Использование поиска

# Подключение Solr

Платформа ECMp5 предполагает 2 варианта подключения поискового процессора Solr: внешний и встроенный.

Первый вариант (внешний Solr) предназначен и рекомендован для промышленной эксплуатации системы. Solr при этом устанавливается силами администратора как отдельное приложение либо на собственный сервер (лучше), либо на один сервер с приложением на платформе ECMp5. Платформа осуществляет взаимодействие с сервером Solr по протоколу HTTP.

Во втором варианте библиотеки Solr включаются в одно приложение (war или ear файл) вместе с платформой ECMp5. Подсистема поиска взаимодействует с Solr непосредственно через Java calls. Этот вариант удобен для локальной установки приложения разработчиками, поскольку не требует установки и настройки дополнительного ПО. Однако, для промышленной эксплуатации данный вариант не рекомендуется по соображениям производительности и надёжности.

Скрипты сборки системы поддерживают оба варианта подключения Solr. Выбор варианта зависит от наличия системного свойства (system property) solr.external. Если задать это свойство при сборке проекта (например, запустить maven с параметром -Dsolr.external=true), то будет собрано приложение, требующее наличия внешнего поискового сервера. При отсутствии этого свойства (вариант по умолчанию) в war-файл приложения включаются библиотеки, содержащие реализацию Solr (общий объём – более 50 МБ). При этом сам архив приложения получит имя cm-sochi-solr-<номер\_версии>.war.

# Настройка Solr

## Внешний Solr

Для работы с внешним сервером Solr необходимо указать его базовый URL в свойстве search.solr.url (файл server.properties). Необходимо отметить, что указание этого свойства приведёт к использованию внешнего сервера Solr, даже если приложение было собрано со встроенным Solr. Если же приложение не содержит встроенный Solr, а свойство не указано, приложение не будет запускаться.

Настройка внешнего сервера Solr для использования платформой ECMp5 будет описана позже.

## Встроенный Solr

Встроенный Solr-сервер может использоваться без какой-либо дополнительной настройки. Однако, подсистема поиска предусматривает возможность настройки некоторых параметров, что может быть необходимым в некоторых случаях. Все эти параметры задаются в файле server.properties.

* search.solr.home – базовый каталог для Solr. Этот каталог может (но не обязан) содержать файл solr.xml, задающий основные настройки Solr. (Формат файла solr.xml описан в документации Solr.) Также в этом каталоге при запуске Solr создаёт подкаталог solr/<имя\_ядра> (описано ниже), в котором он будет размещать свою конфигурацию, управляемую подсистемой поиска, и (по умолчанию) поисковые индексы. Если свойство не задано, используется текущий каталог сервера приложений. Будьте внимательны! Текущий каталог может зависеть от способа запуска сервера приложений (например, из среды разработки или из командного файла). Если ранее созданные файлы конфигурации или поисковые индексы недоступны Solr, могут возникать ошибки поиска, или выдаваться некорректные результаты.
* search.solr.data – каталог для хранения поисковых индексов. Каталог изначально должен быть пустым и позволять создание файлов и подкаталогов пользователю, от имени которого запускается сервер приложений JBoss. Если свойство не задано, поисковые индексы создаются в каталоге <search.solr.home>/solr/<имя\_ядра>/data.
* search.solr.collection – имя поискового ядра Solr, используемого подсистемой поиска платформы. Значение по умолчанию – collection1. Возможность изменить имя ядра может быть полезна, если вы хотите использовать один экземпляр Solr для различных целей (приложений), или просто сделать более удобным путь к рабочим файлам Solr (т.к. этот путь содержит имя ядра – см. выше).

Поисковые индексы должны находиться в соответствии с данными, хранящимися в системе – доменными объектами и файлами вложений. При создании и изменении объектов средствами системы подсистема поиска выполняет их индексацию автоматически. Если вы производите очистку БД вручную, удалите и поисковые индексы Solr (содержимое каталога <search.solr.data> полностью). При ручном изменении отдельных объектов в БД, а также при утере или порче поисковых индексов по какой-либо причине можно выполнить задачу полной или частичной переиндексации.

# Конфигурация поиска

## Области поиска

Подсистема поиска всегда осуществляет поиск только в предварительно проиндексированных объектах. Набор объектов, а также индексируемые поля настраиваются через определения областей поиска. Простой поиск ищет запрошенную строку во всех полях объектов в одной области поиска, расширенный позволяет задавать отдельные условия для каждого поля, а также выбирать несколько областей поиска.

Область поиска задаёт набор условий, которым должен удовлетворять объект, чтобы попасть в неё (быть проиндексированным). Объекты в одной области поиска всегда разделяются по типам. Для каждого типа объекта в области поиска задаётся свой набор индексируемых полей.

