search/solr` . Um einfach den in OpenCms eingebetteten Solr Server zu aktivieren, sollte die `opencms-search.xml` wie folgt beginnen:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE opencms SYSTEM "http://www.opencms.org/dtd/6.0/opencms-search.dtd">
<opencms>
 <search>
   <solr enabled="true"/>
     [...]
 </search>
</opencms>
```

Optional können Sie das **Solr home directory** und **den Namen der Haupt-Konfigurationsdatei von Solr** konfigurieren. OpenCms verbindet die beiden XML-Knoten zu `<solr_home><configfile>`. Ein Beispiel für eine Konfiguration würde wie folgt aussehen:

```
<solr enabled="true">
   <home>/my/solr/home/folder</home>
   <configfile>rabbit.xml</configfile>
</solr>
```

Um Solr systemweit zu deaktivieren, entfernen Sie den <solr/>-Knoten oder setzen Sie, wie unten gezeigt, das `enabled`-Attribut auf `false`:

```
<solr enabled="false"/>
```

Es ist auch möglich, sich mit einem externen HTTP-Solr-Server zu verbinden. Um dies zu tun, ersetzen Sie die Zeile `<solr enabled="true"/>` mit folgenden Inhalt:

```
<solr enabled="true" serverUrl="http://mySolrServer" />
```

Der OpenCms SolrSelect-Request-Handler unterstützt den externen HTTP-Solr-Server nicht. Wenn Ihr HTTP-Solr-Server direkt über `http://<your_server>` erreichbar ist, wird keine Berechtigungsprüfung für indizierte Daten durchgeführt. Somit sind geschützte / geheime Inhalte öffentlich zugänglich. Das bedeutet, dass Sie sich selbst um die Zugriffskontrolle von Ressourcen kümmern müssen, die im OpenCms-VFS Zugriffsbeschränkungen eingestellt haben. Sie können natürlich die Methode `org.opencms.search.solr.CmsSolrIndex.search(CmsObject, SolrQuery`, oder `org.opencms.search.solr.CmsSolrIndex.search(CmsObject, String)` verwenden und dadurch dann sicher sein, dass die VFS-Berechtigungen auch für externe HTTP-Solr-Server geprüft werden. Vielleicht wird eine zukünftige Version von OpenCms einen sicheren Zugriff auf den HTTP-Solr-Server unterstützen.

11.3.1.2 Suchindex (Indizes)

Standardmäßig hat OpenCms einen **Solr Online**-Index. Wenn Sie einen neuen Solr-Index hinzufügen wollen, können Sie die Standardkonfiguration als Kopiervorlage nutzen.

```
<index class="org.opencms.search.solr.CmsSolrIndex">
 <name>Solr Online</name>
 <rebuild>auto</rebuild>
 <project>Online</project>
 <locale>all</locale>
 <configuration>solr_fields</configuration>
 <sources>
   < source >solr_source< /source >
 </sources>
</index>
```

11.3.1.3 Indexquellen

Indexquellen für Solr können in der Datei `opencms-search.xml` genauso konfiguriert werden wie die Lucene-Indizes. Um das erweiterte XSD-Field-Mapping für XML-Inhalte zu verwenden, müssen Sie den neuen Dokumenttyp `xmlcontent-solr` der Liste der zu indizierenden Dokumenttypen hinzuzufügen:

```
<indexsource>
 <name>solr_source</name>
 <indexer class="org.opencms.search.CmsVfsIndexer"/>
 <resources>
   <resource>/sites/default/</resource>
 </resources>
 <documenttypes-indexed>
   <name>xmlcontent-solr</name>
   <name>containerpage</name>
   <name>xmlpage</name>
   <name>text</name>
   <name>pdf</name>
   <name>image</name>
   <name>msoffice-ole2</name>
   <name>msoffice-ooxml</name>
   <name>openoffice</name>
 </documenttypes-indexed>
</indexsource>
```

11.3.1.4 Neuer Dokumenttyp

Mit OpenCms Version 8.5 gibt es einen neuen Dokumenttyp namens `xmlcontent-solr`. Seine Implementierung(`CmsSolrDocumentXmlContent`) generiert einen lokalisierten Inhaltauszug, der später verwendet wird, um das Solr-Input-Dokument zu befüllen. Wie im Abschnitt Benutzerdefinierte Felder für XML-Inhalte erklärt, ist es möglich, ein Mapping zwischen Elementen im XSD eines XML-Ressourcentyps und einem Feld in einem Solr-Dokument zu definieren. Die Werte für diese definierten XSD-Feldzuordnungen werden auch durch den Dokumenttyp `xmlcontent-solr` extrahiert.

```
<documenttype>
 <name>xmlcontent-solr</name>
 <class>org.opencms.search.solr.CmsSolrDocumentXmlContent</class>
 <mimetypes>
   <mimetype>text/html</mimetype>
 </mimetypes>
 <resourcetypes>
   <resourcetype>xmlcontent-solr</resourcetype>
 </resourcetypes>
</documenttype>
```

11.3.1.5 Die standardmäßige Solr Feldkonfiguration

Standardmäßig ist die Feldkonfiguration für OpenCms Solr-Indizes, durch die Klasse `org.opencms.search.solr.CmsSolrFieldConfiguration` implementiert. Die einfachste in der Datei `opencms-search.xml` definierte Solr-Feldkonfiguration sieht wie folgt aus:

- Siehe auch Erweitern der CmsSolrFieldConfiguration

```
<fieldconfiguration class="org.opencms.search.solr.CmsSolrFieldConfiguration">
 <name>solr_fields</name>
 <description>The Solr search index field configuration.</description>
 <fields/>
</fieldconfiguration>
```

11.3.1.6 Migration eines Lucene-Index in einen Solr-Index

Eine bestehende Lucene Feld-Konfiguration kann leicht in eine Solr-Index umgewandelt werden. Um dies zu tun, erstellen Sie eine neue Solr Feld-Konfiguration. Dafür können Sie das im Abschnitt Die standardmäßige Solr Feldkonfiguration gezeigte Snippet als Vorlage nutzen und die Liste der Felder aus dem Lucene-Index kopieren, die Sie in diese Struktur konvertieren wollen.

Es gibt eine spezielle Strategie einem Lucene-Feldnamen einen Solr-Feldnamen zuzuordnen:

- **Exact name matching**: OpenCms versucht, ein explizites Solr-Feld zu bestimmen, welches genau den Wert des Namensattributs hat. Z. B. versucht OpenCms für `<field name="meta">... </field>` eine explizite Solr-Definition namens Meta zu finden. Um diese Strategie zu verwenden, müssen Sie die `schema.xml` von Solr manuell editieren und eine explizite Feld-Definition, benannt nach dem genauen Lucene-Feldnamen, hinzufügen.
- **Type specific fields:** In der bestehenden Lucene-Konfiguration sind typspezifische Feld-Definitionen nicht vorgesehen. Die `schema.xml` von Solr definiert aber unterschiedliche Datentypen für Felder. Wenn Sie interessiert sind, von diesen typspezifischen Vorteilen (wie sprachabhängige Feldanalyse und -Tokenisierung) Gebrauch zu machen, ohne die `schema.xml` zu bearbeiten, müssen Sie mindestens ein Type-Attribut für diese Felder definieren. Der Wert des Attributs kann ein beliebiger Name eines jeden `<dynamicField>` sein, welches in der `schema.xml` definiert ist und mit einem `*_` beginnt. Das resultierende Feld innerhalb des Solr-Dokuments wird dann folgendermaßen benannt `<luceneFieldName>_<dynamicFieldSuffix>`.
- **Fallback:** Wenn Sie kein Type-Attribut definiert haben und kein explizites Feld in der `schema.xml` existiert, das dem Lucene-Feldnamen entspricht, nutzt OpenCms als Fallback `text_general`. Z.B. ein Lucene-Feld `<field name="title" index="true"> ... </field>` wird als ein dynamisches Feld mit dem Namen `title_txt` im Solr-Index gespeichert.

Eine ursprüngliche Feldkonfiguration wie:

```
<fieldconfiguration>
 <name>standard</name>
 <description>The standard OpenCms 8.0 search field configuration.</description>
 <fields>
   <field name="content" store="compress" index="true" excerpt="true">
     <mapping type="content"/>
   </field>
   <field name="title-key" display="-" store="true" index="untokenized" boost="0.0">
     <mapping type="property">Title</mapping>
   </field>
   <field name="title" display="%(key.field.title)" store="false" index="true">
     <mapping type="property">Title</mapping>
   </field>
   <field name="keywords" display="%(key.field.keywords)" store="true" index="true">
     <mapping type="property">Keywords</mapping>
   </field>
```

```
   <field name="description" store="true" index="true">
     <mapping type="property">Description</mapping>
   </field>
   <field name="meta" display="%(key.field.meta)" store="false" index="true">
     <mapping type="property">Title</mapping>
     <mapping type="property">Keywords</mapping>
     <mapping type="property">Description</mapping>
   </field>
 </fields>
</fieldconfiguration>
```

könnte nach der Konvertierung etwa so aussehen:

```
<fieldconfiguration class="org.opencms.search.solr.CmsSolrFieldConfiguration">
 <name>standard</name>
 <description>The standard OpenCms 8.0 Solr search field configuration.</description>
 <fields>
   <field name="content" store="compress" index="true" excerpt="true">
     <mapping type="content"/>
   </field>
   <field name="title-key" store="true" index="untokenized" boost="0.0" type="s">
     <mapping type="property">Title</mapping>
   </field>
   <field name="title" store="false" index="true" type="prop">
     <mapping type="property">Title</mapping>
   </field>
   <field name="keywords" store="true" index="true" type="prop">
     <mapping type="property">Keywords</mapping>
   </field>
   <field name="description" store="true" index="true" type="prop">
     <mapping type="property">Description</mapping>
   </field>
   <field name="meta" store="false" index="true" type="en">
     <mapping type="property">Title</mapping>
     <mapping type="property">Keywords</mapping>
     <mapping type="property">Description</mapping>
   </field>
 </fields>
</fieldconfiguration>
```

11.3.2 Indizierte Daten

Die folgenden Abschnitte zeigen, welche Daten standardmäßig indiziert werden und welche Möglichkeiten von OpenCms angeboten werden, um weitere Feldkonfigurationen / -mappings zu erstellen.

11.3.2.1 Das Solr-Index-Schema (schema.xml)

Werfen Sie zuerst einen Blick auf das Solr `schema.xml`. In der Datei `<CATALINA_HOME>/webapps/<OPENCMS>/WEB-INF/solr/conf/schema.xml` finden Sie die Feld-Definitionen, die von OpenCms verwendet werden, die in den vorherigen Kapiteln kurz zusammengefasst wurden.

11.3.2.2 Standard-Index Felder

OpenCms indiziert standardmäßig mehrere Informationen für jede Ressource:

- **id** – Struktur-Id wird als eindeutige Kennung für ein Dokument benutzt (Die Struktur-Id der Ressource)
- **path** – Kompletter Root-Pfad (Der Root-Pfad der Ressource z.B. /sites/default/flower_en/.content/article.html)
- **path_hierarchy** – Der gesamte Pfad als („path tokenized“ Feldtyp: text_path)

- **parent-folders** - Übergeordnete Ordner (Mehrfach-Felder, die einen Eintrag für jeden übergeordneten Pfad enthalten)
- **type** – Typ-Name (der Ressourcentyp-Name)
- **res_locales** – Vorhandene Sprachknoten für XML-Content und alle verfügbaren Locales von Binär-Dateien
- **created** – Das Erstellungsdatum (Das Datum an dem die Ressource erstellt wurde)
- **lastmodified** – Datum der letzten Änderung (Das Datum an dem die Ressource zum letzten Mal geändert wurde)
- **contentdate** – Das Inhaltsdatum (Das Datum, wann sich die Ressourceninhalte geändert haben)
- **released** – Das Erscheinungs- und Ablaufdatum der Ressource
- **content** – Ein allgemeines Inhaltsfeld, das alle extrahierten Ressourcendaten enthält (Alle Sprachen, Typ `text_general`)
- **contentblob** – Das serialisierte Extraktionsergebnis (`content_blob`). Zur Verbesserung der Extraktions-Performance beim Indizieren
- **category** – Alle Kategorien als allgemeiner Text
- **category_exact**– Alle Kategorien als exakter String für die Facettierung
- **text_<locale>** - Extrahierter textlicher Inhalt, optimiert für sprachspezifische Suche (Standard-Sprachen: en, de, el, es, fr, hu, it)
- **timestamp** – Die Zeit als das Dokument zum letzten Mal indiziert wurde
- ***_prop** – Alle OpenCms-Eigenschaften einer Ressource als suchbarer und gespeicherter Text (Feldname: `<Property_Definition_Name>_prop` als `text_general`)
- ***_exact** – Alle OpenCms-Eigenschaften einer Ressource als genauen, nicht gespeicherten String (Feldname:`<Propery_Definition_Name>_exact` als `string`)

11.3.2.3 Benutzerdefinierte Felder für XML-Inhalte

Eine deklarative Feldkonfiguration mit Feld-Mappings kann auch über die `XSD-Content-Definition` eines XML-Ressourcen-Typs, wie in der `DefaultAppinfoTypes.xsd` definiert, vorgenommen werden.

