Platform, Services, and Utilities

Generated on: 2024-11-08 15:20:25 GMT+0000

SAP Commerce | 2205

PUBLIC

Original content: https://help.sap.com/docs/SAP_COMMERCE/d0224eca81e249cb821f2cdf45a82ace?locale=en-US&state=PRODUCTION&version=2205

Warning

This document has been generated from the SAP Help Portal and is an incomplete version of the official SAP product documentation. The information included in custom documentation may not reflect the arrangement of topics in the SAP Help Portal, and may be missing important aspects and/or correlations to other topics. For this reason, it is not for productive use.

For more information, please visit the https://help.sap.com/docs/disclaimer.

API Registry Module

The API Registry module contains generic functions for storing event configurations, endpoint configurations, and credentials. In addition, it defines how to expose events and destinations to a target system.

Features

Connecting with External Systems

Destination Targets

Endpoints and Destinations

Credentials

Events

Charon Client Integration

Architecture

API Registry Module Architecture

apiregistryservices Extension

apiregistrybackoffice Extension

Implementation

Migrating from Yaas Configuration to API Registry

Configure Consumed Destination Connection Test

Configuring the Standard Host Address for Exposed Destinations

Charon Client Sample Implementation

API Registry Module Features

The API Registry module provides a range of features related to managing event configurations, endpoint configurations, and credentials, as well as exposing events and destinations to a target system.

Connecting with External Systems

You can connect to external systems in order to consume services provided by those systems or expose destinations to allow other services to consume services in SAP Commerce.

Destination Targets

This feature allows you to create destination targets, which act as containers for your events and destinations.

Endpoints and Destinations

This feature allows you to configure the URLs that allow you to connect to external systems or to that allow external systems to connect to SAP Commerce.

Credentials

This feature allows you to manage stored credentials that you assign to exposed or consumed destinations.

Events

This feature allows you to define events and activate or deactivate event exporting.

Charon Client Integration

This feature allows you to integrate your application with the Charon client, which provides a framework for composing asynchronous HTTP transactions.

Connecting with External Systems

You can connect to external systems in order to consume services provided by those systems or expose destinations to allow other services to consume services in SAP Commerce.

Establishing Inbound Connections

You connect external systems to your SAP Commerce systems to allow the external systems to consume services offered by SAP Commerce.

Procedure

1. Configure a destination target.

   Destination targets enable you to group related destinations and events.

2. Configure the endpoints.

   Endpoints are generic destinations. You specify a spec URL or upload spec data.

3. Configure the credentials.

   You assign credentials to the destination to enable external systems to communicate with your SAP Commerce system. You can create different types of credentials depending on the authorization concept that you want to use for your system.

4. Configure the exposed destinations.

   Destinations are specific instances of an endpoint. They provide an address for your external system to direct to.

5. Register the destinations.

6. Connect your external system to one of the destinations you set up.

Results

The external systems are connected to your SAP Commerce system and they can begin consuming services offered by the SAP Commerce system.

Establishing Outbound Connections

You connect SAP Commerce to an external system to allow SAP Commerce to consume the services that the external system provides.

Procedure

1. Configure a destination target.

   Destination targets enable you to group related destinations and events.

2. Configure the endpoints.

   Endpoints are generic destinations. You specify a spec URL or upload spec data.

3. Configure the credentials.

   Assign credentials to destinations to enable your SAP Commerce system to communicate with target systems. You create different types of credentials depending on the authorization concept for the target API or system.

4. Configure the consumed destinations.

   Destinations are specific instances of an endpoint. They point to a specific target system and you assign credentials to them to enable them to communicate with the target system.

Results

Your SAP Commerce system is connected to the external system and you can begin calling the services offered by those systems. To test the connection, in Backoffice, go to **System** > **API** > **Destinations** > **Consumed Destinations**, and click **Test Connection**.

Next Steps

If necessary, you can define events that SAP Commerce exports to the target system.

Destination Targets

This feature allows you to create destination targets, which act as containers for your events and destinations.

Use Case

An administrator wants to connect SAP Commerce to an external system.

Features

Destination Target Configuration

You define destination targets. They act as a container for events and destinations. The destination target also specifies the destination channel. The destination channel defines the type of system that the destinations connect to, for example, SAP BTP Extensions or SAP Commerce Cloud, Context-Driven Services.

When you delete a destination target from Backoffice, any related entities, for example, consumed destination, endpoint, credential, etc., that are being consumed by other destination targets, either directly or indirectly, are not deleted. This behavior applies only to Backoffice and neither ImpEx nor Meta API. For example, there is a destination target (DT_a) with a consumed destination (CD_a) with endpoint (EP_a), and a second destination target (DT_b) with a consumed destination (CD_b) using the same endpoint (EP_a). When DT_a is deleted, it cascade deletes CD_a, but EP_a is not deleted since it is still being consumed by CD_b, which is a valid consumed destination.

Registration Status

For some destination channels, you can see the registration status. If the selected destination channel doesn't support this feature, in the editor panel of the **Destination Targets** view, the **Registration Status** field is blank.

Destination Target Configuration

You create destination targets in Backoffice.

Classes

The following classes are available:

Destination Target

Destination Target

Attribute Name	Type	Mandatory	Description
`id`	String	x	ID of the `DestinationTarget`
`destinationChannel`	DestinationChannel	-	Reference to the destination channel. The destination channel is the type of system that the destinations in the destination target are connected to, for example, `KYMA` or `DEFAULT`. For more information about the enum values, see the documentation for the solution that you want to integrate with. If you create a destination channel in Backoffice, you do not have to enter a destination channel, but if you do not, you cannot set one or change it later.
`template`	Boolean	-	Defines whether the destination target is a template. For more information about templates, see the relevant step in Configuring Destination Targets. To define a destination that is not a template, enter the value `NULL`. In Backoffice, you can create a destination target with an empty destination flag.