У объекта могут индексироваться не только его собственные поля, но и поля связанных объектов (включая связанные косвенно). Например, можно искать документ по полям выданных по нему (связанных) поручений, или искать по фамилии автора (поле объекта из справочника персон). Этим двум примерам соответствуют два сценария индексирования. Первый заключается в индексировании самих связанных объектов, с указанием набора их полей и способа связи с исходным объектом. Этот сценарий предназначен для часто изменяемых объектов, каковыми являются объекты поручений. Второй сценарий предполагает индексирование полей связанных объектов непосредственно в исходном объекте. При этом изменение связанного объекта не приводит к обновлению индекса всех зависящих от него объектов, и такое обновление (переиндексация) должно быть выполнено отдельной задачей. Таким образом, этот сценарий более применим для справочников, т.к. они изменяются достаточно редко, но имеют множество ссылок на них, т.е. переиндексация зависимых объектов может оказаться весьма ресурсоёмкой задачей. Поэтому такую задачу целесообразно выполнить в отложенном режиме в период низкой загрузки сервера.

Результат поиска всегда содержит однотипные объекты, и их тип определяется конфигурацией коллекции, запрошенной для возврата этого результата. При расширенном поиске все запрошенные области должны поддерживать этот общий тип.

Один и тот же объект может попадать в несколько разных областей поиска, если он удовлетворяет условиям отбора каждой из них.

### Пример конфигурации области поиска

<search-area name=*"Сотрудники"*>

<target-domain-object type=*"Employee"*>

<indexed-field name=*"FirstName"*/>

<indexed-field name=*"LastName"*/>

<indexed-field name=*"EMail"*/>

<indexed-field name=*"Position"*/>

<indexed-field name=*"Department"*/>

<indexed-field name=*"Department\_Name"*>

<doel>Department.Name</doel>

</indexed-field>

<indexed-field name=*"Organization"*>

<doel>Department.Organization</doel>

</indexed-field>

<indexed-field name=*"Organization\_Name"*>

<doel>Department.Organization.Name</doel>

</indexed-field>

<target-collection name=*"PersonByLogin"*/>

</target-domain-object>

<target-domain-object type=*"Department"*>

<indexed-field name=*"Name"*/>

<indexed-field name=*"Organization"/*>

<indexed-field name=*"Organization\_Name"*>

<doel>Organization.Name</doel>

</indexed-field>

<indexed-field name=*"ParentDepartment"*>

<doel>ParentDepartment</doel>

</indexed-field>

<indexed-field name=*"ParentDepartment\_Name"*>

<doel>ParentDepartment.Name</doel>

</indexed-field>

<linked-domain-object type=*"Employee"*>

<indexed-field name=*"FirstName"*/>

<indexed-field name=*"LastName"*/>

<parent-link>

<doel>Department^Boss</doel>

</parent-link>

</linked-domain-object>

<target-collection name=*"Departments"*/>

</target-domain-object>

</search-area>

Приведённая конфигурация указывает, что в области поиска «Сотрудники» можно искать объекты двух типов: Employee и Department. У объекта Employee индексируются не только его собственные поля, но и некоторые поля связанного объекта Department (Name и Organization), и даже одно поле косвенно связанного объекта Organization (Name). Таким образом, поиск сотрудников по имени подразделения найдёт всех сотрудников этого подразделения, а по имени организации – всех сотрудников всех подразделений организации. Видно, что для полей связанных объектов используется тег <doel>, указывающий DOEL-выражение для получения значения этого поля из исходного объекта. Следует отметить, что этот тег может применяться и для собственных полей объекта, если необходимо индексировать поле под другим именем. Например:

<indexed-field name=*"FullName"*>

<doel>Name</doel>

</indexed-field>

Ещё один момент, требующий разъяснения, это индексирование полей связей (Department и Organization). В отличие от названий подразделения и организации, поиск которых осуществляется по тексту и полезен преимущественно при простом поиске (ввели в строку поиска часть названия организации – получили всех её сотрудников), индексы полей связей более полезны для расширенного поиска, где удобнее выбрать организацию из списка (справочника), а в поисковый запрос передать идентификатор этого объекта. Также нужно отметить, что способ индексации полей подразделения и организации выбран из предположения о том, что эти объекты (или, по крайней мере, их поля Name) изменяются весьма редко.

Во втором целевом типе объекта (Department) используется другой способ индексации полей связанных объектов (Employee).

## Задачи переиндексации

Задачи переиндексации находятся в стадии реализации и будут описаны позже.