```
<xsd:complexType name="OpenCmsDefaultAppinfoSearchsetting">
   <xsd:sequence>
     <xsd:element name="solrfield"
                  type="OpenCmsDefaultAppinfoSolrField"
                  minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="element" type="xsd:string" use="required" />
   <xsd:attribute name="searchcontent"
                  type="xsd:boolean" use="optional" default="true" />
 </xsd:complexType>

 <xsd:complexType name="OpenCmsDefaultAppinfoSolrField">
   <xsd:sequence>
     <xsd:element name="mapping"
                  type="OpenCmsDefaultAppinfoSolrFieldMapping"
                  minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
```

```
   <xsd:attribute name="targetfield" type="xsd:string" use="required" />
   <xsd:attribute name="sourcefield" type="xsd:string" use="optional" />
   <xsd:attribute name="copyfields" type="xsd:string" use="optional" />
   <xsd:attribute name="locale" type="xsd:string" use="optional" />
   <xsd:attribute name="default" type="xsd:string" use="optional" />
   <xsd:attribute name="boost" type="xsd:string" use="optional" />
 </xsd:complexType>
```

Für XML-Content-Elemente können Sie Suchfeldmappings auch direkt in der XSD-Content-Definition deklarieren. Eine XSD kann dann wie folgt aussehen:

```
<searchsettings>
   <searchsetting element="Title" searchcontent="true">
     <solrfield targetfield="atitle">
       <mapping type="property">Author</mapping>
     </solrfield>
   </searchsetting>
   <searchsetting element="Teaser">
     <solrfield targetfield="ateaser">
       <mapping type="item" default="Homepage n.a.">Homepage</mapping>
       <mapping type="content"/>
       <mapping type="property-search">search.special</mapping>
       <mapping type="attribute">dateReleased</mapping>
       <mapping type="dynamic"
                class="org.opencms.search.solr.CmsDynamicDummyField">special
       </mapping>
     </solrfield>
   </searchsetting>
   <searchsetting element="Text" searchcontent="true">
     <solrfield targetfield="ahtml" boost="2.0"/>
   </searchsetting>
   <searchsetting element="Release" searchcontent="false">
     <solrfield targetfield="arelease" sourcefield="*_dt" />
   </searchsetting>
   <searchsetting element="Author" searchcontent="true">
     <solrfield targetfield="aauthor" locale="de"
                copyfields="test_text_de,test_text_en" />
   </searchsetting>
   <searchsetting element="Homepage" searchcontent="true">
     <solrfield targetfield="ahomepage" default="Homepage n.a." />
   </searchsetting>
 </searchsettings>
```

11.3.2.4 Dynamische Feldzuordnungen

Sind die Anforderungen an das Feld-Mapping „dynamischer“, dann nur: `'static piece of content' -> 'specified field defined in the Solr schema'` sind Sie in der Lage das Interface `org.opencms.search.fields.I_CmsSearchFieldMapping` zu implementieren.

11.3.2.5 Benutzerdefinierte Feldkonfiguration

Deklarative Feldkonfigurationen mit Feld-Mappings können in der `opencms-search.xml` vorgenommen werden. Sie können genau die gleichen Funktionen verwenden, wie Sie sie bereits von der OpenCms-Lucene-Feldkonfiguration her kennen.

- Siehe auch Kapitel Migration eines Lucene-Index in einen Solr-Index

11.3.2.6 Erweitern der CmsSolrFieldConfiguration

Wenn die Standard-Konfigurationsoptionen immer noch nicht flexibel genug sind, können Sie die Klasse: `org.opencms.search.solr.CmsSolrFieldConfiguration` erweitern und eine benutzerorientierte Solr-Feldkonfiguration in der `opencms-search.xml` definieren.

```
<fieldconfiguration class="your.package.YourSolrFieldConfiguration">
  <name>solr_fields</name>
  <description>The Solr search index field configuration.</description>
  <fields/>
</fieldconfiguration>
```

11.4 Hinter der Solr-Kulisse

11.4.1 Der Request-Handler

Die Klasse `org.opencms.main.OpenCmsSolrHandler` bietet die gleiche Funktionalität wie der standardmäßige Select-Request-Handler einer Standard-Solr-Server-Installation. In der Standard-OpenCms-Systemkonfiguration (`opencms-system.xml`) ist der Solr-Request-Handler konfiguriert.

```
<requesthandlers>
      <requesthandler class="org.opencms.main.OpenCmsSolrHandler" />
</requesthandlers>
```

Alternativ kann die Request-Handlerklasse als Servlet genutzt werden. Dafür fügen Sie die Handlerklasse in die `WEB-INF/web.xml` Ihrer OpenCms-Applikation ein.

```
  <servlet>
      <description>
          The OpenCms Solr servlet.
      </description>
      <servlet-name>OpenCmsSolrServlet</servlet-name>
      <servlet-class>org.opencms.main.OpenCmsSolrHandler</servlet-class>
      <load-on-startup>1</load-on-startup>
  </servlet>
    [...]
  <servlet-mapping>
      <servlet-name>OpenCmsSolrServlet</servlet-name>
      <url-pattern>/solr/*</url-pattern>
  </servlet-mapping>
```

11.4.2 Berechtigungsprüfung

OpenCms führt eine Überprüfung der Berechtigungen für alle gefundenen Dokumente durch und wirft diejenigen weg, für die der aktuelle Benutzer keine Berechtigung hat sie aufzurufen und erweitert das Ergebnis um die nächstbesten passenden Dokumente „on the fly“. Diese Berechtigungsprüfung ist sehr rechenintensiv und sollte durch eine reine indexbasierte Berechtigungsüberprüfung ersetzt bzw. verbessert werden.

11.4.3 Konfigurierbarer Post-Prozessor

OpenCms bietet die Fähigkeit der nachträglichen Suchverarbeitung von Solr-Dokumenten, nachdem die Berechtigungen des Dokuments geprüft wurden. Diese Fähigkeit ermöglicht Ihnen, dem gefundenen Dokument Felder hinzuzufügen, bevor das Suchergebnis zurückgegeben wird. Um den Post-Prozessor zu nutzen, müssen Sie einen optionalen Parameter für den Suchindex hinzufügen:

```
<index class="org.opencms.search.solr.CmsSolrIndex">
   <name>Solr Offline</name>
   <rebuild>offline</rebuild>
   <project>Offline</project>
   <locale>all</locale>
   <configuration>solr_fields</configuration>
   <sources>
```

```
    [...]
  </sources>
  <param name="org.opencms.search.solr.CmsSolrIndex.postProcessor">
    my.package.MyPostProcessor
  </param>
</index>
```

Die spezifizierte Klasse des Parameters `org.opencms.search.solr.CmsSolrIndex.postProcessor` must be an implementation of `org.opencms.search.solr.I_CmsSolrPostSearchProcessor`.

11.4.4 Mehrsprachige Unterstützung

Es gibt eine Standardstrategie wie die Unterstützung der Mehrsprachigkeit innerhalb des OpenCms-Solr-Suchindex implementiert ist. Für binäre Dokumente wird die Sprache automatisch auf der Grundlage des extrahierten Textes bestimmt. Der Standard-Mechanismus ist implementiert durch: http://code.google.com/p/language-detection/

Für XML-Inhalte haben wir die konkrete Sprach- / Locale-Information und die lokalisierten Felder, die mit Unterstrich, gefolgt von der Locale enden. Z.B.: `content_de`, `content_en` oder `text_de`, `text_en`. Standardmäßig werden alle Feld-Zuordnungen innerhalb der XSD eines Ressource-Typs durch die `'_<locale>'` erweitert.

11.4.5 Auflösung mehrsprachiger Abhängigkeiten

Basierend auf dem Dateinamen einer Ressource in OpenCms existiert ein Konzept zur Indexierung von Dokumenten, die über mehr als eine Ressource in OpenCms verteilt sind. Die Standard-Implementierung finden Sie unter:
`org.opencms.search.documents.CmsDocumentDependency`

11.4.6 Extrahierter Ergebnis-Cache

Für eine bessere Index-Performance wird das extrahierte Ergebnis für Verknüpfungen (Siblings) zwischengespeichert. Siehe:
`org.opencms.search.extractors.I_CmsExtractionResult`

11.5 Häufig gestellte Fragen

11.5.1 Wie ist Solr allgemein integriert?

Unabhängig von OpenCms bietet ein Standard-Solr-Server eine HTTP-Schnittstelle in einer Standard-Apache-Solr-Installation über http://localhost:8983/solr/select erreichbar ist:

Es ist Ihnen möglich jede gültige Solr-Query, die auf der Seite http://wiki.apache.org/solr/SolrQuerySyntax dokumentiert ist, abzufragen.

Die HTTP-Antwort kann entweder JSON oder XML sein. Und die Antwort der Abfrage http://localhost:8983/solr/select?q=*:*&rows=2 könnte so aussehen:

```
<response>
  <lst name="responseHeader">
    <int name="status">0</int>
    <int name="QTime">32</int>
    <lst name="params">
    <str name="q">*:*</str>
    <int name="rows">2</int>
    <long name="start">0</long>
  </lst>
  <result name="response" numFound="139" start="0">
    <doc>...</doc>
```

```
    <doc>...</doc>
  </result>
</response>
```

Solr ist in Java implementiert und es gibt eine Apache-Bibliothek namens `solrj`, durch die man auf einen laufenden Solr-Server durch das Schreiben von Java-Code zugreifen kann.
Die Solr-Integration in OpenCms bietet beide Schnittstellen: Java und HTTP. Die Standard-URL für das Senden von Solr-Abfragen an OpenCms ist:
http://localhost:8080/opencms/opencms/handleSolrSelect

Dieser Handler kann jede syntaktisch korrekte Solr-Abfrage beantworten (http://wiki.apache.org/solr/SolrQuerySyntax).

Der folgende Code zeigt ein einfaches Beispiel, wie man gegen die Java API programmiert:

```
//////////////////
// SEARCH START //
//////////////////
CmsObject cmsO = new CmsJspActionElement(pageContext, request,
response).getCmsObject();
String query = ((request.getParameter("query") != null && request.getParameter("query")
!= "") ? "q=" + request.getParameter("query") : "") + "&fq=type:ddarticle&sort=path
asc&rows=5";
CmsSolrResultList hits = OpenCms.getSearchManager().getIndexSolr("Solr
Offline").search(cmsO, query);
if (hits.size() > 0) { %>
  <h4>New way:

    <fmt:message key="v8.solr.results" />

    <%= hits.getNumFound() %> found / rows <%= hits.getRows() %></h4>
      <div class="boxbody"><%
        //////////////////
        // RESULTS LOOP //
        //////////////////
        for (CmsSearchResource resource : hits) { %>
          <div class="boxbody_listentry">
            <div class="twocols">
            <div>Path: <strong><%= resource.getRootPath() %></strong></div>
            <div>German: <strong>

              <%= resource.getDocument().getFieldValueAsString("Title_de") %></strong>

            </div>

            <div>English: <strong>

              <%= resource.getDocument().getFieldValueAsString("Title_en") %></strong>

            </div>
          </div>
        </div> <%
      }
    %>

  </div><%
}
```

11.5.2 Wie sortiert man Text für bestimmte Sprachen?

In diesem Beispiel ist der Text standardmäßig durch die durch Java bereitgestellten deutschen Vorschriften sortiert. Die Regeln für die deutsche Sortierung in Java wird definiert in einem Paket namens „Java Locale“.
Locales sind in der Regel definiert als eine Kombination aus Sprache und Land. Wenn man

möchte, kann man auch nur die Sprache angeben. Zum Beispiel, wenn Sie "de" als Sprache angeben, erhalten Sie eine Sortierung, die gut für die deutsche Sprache funktioniert. Wenn Sie "de" als Sprache und "CH" als das Land angeben, erhalten Sie deutsche Sortierung speziell auf die Schweiz zugeschnitten. Eine Liste der unterstützten Locales finden Sie hier. Sie erhalten mehr allgemeine Informationen darüber, wie die Textanalysen mit Solr funktionieren, wenn Sie einen Blick auf diese Seiten werfen: Sprachanalyse

```
<!-- define a field type for German collation -->
<fieldType name="collatedGERMAN" class="solr.TextField">
  <analyzer>
   <tokenizer class="solr.KeywordTokenizerFactory"/>
    <filter class="solr.CollationKeyFilterFactory"
        language="de"
        strength="primary"
    />
  </analyzer>
</fieldType>
...
<!-- define a field to store the German collated manufacturer names -->
<field name="manuGERMAN" type="collatedGERMAN" indexed="true" stored="false" />
...
<!-- copy the text to this field. We could create French, English, Spanish versions
     too, and sort differently for different users! -->
<copyField source="manu" dest="manuGERMAN"/>
```

11.5.3 Wie wird das Highlighting im Suchergebnis angewendet?

11.5.3.1 Unterstützt OpenCms das Highlighting im Suchergebnis?

Ja, verwenden Sie den OpenCms-Solr-Select-Handler unter:
localhost: 8080/opencms/opencms/handleSolrSelect
und Sie werden den Highlighting-Bereich unterhalb der Liste der Dokumente innerhalb des zurückgegebenen XML / JSON finden:

```
<lst name="highlighting">
  <lst name="a710bb16-1e04-11e2-b767-6805ca037347">
    <arr name="content_en">
      <str><em>YIPI</em> <em>YOHO</em> text text text</str>
    </arr>
  </lst>
  [...]
</lst>
```

11.5.3.2 Unterstützt das Java API von OpenCms Highlighting?

Derzeit unterstützt das OpenCms-Such-API das Solr-Highlighting nicht vollständig. Aber Sie können den standardmäßigen Solr-Highlighting-Mechanismus nutzen. Siehe [1] u. [2]

1. Wenn Sie `org.opencms.search.solr.CmsSolrResultList#getSolrQueryResponse()` aufrufen erhalten Sie einen `SolrQueryResponse`. Dieser ist auf http://lucene.apache.org/solr/api-3_6_1/org/apache/solr/response/SolrQueryResponse.html dokumentiert
2. Oder nutzen Sie den oben genannten OpenCms-Solr-Select-Handler unter: localhost:8080/opencms/opencms/handleSolrSelect

11.5.3.3 Ist Highlighting ein Performance-Killer?

Ja, aus diesem Grund ist das Highlighting ausgeschaltet, bevor die erste Suche ausgeführt wird. Nachdem alle verbotenen Ressourcen aus der Ergebnisliste gefiltert sind, wird Highlighting erneut durchgeführt.

11.5.4 Solr Indizierungs-Fragen

11.5.4.1 Bitte erklären Sie den Unterschied zwischen “Solr Online und Offline”

Wie der Name der Indizes vermuten lässt, enthalten Offline-Indizes auch Änderungen, die noch nicht veröffentlicht wurden. Online-Indizes enthalten nur die Ressourcen, die bereits veröffentlicht wurden. Die "Online EN VFS" ist ein Lucene basierender Index und enthält auch nur die Ressourcen, die veröffentlicht wurden.

11.5.4.2 Wird bei einer Solr-Abfrage nur der Solr Index gebraucht?

Nein, die Berechtigungen werden anschließend von dem OpenCms API überprüft.

11.5.5 Wo finde ich allgemeine Information über Solr?

Wenn Sie an Solr ein allgemeines Interesse haben, ist das Solr Wiki eine gute Möglichkeit sich einen ersten Überblick zu verschaffen: http://wiki.apache.org/solr/. Die Dokumentation aus CMS-Sicht finden Sie auf den Seiten der mitgelieferten PDF-Datei.

11.5.5.1 Gibt es eine Möglichkeit, eine vollständige Sicherung des gesamten Index zu erstellen?

Sie können den Index-Ordner `WEB-INF/index/${INDEX_NAME}` von Hand kopieren.

11.5.5.2 Wie kann man einen ausfallsicheren Index wiederherstellen?

Bearbeiten Sie die `opencms-search.xml` innerhalb Ihres `WEB-INF/config`-Verzeichnisses und fügen Sie den folgenden Knoten zu Ihrem Index hinzu:

```
<param name="org.opencms.search.CmsSearchIndex.useBackupReindexing">true</param>
```

Dadurch wird ein Snapshot, wie auf der folgenden Seite erläutert, erstellt:
http://wiki.apache.org/solr/CollectionDistribution

11.5.5.3 Die Solr Ergebnisliste wird standardmäßig auf 50 begrenzt, wie kann man mehr als 50 Ergebnisse bekommen?

Um nur berechtigungsgeprüfte Ressourcen zurückzubekommen (was eine rechenintensive Aufgabe ist), liefern wir nur ein limitierte Anzahl von Ergebnissen zurück. Für das Paging der Suchergebnisse werfen Sie bitte einen Blick auf die Solr-Parameter und starten mit: http://wiki.apache.org/solr/CommonQueryParameters. Seit der Version 8.5.x können Sie die Anzahl der angezeigten Dokumente im Suchergebnis bis zu einer Größe Ihrer Wahl erhöhen.

11.5.6 Solr Mailinglist Fragen

11.5.6.1 Eine ClassCastException wird geworfen, was kann ich tun?

Sie müssen die richtigen Klassen für den Index und die Feldkonfiguration angeben. Ansonsten wird der implementierte Lucene-Such-Index verwendet.

```
<index class="org.opencms.search.solr.CmsSolrIndex">[...]</index>
<fieldconfiguration class="org.opencms.search.solr.CmsSolrFieldConfiguration">[...]
</fieldconfiguration>
```

11.5.6.2 Ist es möglich die Elemente mit maxoccurs > 1 abzubilden?

Seit der Version >= 8.5.1 werden diese auf ein Multivalue-Feld abgebildet.

11.5.6.3 Wie werden OpenCmsDateTime-Elemente indiziert?

```
<searchsetting element="Release" searchcontent="false">
    <solrfield targetfield="arelease" sourcefield="*_dt" />
</searchsetting>
```

12 SEO

12.1 Einführung

OpenCms 8.5 enthält zwei neue Funktionen für den Zweck der Suchmaschinenoptimierung: Aliase und SEO-Dateien. Aliase erlauben Ihnen alternative URLs für Ihre Seiten zu definieren, die nicht den tatsächlichen Pfaden im VFS entsprechen müssen. SEO-Dateien können verwendet werden, um automatisch XML-Sitemaps und robots.txt-Dateien für Ihre Website zu generieren.

12.2 Aliase

12.2.1 Einfache Aliasnamen

Es gibt zwei Arten von Aliasnamen, die definiert werden können: **Einfache Aliasnamen** und **Rewrite-Aliasnamen**. **Einfache Aliase** bestehen aus einem Alias-Pfad, einer Ziel-Ressource und einer Aktion, die bestimmt was passieren soll, wenn der Alias-Pfad von OpenCms angefordert wird. **Einfache Aliasnamen** werden direkt mit ihrer Ziel-Ressource verbunden, so dass sie weiterhin auf die Ressource verweisen, auch wenn diese verschoben oder umbenannt wird. Gültige Alias-Pfade bestehen aus einem oder mehreren Segmenten, welche immer aus einem "/" und einem oder mehreren anderen Zeichen bestehen. Dies bedeutet, dass sie nicht mit einem abschließenden "/" enden dürfen. Es gibt drei mögliche Aktionen, die durchgeführt werden können, wenn der Pfad eines Aliases von OpenCms angefordert wird:

- **Temporäre Weiterleitung (302):** Eine temporäre Umleitung wird an den Browser gesendet, mit dem aktuellen Link zur Zielressource als neue URL
- **Permanente Weiterleitung (301):** Eine dauerhafte Umleitung wird an den Browser gesendet, mit dem aktuellen Link zur Zielressource als neue URL
- **Seite zeigen:** OpenCms wird versuchen, die Zielressource direkt im aktuellen Request zu laden.

12.2.2 Rewrite-Aliase

Während ein **einfacher Alias** einen einzelnen Pfad zu einer einzelnen Ressource abbildet, kann ein **Rewrite-Alias** verwendet werden, um einen Alias für ganze Klassen von Pfaden durch die Angabe eines regulären Ausdrucks zu definieren. Dafür wird ein regulärer Ausdruck für die Quellpfade und ein Ersetzungsstring für die Zielpfade definiert.

OpenCms wird einen eingehenden Pfad gegen die Rewrite-Alias-Muster testen und wendet den ersten passenden Rewrite-Alias an. Es ist keine Reihenfolge für die Zuordnung definiert, deshalb sollten Sie keine Rewrite-Aliasnamen mit überlappenden Pattern-Matches definieren.

Das Muster für einen Rewrite-Alias folgt der Standard-Java-Syntax für reguläre Ausdrücke. Der Ersetzungs-String folgt der Syntax für den Parameter der Methode

`java.util.regex.Matcher#replaceFirst`. D.h. auf den Inhalt der Capture-Gruppen aus den regulären Ausdrücken kann mit der Dollar-Syntax ($ 1, $ 2, ...) zugegriffen werden. Das Muster wird immer gegen den gesamten Pfad geprüft. **Rewrite-Aliase** haben immer Vorrang vor **einfachen Aliasnamen**, d.h. wenn ein **Rewrite-Alias** dem aktuellen Anforderungspfad entspricht, wird ein **einfacher Alias** für diesen Pfad wird nicht angewendet.

Es gibt drei mögliche Aktionen für Rewrite-Aliasnamen:

- **Temporäre Weiterleitung (302):** Eine temporäre Umleitung wird an den Browser gesendet, mit dem aktuellen Link zur Zielressource als neue URL
- **Permanente Weiterleitung (301):** Eine dauerhafte Umleitung wird an den Browser gesendet, mit dem aktuellen Link zur Zielressource als neue URL
- Passthrough: Der Zielpfad, der sich aus dem Ersetzungsmuster ergibt, wird an den nächsten Ressource-Handler im `<resourceinit>`-Element nach dem Alias-Handler übergeben.

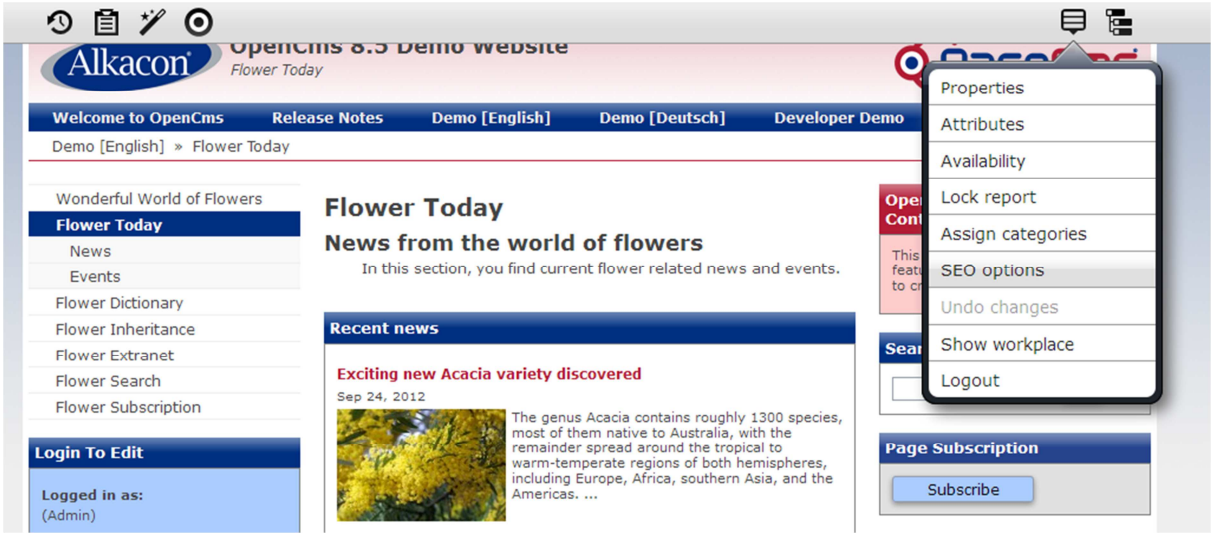
12.2.3 Interna

Alle Aliase für eingehende Requests werden durch die Klasse `org.opencms.main.CmsAliasResourceHandler` behandelt. Standardmäßig wird dies in der `opencms-system.xml`-Konfigurationsdatei im `<resourceinit>`-Element konfiguriert. Auch wenn dieser Handler von der Konfiguration entfernt wird, ist es immer noch möglich, Aliasnamen über die Benutzeroberfläche zu bearbeiten, obwohl diese keinen Effekt haben wird.

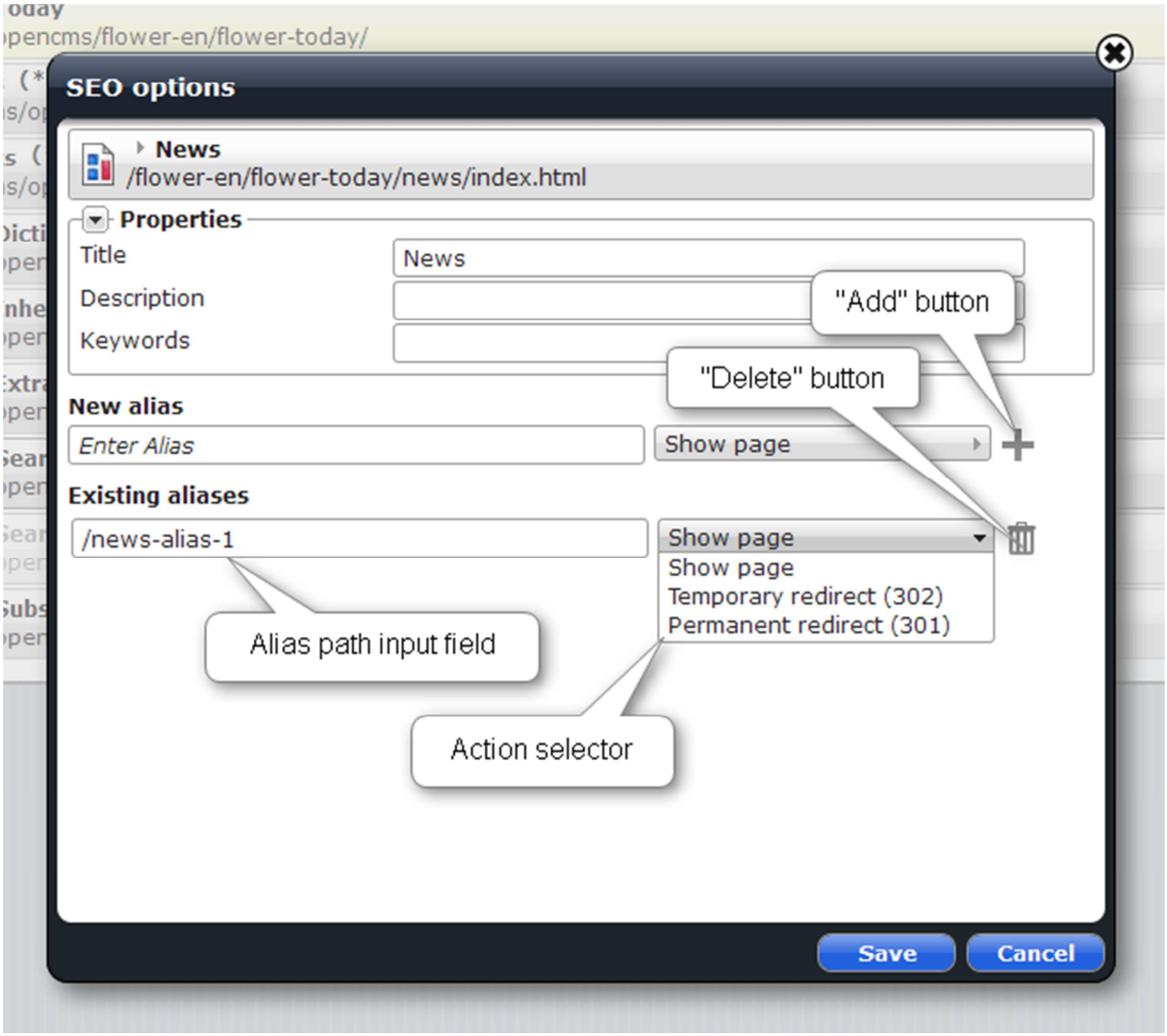
Für Pfade, die tatsächlich im VFS existieren und für Pfade, mit `/system/` beginnen, wird der Alias-Ressource-Handler nicht verwendet.
Beachten Sie, dass Aliase nur für eine einzelne OpenCms-Site gelten. Zum Beispiel ist es möglich, den gleichen Aliaspfad für verschiedene einfache Aliase in verschiedenen OpenCms-Sites zu verwenden.
Beachten Sie, dass aufgrund der Beschränkungen von Redirects, der Request-Parameter nicht beibehalten wird, wenn eine Umleitungsaktion durchgeführt, weil ein POST-Request-Pfad mit einem Alias-Pfad übereinstimmt.

12.2.4 SEO Optionsdialog

Es ist ein neuer SEO-Optionen Kontextmenüeintrag im Seiteneditor-Kontextmenü und im Kontextmenü für Sitemapeinträge verfügbar:



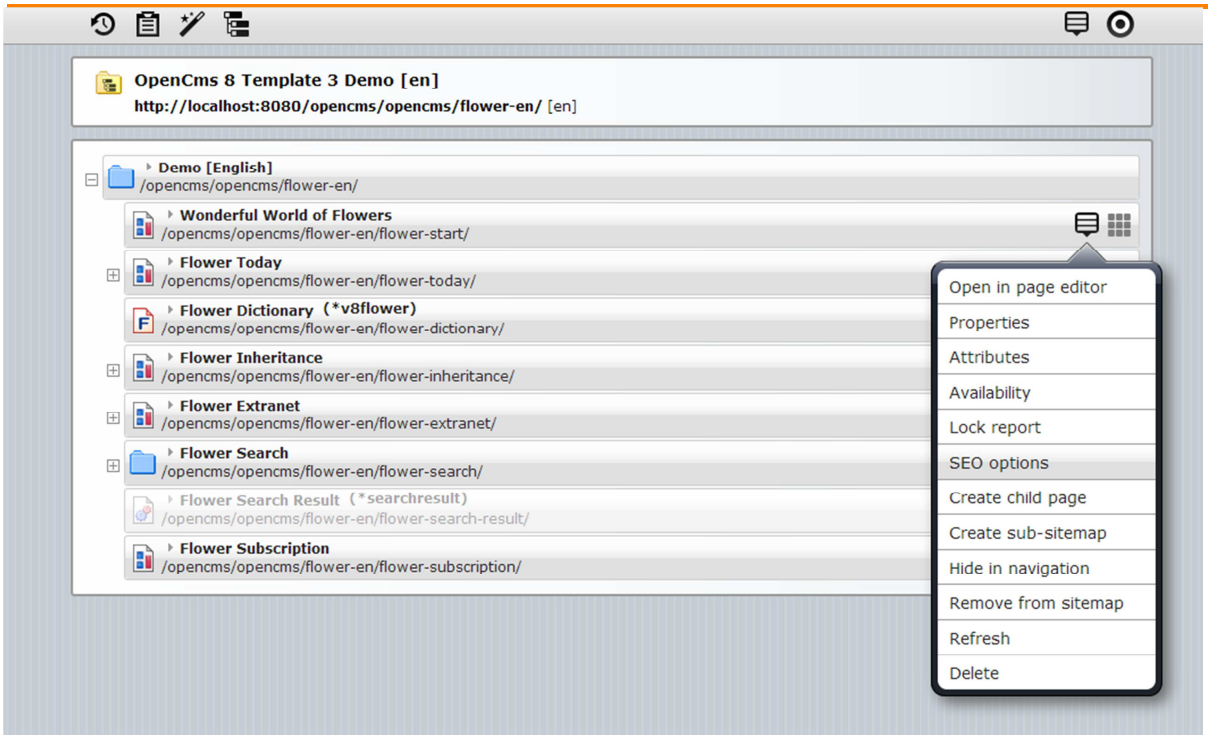
Wenn dieser Eintrag aus dem Kontextmenü ausgewählt wird, öffnet sich der SEO-Optionen-Dialog:



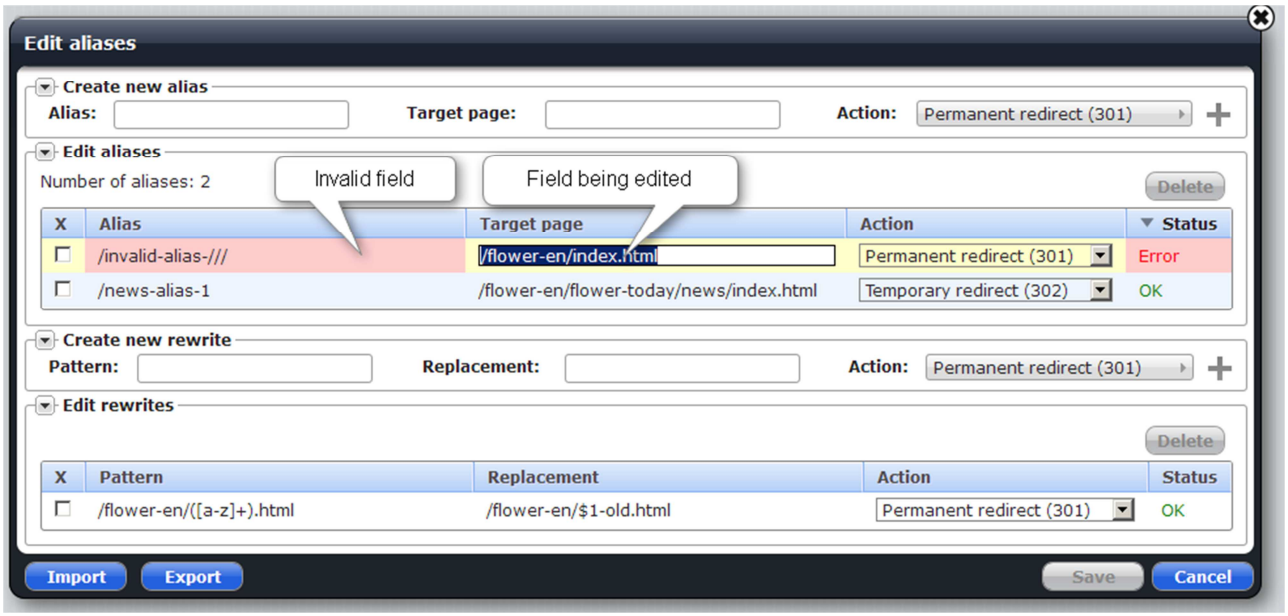
Im SEO-Optionen-Dialog können Sie die Eigenschaften **Titel**, **Beschreibung** und **Keywords** ändern und den **einfachen Alias** für die Ressource bearbeiten. Beachten Sie, dass Änderungen nur angewendet werden, wenn Sie die Schaltfläche **Speichern** klicken.
Sie können einen vorhandenen Alias ändern, indem Sie den Pfad im Textfeld ändern oder die Aktion mit der Aktion-Auswahlbox bearbeiten. Die Schaltfläche **Löschen** kann verwendet werden, um den Alias zu entfernen.
Durch Eingabe des Alias-Pfads im Textfeld für neue Alias-Pfade, die Auswahl der Aktion aus der Select-Box und anschließendem Klick auf die Schaltfläche "Hinzufügen" können Sie einen neuen Alias hinzufügen.

12.2.5 Aliase Bearbeiten-Dialog

Der **Aliase bearbeiten**-Dialog ermöglicht Ihnen alle Aliase (einfache und Rewrite) für die aktuelle OpenCms-Site anzuzeigen oder zu bearbeiten. Sie können diese über das Sitemap-Editor-Kontextmenü öffnen:



Wenn dieser Eintrag aus dem Kontextmenü ausgewählt ist, öffnet sich der SEO-Optionen-Dialog:



Dieser Dialog ermöglicht es Ihnen neue und bestehende Aliase zu editieren und Aliase der aktuellen Site aus einer CSV-Datei zu importieren und in eine CSV-Datei zu exportieren.

12.2.6 Neue Aliase erstellen

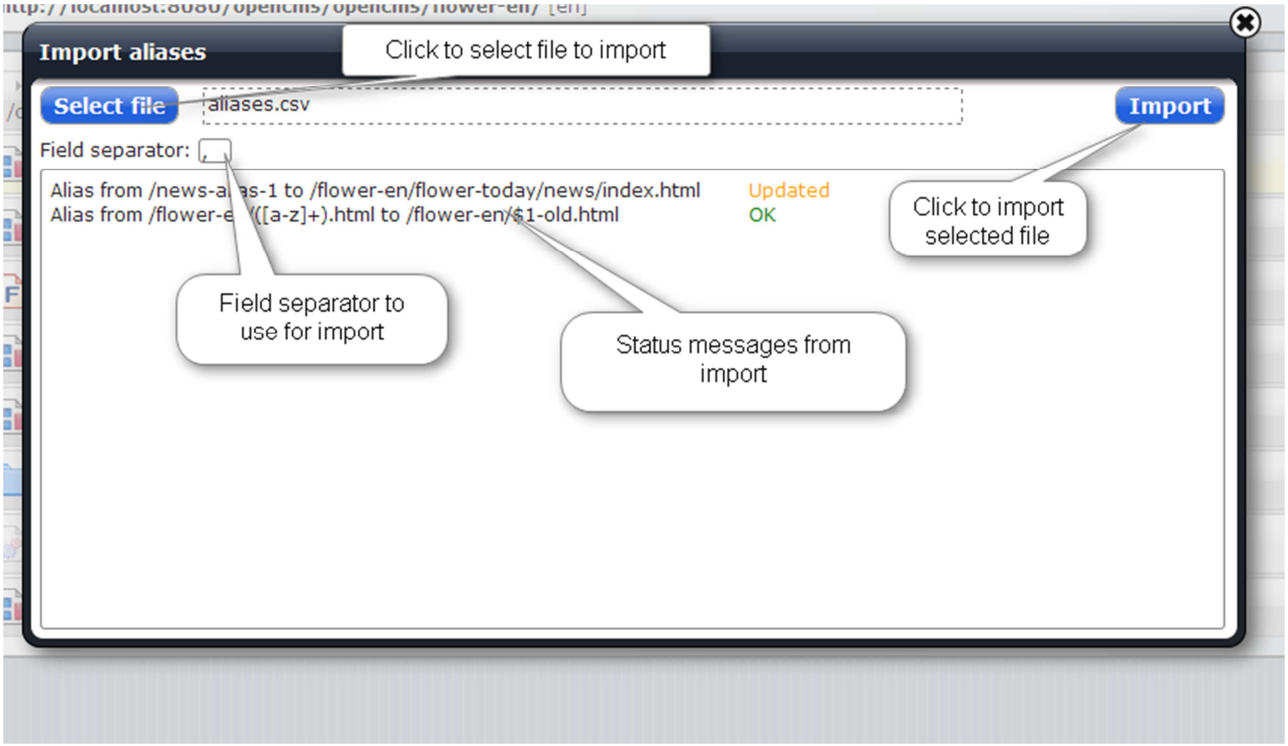
Sie können neue Aliase erstellen, in dem Sie einen Alias-Pfad und die Zielseite eingeben (oder einen regulären Ersetzungsausdruck), dann eine Aktion auswählen und auf den Button **Hinzufügen** klicken.

12.2.7 Bearbeiten von vorhandenen Aliasen

Bestehende Aliasnamen (entweder einfache oder Rewrite-Aliasnamen) können in ihren jeweiligen Tabellen bearbeitet werden. Sie können ein vorhandenes Alias-Attribut ändern, indem Sie auf den entsprechenden Tabelleneintrag klicken. Wenn Sie ungültige Aliaspfade eingeben, wird das Feld **Status** der entsprechenden Tabelle einen **Error**-Label anzeigen. Sie können eine ausführlichere Meldung erhalten, wenn Sie mit Ihren Mauszeiger über dieses Label fahren.
Sie können vorhandene Aliasnamen löschen, indem Sie die Checkboxen in der Spalte **X** der entsprechenden Tabellenzeilen setzen und dann auf die Schaltfläche **Löschen** über der Tabelle klicken.

12.2.8 Exportieren und Importieren von Aliasen

Sie können auf die Schaltfläche **Exportieren** klicken, um eine CSV-Datei mit den aktuell definierten Aliasnamen für die Site herunterzuladen. Beachten Sie, dass nicht gespeicherte Alias-Änderungen aus dem Dialog nicht berücksichtigt werden.
Durch Klicken auf die Schaltfläche **Importieren** öffnet sich ein weiteres Dialogfeld, mit dem Sie eine zuvor exportierte CSV-Datei importieren können.



Klicken Sie zuerst auf den **Datei auswählen**-Button um eine CSV-Datei für den Import auszuwählen. Dann klicken Sie den **Import-**Button. Es ist möglich, dass die CSV-Datei mit einem Programm generiert wurde, welches ein anderes Feldtrennzeichen als',' verwendet. Dann können Sie ein Trennzeichen setzen damit die Datei geparst wird, bevor Sie auf **Import** klicken.

Die Aliase werden dann importiert und die Statusmeldung wird im Textfeld am Ende des Dialoges angezeigt. Das Importformat einer CSV-Datei sieht wie folgt aus:

```
"/news-alias-1","/flower-en/flower-today/news/index.html","redirect"
"/news-alias-2","/flower-de/flower-today/news/index.html"
"/flower-en/([a-z]+).html","/flower-en/$1-old.html","permanentRedirect","rewrite"
```

Jede Zeile besteht aus zwei, drei oder vier Feldern. Wenn die Zeile aus 3 Feldern besteht, wird sie als ein **einfacher Alias** mit folgenden Felddefinitionen interpretiert:

- Feld 1: Der Alias-Pfad
- Feld 2: Der Seiten-Pfad des Ziel-Alias
- Feld 3: Die Aktion für rewrite.
    - Seite: Seite anzeigen
    - permanentRedirect: Sende dauerhafte Umleitung
    - umleiten: Sende temporäre Umleitung

Zeilen mit 2 Feldern werden auch als einfache Alias-Pfade interpretiert und implizit auf den Alias-Modus **permanentRedirect** gesetzt.

Wenn die Zeile 4 Felder enthält, wird es als ein Rewrite-Alias mit folgenden Feld Definitionen interpretiert:

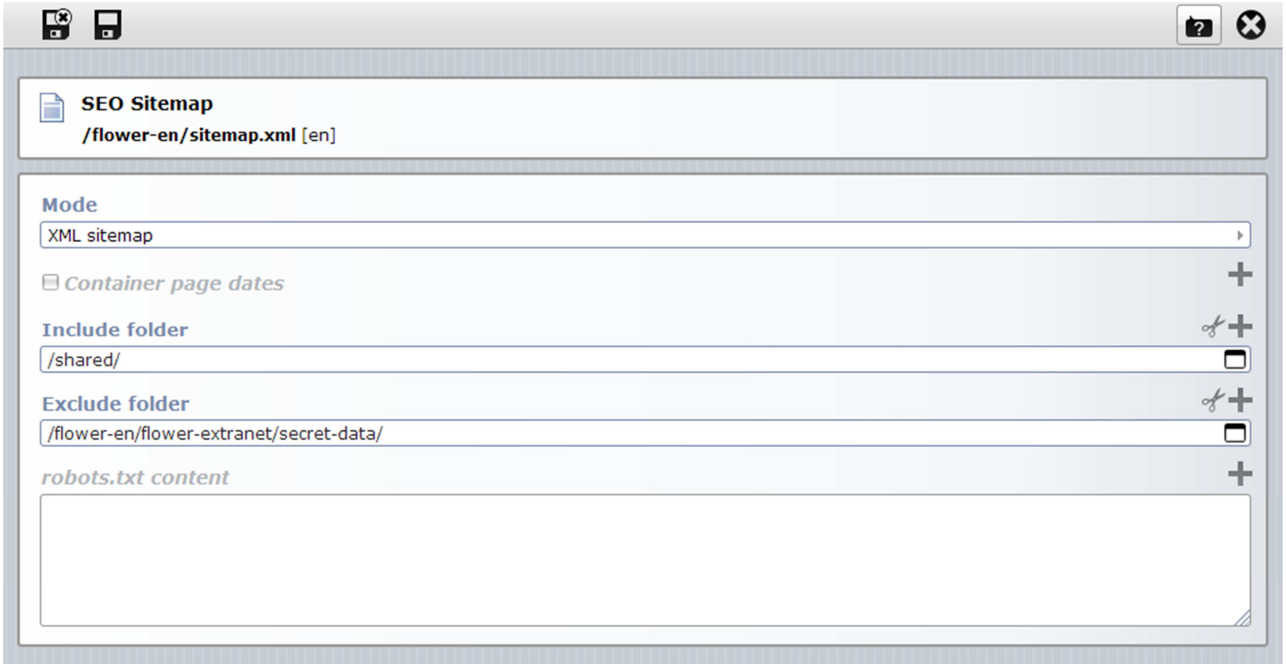
- Feld 1: Der Alias-Pattern
- Feld 2: Der Ersetzungs-String
- Feld 3: Die Aktion für die Rewrite
    - Passthrough: Übergebe den umgeschriebenen Pfad an den nächsten Ressource-Handler
    - permanentRedirect, redirect: wie oben
- Feld 4: Muss der konstante Wert **rewrite** sein

12.3 Automatische robots.txt- und XML-Sitemap-Generierung

OpenCms bietet eine automatische Generierung von XML-Sitemaps und robots.txt-Dateien, die auf die Sitemaps verweisen an. Beide stehen durch den neuen Ressourcentyp `seo_file` zur Verfügung. Wenn eine Ressource dieser Art von OpenCms angefordert wird, generiert OpenCms dynamisch eine Sitemap oder die robots.txt-Daten und sendet sie an den Client.

12.3.1 XML-Sitemap-Generierung

Erstellen Sie eine neue Ressource vom Typ **SEO-File** in dem Ordner, für den Sie die XML-Sitemap generieren möchten. Öffnen Sie diese Datei mit dem Inhaltseditor. Achten Sie darauf, dass der Modus **XML-Sitemap** ausgewählt ist.



Standardmäßig enthält die XML-Sitemap-Navigation alle Einträge in und unterhalb des aktuellen Ordners, die URLs für alle Detailseiten, deren Basis-Containerseiten im oder unterhalb des aktuellen Verzeichnisses sind und alle Aliase deren Alias-Pfade unterhalb des aktuellen Ordners sind und deren Aufgabe als **Zeige Seite** definiert ist.
Ordner hinzufügen-Einträge erlaubt Ihnen Ordner zu definieren, deren gesamter Inhalt, inklusive Ressourcen aus Unterordnern in die XML-Sitemap eingefügt wird.

Ordner ausschließen-Einträge erlaubt Ihnen Ordner zu definieren, deren gesamter Inhalt, inklusive Ressourcen aus Unterordnern aus der XML-Sitemap ausgeschlossen wird.

Wenn Sie die **Containerpage-Änderungsdaten**-Option auswählen, wird das Datum der letzten Änderung eines Containerseiten-Inhalts berechnet.
Beachten Sie, dass die XML-Sitemap-Ausgabe immer den Online-Projekt-Zustand der Ressourcen berücksichtigt und nur Seiten, die für den Guest-Benutzer sichtbar sind enthält.

12.3.2 Generierung einer robots.txt-Datei

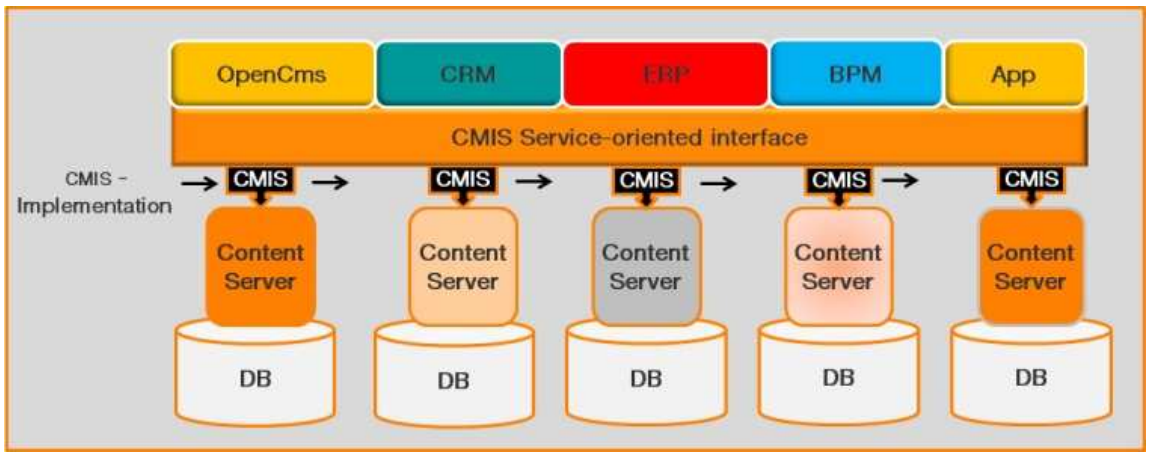
Um diese Funktion nutzen zu können, erstellen Sie eine neue Datei vom Typ **SEO-Datei** im Root-Verzeichnis Ihrer Website. Diese Funktion ist nur sinnvoll, wenn Ihr Host so eingerichtet ist, dass die OpenCms Website dem Root Ihrer Domain zugeordnet ist und http://yourhost.com/robots.txt vom OpenCms-Website-Root-Ordner abgerufen werden kann. Bearbeiten Sie die SEO-Datei und wählen Sie den Modus **robots.txt**. Wenn die **robots.txt**-Datei von OpenCms angefordert wird, enthält diese standardmäßig nur Referenzen auf bestehende XML-Sitemaps Ihrer Website. Um weitere Einträge hinzuzufügen, müssen Sie diese in das **robots.txt-Inhalt**-Feld Ihrer SEO-Datei eintragen.

13 CMIS

Viele Unternehmen wünschen sich seit Jahren eine einheitliche Lösung, die es ihnen erlaubt, Dokumente unabhängig vom System über standardisierte Technologien zu durchsuchen und zu verwenden. Es wird erwartet, dass die verschiedenen Business Applikationen zusammenarbeiten. Damit müssen auch der Zugriff und die Bearbeitung von Dokumenten unter Berücksichtigung der verschiedenen Dokumentenablagen im Unternehmen überall funktionieren. Der Standard CMIS – Content Management Interoperability Services – zeigt den Weg.
Der CMIS Standard bringt Interoperabilität und Offenheit in die Enterprise Content Management Welt, die CMIS Solutions Plattform verbindet die beteiligten Hersteller, Partner und Endanwender, damit diese Interoperabilität auch tatsächlich funktioniert.

13.1 OpenCms und CMIS

Mit OpenCms 8.5 ist es möglich auf Inhalte im virtuellen Dateisystem von OpenCms (VFS) über CMIS (Content Management Interoperability Services) zuzugreifen. Das folgende Kapitel enthält ein paar praktische Beispiele, wie man per CMIS auf eine laufende OpenCms-Instanz zugreift. Das übernächste Kapitel wiederum enthält detaillierte Informationen über die CMIS-Implementierung von OpenCms. Bitte beachten Sie, dass dieses Dokument den CMIS-Standard nicht bis ins Detail beschreibt. Dazu lesen Sie bitte die CMIS-Spezifikation unter http://docs.oasis-open.org/cmis/CMIS/v1.0/cmis-spec-v1.0.html.

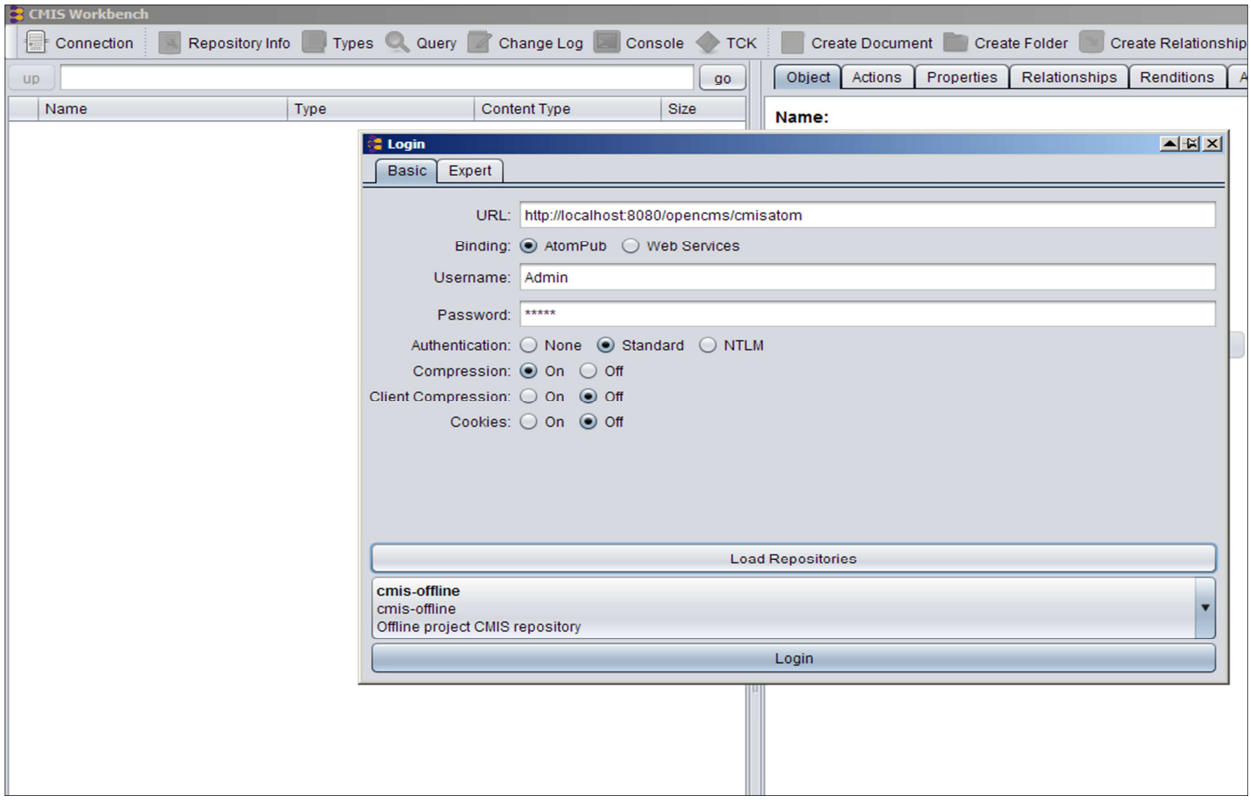


13.2 Zugriff auf OpenCms via CMIS

OpenCms Version 8.5 und höher kommen mit CMIS-Konnektivität Out-of-the-Box. Deswegen ist keine Konfiguration auf der OpenCms-Seite erforderlich. Wir gehen davon aus, dass eine OpenCms-Instanz unter http://localhost:8080/opencms lokal ausgeführt wird, wobei `opencms` der Webapplikationsname ist.

Wir nutzen die Apache Chemistry Workbench für dieses Beispiel. Dies ist eine Java -Client-GUI-Anwendung für den Zugriff auf beliebige CMIS-Repositorys. Sie können es unter http://chemistry.apache.org/java/download.html. herunterladen.

Starten Sie den Apache Chemistry Workbench und klicken Sie auf die Schaltfläche **Verbinden**. Der folgende Dialog öffnet sich:



Der CMIS-Standard definiert sowohl eine AtomPub-Bindung und eine SOAP-Web-Service-Bindung. OpenCms unterstützt beide, aber wir nutzen die AtomPub-Bindung für dieses Beispiel. Darum geben Sie die Verbindungs-URL http://localhost:8080/opencms/cmisatom ein und wählen Sie den **AtomPub**-Radio-Button.
Geben Sie den Benutzernamen und das Kennwort des OpenCms-Benutzers ein, mit dem sie sich einloggen wollen. Alle Operationen die über CMIS durchführt werden, laufen im Kontext des angemeldeten OpenCms-Benutzers. Anschließend klicken Sie auf den **Load Repositories**-Button.

Zwei Repositorys werden angezeigt: `cmis-online`, mit dem Sie auf Inhalte aus dem OpenCms Online-Projekt zugreifen, aber keine Änderungen vornehmen können und `cmis-offline`, das Ihnen den Zugriff auf Inhalte und Änderungen an den Offline-Inhalten ermöglicht. Wählen Sie `cmis-offline` und klicken Sie auf die den **Login**-Button.

Die GUI wird nun die Ressourcen aus dem Root-Ordner des VFS (Virtual File System) anzeigen. Durch einen Doppelklick können Sie in andere Ordner navigieren. Ein Doppelklick auf andere Ressourcen wird die Ressourcen aus dem VFS laden und anzeigen. Bitte beachten Sie, dass Sie mit CMIS nur auf den Rohinhalt der Ressourcen zugreifen können. Genauer gesagt, können Sie keine gerenderte Ausgabe von XML-Inhalten via CMIS erhalten, sondern nur den Quellcode.

Auf der rechten Seite der Workbench, können Sie verschiedene Tabs für den Zugriff auf verschiedene Informationen auswählen und Aktionen für die auf der linken Seite ausgewählten Ressource ausführen. Zum Beispiel, erlaubt Ihnen der Tab **Actions** beispielsweise das Löschen der aktuellen Ressource, während über den Tab **Properties** die Eigenschaften der aktuell ausgewählte Ressource anzeigt und geändert werden können.

13.3 CMIS Integration

CMIS ist ein Standard für den Zugriff auf Content-Repositorys über Web-Services. OpenCms bietet eine CMIS-Schnittstelle, über die CMIS-Client-Software auf das OpenCms-VFS zugreifen kann. Zu diesem Zweck, liefert OpenCms zwei zusätzliche Servlets mit, die in der Datei `WEB-INF/web.xml` konfiguriert sind und für die beiden möglichen Binding-Typen im CMIS-Standard definiert sind: **AtomPub** und **SOAP**. Den Servlets sind die Pfade `/cmisatom` und `/services` zugeordnet.

Darüber hinaus werden einige Servletkontext-Listener in der Datei `web.xml` definiert. Diese Servlets werden durch die Apache-Chemistry-Server-Implementierung bereitgestellt. Wenn Sie CMIS nicht benutzen wollen, kommentieren Sie den CMIS-Bereich in der `web.xml`-Datei, markiert mit "Start of/End of CMIS configuration", aus. Dies reduziert die Startzeit der OpenCms-Webanwendung. Diese Servlets bieten einen Zugriff auf alle CMIS-Repositorys, die in OpenCms in `opencms-importexport.xml` konfiguriert wurden. CMIS-Repositorys sind, wie WebDav-Repositorys unter dem `<repositories>`-Element konfiguriert. Standardmäßig hat diese Konfiguration zwei konfigurierte Repositorys:

```
<repositories>...
<repository name="cmis-offline" class="org.opencms.cmis.CmsCmisRepository">
 <params>
   <param name="project">Offline</param>
   <param name="description">Offline project CMIS repository</param>
   <param name="index">Solr Offline</param>
        </params>
</repository>
<repository name="cmis-online" class="org.opencms.cmis.CmsCmisRepository">
 <params>
   <param name="project">Online</param>
   <param name="description">Online project CMIS repository</param>
   <param name="index">Solr Online</param>
```

```
 </params>
</repository>
...
</repositories>
```

Der `<repositories>`-Abschnitt der Konfiguration wird auch verwendet, um WebDAV-Repositorys für OpenCms zu konfigurieren. OpenCms unterscheidet CMIS-Repositorys von WebDAV-Repositorys durch die Tatsache, dass der Wert des **class**-Attributs auf den Namen einer Java-Klasse verweist, die die Schnittstelle `org.opencms.cmis.I_CmsCmisRepository` implementiert.

Derzeit ist die einzige Klasse, die diese Schnittstelle implementiert und mit OpenCms mitgeliefert wird, die Klasse `org.opencms.cmis.CmsCmisRepository`. Diese Klasse unterstützt verschiedene Konfigurationsparameter, die im **params**-Element definiert werden.

project: Das „Arbeitsprojekt" des CMIS-Repositorys. Dies ist ein erforderlicher Parameter. Wenn das das Online-Projekt ist, bietet das Repository nur Lesezugriff auf das Online-Projekt des VFS.
Wenn dies irgendein anderes Projekt ist, greift das CMIS-Repository auf den Offline-Zustand des VFS zu und alle Änderungen werden im konfigurierten Projekt durchgeführt. Man beachte, dass aus der Sicht des CMIS-Standards, verschiedene Repositorys unabhängige Objekte sind. Dies bedeutet, dass wenn Sie eine Client-Anwendung schreiben, die sowohl auf Offline-Inhalte, als auch auf Online-Inhalte von OpenCms über CMIS zugreifen soll, dass Sie eine Verbindung zu beiden, also zu Offline-und Online-Repository herstellen müssen.

description: Ein Beschreibungstext für das Respository, welcher von CMIS-Clients angezeigt werden kann.

index: Der Name des OpenCms-Solr-Index, der für die Abfrage des Repository verwendet wird. Dies funktioniert mit einem Lucene-Index **nicht**. Dieser verwendet die OpenCms-Standard-Solr Such-Funktionen, die an andere Stelle in diesem Handbuch beschrieben sind. Wenn dieser Parameter weggelassen wird, meldet die CMIS-Implementierung das Repository als "nicht durchsuchbar".

rendition: Der Name einer „Darstellungs-Provider"-Klasse. CMIS verwendet das Konzept der "Darstellungen" (renditions), dies sind alternative Ansichten des gleichen Inhalts, z. B. Miniaturansichten von Bildern. Die Klasse, die hier konfiguriert ist kann solche "renditions" anbieten. Die Klasse muss die Schnittstelle `org.opencms.cmis.I_CmsCmisRenditionProvider` implementieren. Es sind keine Klassenimplementierungen dieser Schnittstelle in der Standard-OpenCms-Distribution enthalten.

property: Der Name einer "Eigenschaften-Provider"-Klasse. Mit diesem Parameter ist es möglich, spezielle Eigenschaften des OpenCms-VFS der CMIS-Ansicht hinzuzufügen, die nicht direkt einer Ressourcen-Eigenschaft oder -Attribut entspricht, sondern stattdessen von aufrufenden Methoden einer Instanz einer zur Verfügung gestellten Klasse gelesen oder geschrieben werden. Sie können dies nutzen, um eigenen Code in OpenCms durch CMIS-Abrufe zur Verfügung zu stellen. Die konfigurierten Klassen müssen die Schnittstelle `org.opencms.cmis.I_CmsPropertyProvider` implementieren. Es sind keine Klassenimplementierungen dieser Schnittstelle in der Standard-OpenCms-Distribution enthalten.

OpenCms stellt Ressourcen und Beziehungen als CMIS-Objekte zur Verfügung. Alle Dateien sind vom CMIS-Typ **cmis:document**. Alle Ordner sind vom CMIS-Typ **cmis:folder**. Verknüpfungen sind vom CMIS-Typ **opencms:RELATIONTYPE**, wobei **RELATIONTYPE** den Beziehungstyp-Namen in OpenCms definiert.

Dateien und Ordner können durch CMIS erstellt, gelöscht und geändert werden, vorausgesetzt, das Projekt-Repository ist nicht das Online-Projekt und der aktuelle Benutzer hat die Berechtigungen zum Ausführen der Operationen. Verknüpfungen können nur erstellt oder gelöscht werden, wenn der Beziehungs-Typ nicht markiert ist als "defined in content" (Siehe die Methode `org.opencms.relation.CmsRelationType.isDefinedInContent`).

Für jede OpenCms-Eigenschaft "X" gibt es zwei CMIS-Eigenschaften, **cmis:document** und **cmis:folder**: **opencms:X** und **opencms-inherited:X** verfügbar sind. Die **opencms:X**-Eigenschaft für einen CMIS-Dokument oder -Ordner enthält den Wert der Eigenschaft für die Ressource, die direkt auf die Ressource gesetzt ist, während die **opencms-inherited:X**-Eigenschaft den Wert hat, der entweder direkt auf die Ressource gesetzt oder von einem übergeordneten Ordner vererbt wurde. Die **opencms:X**-Eigenschaft kann sowohl gelesen, als auch geschrieben werden. Die **opencms-inherited:X**-Eigenschaft dagegen ist schreibgeschützt. Das Schreiben der **opencms:X**-Eigenschaft mit CMIS setzt sowohl den Struktur-, als auch den Ressourcen-Wert der Eigenschaft in OpenCms.

Einige standardmäßige CMIS-Eigenschaften sind mit Ressourcen-Attributen oder anderen nützlichen Ressourcen-Informationen von OpenCms befüllt:

cmis:objectId - die interne OpenCms-Struktur-Id der Ressource

cmis:path – der Root Pfad der Ressource

cmis:lastModifiedBy – der Name des Benutzers, der die Ressource zuletzt geändert hat

cmis:lastModificationDate – das Datum, an welchem die Ressource zuletzt geändert wurde

cmis:createdBy – der Name des Benutzers, der eine Ressource erstellt

cmis:creationDate – das Datum, an dem die Ressource erstellt wurde

opencms-special: resource-type – Enthält den OpenCms-Ressourcentypnamen

opencms-dynamic:PROPERTYNAME - Eine dynamische Eigenschaft, die durch eine benutzerdefinierte Providerklasse, die in der Datei `opencms-importexport.xml` konfiguriert ist, behandelt wird (siehe Beschreibung weiter oben).

Es ist nicht möglich durch CMIS völlig neue OpenCms-Eigenschaften zu erstellen. Sie können neue Eigenschaften nur in OpenCms definieren. Diese neuen Eigenschaften sind möglicherweise durch die CMIS-Schnittstelle nicht sofort sichtbar, da die Liste der gültigen OpenCms-Eigenschaften in der CMIS-Implementierung nur alle 5 Minuten aktualisiert werden.

Beim Hochladen neuer Dateien in OpenCms durch CMIS wird der OpenCms-Ressourcentyp automatisch aus der Datei-Endung der hochgeladenen Datei bestimmt. Sie können dies durch die Spezifizierung der **opencms-special:resource-type**-Eigenschaft mit dem OpenCms-Ressourcentyp als Wert, beim Erstellen eines neuen Dokumentes mit CMIS, überschreiben.

13.3.1 Begrenzung der CMIS Implementierung

Die CMIS-Implementierung von OpenCms unterstützt nicht alle optionalen CMIS-Features und nicht alle OpenCms-Funktionalitäten. Diese nicht unterstützten Features sind:

- Versionierung und PWCs
- unabgelegte und mehrfach abgelegte Ressourcen
- Ändern der Berechtigungen / Zugriffskontrolleinträge
- Changelogs
- Richtlinien

- CMIS-SQL-Support

Obwohl Abfragen CMIS-SQL nicht unterstützen, sind Abfragen immer noch mit Solr-Abfragen möglich. Wenn ein Solr-Index wie oben beschrieben für das Repository konfiguriert ist, wird OpenCms berichten, dass das Repository abfragbar ist. Standard-CMIS-Clients, die sich auf CMIS-SQL verlassen, die für abfragbare Repositorys zur Verfügung stehen, könnten nicht funktionieren.

Veröffentlichen von Dateien wird nicht unterstützt, da der CMIS-Standard kein Konzept für den "Offline"- und "Online"-Modus wie in OpenCms hat. Die Offline-und Online-Repositorys in der Standard-Konfiguration sind aus CMIS-Sicht völlig unabhängig voneinander. Obwohl die CMIS-Implementierung einfache Zugriffskontrollmöglichkeiten bietet, sind diese im Vergleich zum Berechtigungssystem von OpenCms sehr unvollständig. Sie sind nicht ausreichend, um zu bestimmen, ob ein Benutzer eine Aktion durch CMIS durchführen darf. Also, wenn Sie eine Client-Anwendung schreiben, um auf OpenCms durch CMIS zuzugreifen, müssen Sie die "allowable actions"-API aufrufen, um zu bestimmen, ob eine Operation durchgeführt werden kann, bevor Sie sie tatsächlich ausführen.

13.3.2 CMIS-Client-Programm-Beispiel

Dank CMIS, können Sie eine beliebige Client-Bibliothek nutzen, um eine benutzerdefinierte Client-Anwendung zu schreiben, die auf das VFS zugreift. Eine Beispiel-Anwendung, die die Apache Chemistry Client-Bibliotheken verwendet wird unten angehängt. Diese Beispiel Anwendung zeigt das Hochladen von Dokumenten aus einem Ordner im lokalen Dateisystem in das RFS. Sie erfahren mehr über die Entwicklung von Client-Anwendungen mit Apache Chemistry in Java auf: http://chemistry.apache.org/java/developing/index.html

```
/**
* Demo for uploading files to CMIS.
*/
public class CmisUploader {

  /** The repository id. */
  public static final String REPOSITORY_ID = "cmis-offline";

  /** The session factory. */
  private static SessionFactory sessionFactory = SessionFactoryImpl.newInstance();

  private String m_repositoryUrl;

  /** The session. */
  private Session m_session = null;

  public CmisUploader(String repositoryUrl) {

      m_repositoryUrl = repositoryUrl;
  }

  /**
   * Main method.
   */
  public static void main(String[] args) throws Exception {

      String repositoryUrl = args[0];
      String source = args[1];
      String target = args[2];
      CmisUploader uploader = new CmisUploader(repositoryUrl);
      uploader.copyFiles(source, target);
  }

  /**
   * Copies files from a local folder to a CMIS folder.
   */
  public void copyFiles(String sourceFolder, String targetFolder) throws Exception {
```

```
        File sourcefolder = new File(sourceFolder);
        for (File child : sourcefolder.listFiles()) {
            if (child.isFile()) {
                uploadFileToCmis(child.getAbsolutePath(), targetFolder);
            }
        }
    }

    /**
     * Uploads a single file from the local file system to a CMIS folder.
     */
    public void uploadFileToCmis(String fileSystemPath, String targetFolder) throws Exception {

        Session session = getSession();
        File file = new File(fileSystemPath);
        String name = file.getName();
        byte[] fileContent = readFile(file);
        Folder parent = (Folder)(session.getObjectByPath(targetFolder));
        String targetPath = (targetFolder + "/" + name).replaceAll("/+", "/");
        try {
            session.getObjectByPath(targetPath);
            System.out.println("Not creating " + targetPath + " since it already exists. ");
        } catch (CmisObjectNotFoundException e) {
            System.out.println("Creating " + targetPath);
            Map<String, Object> properties = new HashMap<String, Object>();
            properties.put(PropertyIds.OBJECT_TYPE_ID, "cmis:document");
            properties.put(PropertyIds.NAME, name);
            ContentStream contentStream = new ContentStreamImpl(
                name,
                BigInteger.valueOf(fileContent.length),
                "binary/octet-stream",
                new ByteArrayInputStream(fileContent));
            parent.createDocument(properties, contentStream, VersioningState.MAJOR);
        }
    }

    /**
     * Gets the session, creating it if possible.
     */
    private Session getSession() {

        if (m_session == null) {
            Map<String, String> parameters = new HashMap<String, String>();
            // OpenCms user name and password
            parameters.put(SessionParameter.USER, "Admin");
            parameters.put(SessionParameter.PASSWORD, "admin");
            parameters.put(SessionParameter.ATOMPUB_URL, m_repositoryUrl);
            parameters.put(SessionParameter.BINDING_TYPE, BindingType.ATOMPUB.value());
            parameters.put(SessionParameter.REPOSITORY_ID, "cmis-offline");
            m_session = sessionFactory.createSession(parameters);
        }
        return m_session;
    }

    /**
     * Utility method for reading a file's content.
     */
    private byte[] readFile(File file) throws IOException {

        byte[] fileContent = new byte[(int)file.length()];
        FileInputStream istream = new FileInputStream(file);
        istream.read(fileContent);
        istream.close();
        return fileContent;
    }
}
```

14 JSP Grundlagen

Dieser Teil der Dokumentation beschreibt die Grundlagen der JSP-Anwendungsentwicklung mit OpenCms. Es bietet einen Überblick und einige Hintergrundinformationen über die Technologie

die für die OpenCms-JSP-Integration verwendet wird. Andere Abschnitte in der Alkacon OpenCms Dokumentation begleiten diese Einführung und dienen als weiteres Referenzmaterial. Bitte beachten Sie, dass dies keine Einführung in die JSP-Entwicklung ist. Dafür gibt es viele andere gute Tutorials oder Bücher.

14.1 JSP Funktionen

- Hinzufügen und Verwalten von JSPs im OpenCms Workplace
- Unterstützt WYSIWYG-editierbare Seiten, die JSP-Templates verwenden
- Verwende das gleiche JSP-Template für editierbare Seiten und interaktive Formulare
- JSP-Taglib für allgemeine OpenCms-Aufgaben
- JSP-API, für direkten Zugriff auf OpenCms-Funktionalitäten
- Separate Online- und Offline-Versionen der gleichen JSP
- Der FlexCache, ein mächtiger deklarativer Caching-Mechanismus
- Optionales Output-Streaming pro Seite
- Optionaler statischer Export von JSPs

14.2 Die JSP <cms>-Taglib

14.2.1 Wie wird die "taglib"-Direktive eingefügt?

Wenn Sie die OpenCms JSP-Taglib nutzen möchten, müssen Sie die `taglib`-Direktive in ihrer JSP-Seite einfügen, um festzulegen, wo die OpenCms-Taglib-Definition gefunden werden kann. Dies kann auf zwei verschiedene Arten erfolgen:

1. Mit Standard JSP-/Servlet-Technology:

```
<%@ taglib prefix="cms" uri="http://www.opencms.org/taglib/cms" %>
```

Diese Richtlinie muss gesetzt werden bevor irgendwelche `cms`-Tags verwendet werden. Wenn Sie, wie im Beispiel gezeigt, verwendet wird, ist das Präfix "`cms`" eindeutig für die OpenCms-Taglib. In unseren Beispielen beginnen alle Tags der OpenCms-Taglib mit dem Präfix "`cms:`".

2. Mit der OpenCms-spezifischen Kurzform:

```
<%@page buffer="none" session="false" taglibs="c,cms" %>
```

14.2.2 Verfügbare Tags

Die folgende Liste von <cms>-Tags sind in der OpenCms-Standard-Taglib enthalten. Ein Tag kann durch folgenden Aufruf ausgeführt werden:

```
<cms:tagName attributes...></cms:tagName>
```

- **cms:container** – Ermöglicht den Template-Mechanismus für Containerseiten.
- **cms:contentaccess** – Ermöglicht den Zugriff auf den Inhalt über JSP EL.
- **cms:contentcheck** – Bietet Bedingungs-Logik zur Überprüfung der Elemente von XML-Inhalten.
- **cms:contentinfo** – Ermöglicht den Zugriff auf die Ergebnisliste eines „contentload“.

- **cms:contentload** – Lädt XML-Content-Elemente aus den OpenCms-VFS.
- **cms:contentloop** – Erlaubt das Durchlaufen von XML-Inhaltsknotenelement-Werten in einer Schleife.
- **cms:contentshow** – Ermöglicht den Zugriff auf einzelne XML-Inhaltsknotenelement-Werte.
- **cms:decorate** – Bietet Gestaltung von HTML-Inhalten.
- **cms:device** – Bietet verschiedene HTML-Ausgaben für verschiedene Gerätetypen
- **cms:editable** – Aktiviert die direkte Inhaltsbearbeitung in einem Template.
- **cms:elementsetting** – Bietet Zugriff auf die Einstellungen eines ADE-Container-Elements.
- **cms:enable-ade** – Aktiviert die erweiterte direkte Bearbeitung in einem Template.
- **cms:export** – Schreibt JSP-Code aus JSP-Dateien in den statischen Export.
- **cms:formatter** – Lädt einzelne XML-Content-Elemente für ADE-Formatter.
- **cms:headincludes** – Bindet erforderliche CSS- oder Javascript-Ressourcen im html/head ein.
- **cms:img** – Unterstützt Bilderskalierung mit dem OpenCms-nativen Bildskalierungs-mechanismus.
- **cms:include** – Bindet andere JSP-Elemente dynamisch zur Laufzeit ein.
- **cms:info** – Ermöglicht den Zugriff auf System-Informationen, wie der OpenCms-Version usw.
- **cms:jquery** – Ermöglicht JQuery- und einige JQuery-Plugins-Codes und Stylesheets einzubinden.
- **cms:label** – Liest lokalisierte Werte von den Resource-Bundles (Java Property-Dateien wie workplace.properties), die Benutzerinformationen enthalten.
- **cms:link** – erstellt eine gültige OpenCms-URL für eine bestimmte Ressource.
- **cms:navigation** – Bietet Zugriff auf die Navigations-Informationen.
- **cms:nocache** – Sendet no-cache-Header an den Browser.
- **cms:param** – Fügt einen Parameter in den umgebenden Tag ein (falls unterstützt).
- **cms:parse** – gestaltet HTML mit benutzerdefinierten `A_CmsConfiguredHtmlParser`-Implementierungen die in dem `parserClass`-Attribut mitgegeben werden.
- **cms:property** – Ermöglicht den Lesezugriff auf die Eigenschaften der aktuellen VFS-Datei.
- **cms:resourceaccess** – Bietet Zugriff auf die Ressourcen über JSP EL.
- **cms:resourceload** – Lädt Ressourcen aus dem OpenCms-VFS.
- **cms:secureparams** – Aktiviert das automatische Parameter-Escaping für den aktuellen Flex-Request.
- **cms:template** – Teilt eine JSP-Seite in Elemente, die mit dem `'cms:include'`-Tag inkludiert werden können.
-

- **cms:user** – Ermöglicht den Zugriff auf die Eigenschaften des aktuell angemeldeten Benutzers.
- **cms:usertracking** – Führt Benutzer Tracking Aktionen wie besuchte Ressourcen oder anmelden/abmelden.

14.2.3 Verfügbare EL-Funktionen

Die folgende Liste von EL-Funktionen finden Sie in der OpenCms-Standard-Taglib. Eine EL-Funktion kann so aufgerufen werden:

```
${cms:functionName(parameters)}
```

- **cms:vfs** – Bietet einfachen Zugriff auf eine OpenCms JSP/EL-Content-VFS-Access-Bean.
- **cms:convertDate** – Ermöglicht die Konvertierung von Long-Werten in Datumswerte. Kann auch Strings behandeln, die ein Long oder ein Datum repräsentieren.
- **cms:convertLocale** – Ermöglicht die Konvertierung von Objekten nach Locales. Kann auch Strings behandeln, die Locales sind, oder Locales selber.
- **cms:convertList** – Gibt eine Liste von Attributwerten zurück, die durch den Attributnamen der Elemente der Liste spezifiziert werden.
- **cms:getCmsObject** – Gibt den aktuellen OpenCms-Benutzer-Kontext aus dem Page-Kontext zurück.
- **cms:getListSize** – Gibt die Größe der Liste zurück.
- **cms:striphtml** – Entfernt alle HTML-Markups aus der Eingabe.
- **cms:TrimToSize** – Gibt einen Substring der Eingabe zurück, der nicht länger ist als der angegebene int-Wert.
- **cms:convertUUID** – Ermöglicht die Konvertierung von String-Werten zu CmsUUIDs. Kann auch byte[] behandeln, die CmsUUIDs sind, oder CmsUUIDs selber.
- **cms:getRequestParam** – Gibt den Wert eines Parameters aus einem String, der für eine GET-Anforderung formatiert ist.
- **cms:getRequestLink** – Liefert den Link ohne Parameter aus einem String, der für eine GET-Anforderung formatiert ist, zurück.
- **cms:escape** – Kodiert einen String in einer Weise, die kompatibel mit der Javascript-escape-Funktion ist.
- **cms:unescape** – Decodiert einen String in einer Weise, die kompatibel mit der Javascript-unescape-Funktion ist.
- **cms:navUri** – Liefert die aktuelle Navigation URI zurück.

15 Impressum

Alkacon Software GmbH

An der Wachsfabrik 13
DE-50996 Köln
DEUTSCHLAND

Telefon: + 49 2236 3826 - 0
Fax: + 49 2236 3826 - 20

Geschäftsführer: Alexander Kandzior

Amtsgericht Köln
HRB 54613
Umsatzsteuer-ID: DE 259882372

---

comundus GmbH

Heerstr. 111
71332 Waiblingen
DEUTSCHLAND

Telefon: +49 7151 96528-0
Fax : +49 7151 96528-999

Geschäftsführer: Klaus Hillemeier

Amtsgericht Stuttgart,
HRB 264290
Umsatzsteuer-ID: DE 213460346