Attribute Name	Type	Mandatory	Description
`eventConfigurations`	Collection<EventConfiguration>	-	All event configurations associated with the `DestinationTarget`
`destinations`	Collection<AbstractDestination>	-	All destinations associated with the `DestinationTarget`
`registrationStatus`	RegistrationStatus	-	Shows the status of all Extension Factory destinations. The possible values are `STARTED`, `IN_PROGRESS`, `REGISTERED`, and `ERROR`. For more information about these statuses, see Registration Statuses.
`registrationStatusInfo`	String	-	If an error occurs, this field contains a detailed description of the registration status.

Configuring Destination Targets

You create destination targets and assign a destination channel, destinations, and event configurations.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to **System** > **API** > **Destination Targets**.
3. Click *(Add)*.
4. Enter a unique ID for the destination target.
5. Select whether the destination target is a template.
6. Set the destination channel to **Default**.
7. Click **Save**.

   In the list view, the destination target is visible. If it is not, refresh your browser.
8. In the list view, select the target destination.
9. Add the destinations and event configurations that are relevant for the target destination.

> **Note**
>
> You can also add destinations and event configurations to a target destination in the **Destinations** and **Event Configurations** view.

Registration Statuses

For some destination channels, you can see the registration status. The following tables list the available statuses for each destination channel.

SAP BTP, Kyma runtime

Registration Statuses for SAP BTP Extensions

Status	Description
No status available	The system has not requested a certificate or the destination channel is not using the Kyma runtime channel.
`STARTED`	The system has requested a certificate and is waiting for a reply.
`IN_PROGRESS`	The certificate has been retrieved and the Kyma runtime cluster is being provisioned.

Status	Description
REGISTERED	The Kyma runtime cluster is provisioned.
ERROR	There is an issue with certificate retrieval, cluster provisioning, or certificate renewal. You can find details of the error in Backoffice, in the **Destination Targets** view, in the editor area.

Endpoints and Destinations

This feature allows you to configure the URLs that allow you to connect to external systems or to that allow external systems to connect to SAP Commerce.

Use Case

A customer wants to configure an endpoint and expose a destination or define a consumed destination for it.

An endpoint is the generic definition of a web service and provides the metadata for the services by specifying a specification for it. A destination defines a specific instance of the web service. A destination allows you to define how a web service is used by a certain consumer, including how the web service is exposed and which credentials must be used to access it.

The following destination types are available:

- Exposed destinations

  These destinations expose web services provided by SAP Commerce.

- Consumed destinations

  These destinations are for web services in external systems.

Features

Standard Host Address for Inbound Connections

SAP provides a parameter that you can use to change the host address that you use for exposed destinations and their endpoints. This parameter allows you to change multiple URLs without having to change each host address separately. Reregister the exposed destination URLs for the changes to take effect.

Endpoint Configuration

The details of both the exposed SAP Commerce webservices and the external webservices the platform can access are stored in a configuration table. This table is used as a source when exporting a web API specification to a target system.

Consumed Destination Configuration

You can configure a consumed destination. You can test the connection to existing consumed destinations.

Exposed Destination Registration

You can register and deregister exposed destinations.

Endpoint Configuration

The details of both the exposed SAP Commerce web services and the external web services the platform can access, are stored in a configuration table which is used as a source when exporting the web API specification to a target system.

You can check the available endpoints and create endpoints in Backoffice under **System** > **API** > **Endpoint**.

Classes

The following classes are available:

Endpoint Configuration

EndpointConfiguration

Attribute Name	Type	Mandatory	Description
`id`	String	x	Unique identifier of the endpoint together with version
`version`	String	x	The version of the endpoint
`name`	String	x	The full name of an endpoint for display purposes
`description`	String	-	The full description of an endpoint
`specUrl`	String	-	The URL of the endpoint swagger (OpenAPI) specification documentation. For example: `https://electronics.local:9002/assistedservicewebservices/v2/ap docs`
`specData`	String	-	If `specData` provides the full spec in a JSON format that conforms to the OpenAPI specification, it overrides the `specUrl`.
`extensionName`	String	-	A technical value. The name of the SAP Commerce extension that the `Endpoint` belong to. The system uses this value to ensure that the `Endpoint` is valid, by checking that th extension is installed.
`destinations`	Collection<AbstractDestination>	-	All `abstract destinations` associated with the `Endpoint`

Configuring Endpoints

You configure an endpoint for your destination. An endpoint is the generic definition of a web service and provides the metadata for the services by specifying a specification for it.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to **System** > **API** > **Endpoints**.
3. Click **+**.
4. Enter a unique identifier, name, and version for your endpoint.
5. (Optional) Enter a description for your endpoint.
6. Complete one of the following fields:
   - **SpecURL**: Enter the URL that contains your endpoint specification. If you enter any value in the **SpecData** field, the endpoint does not use this URL.
   - **SpecData**: Enter the specification directly in this field. If you enter any value in this field, the endpoint does not use the URL specified in the **SpecURL** field.
7. (Optional) Enter the name of the extension that the web services is located in.
8. Click **Save**.

Next Steps

Configure credentials and destinations.

Destination Configuration

You can create destination configurations in Backoffice Administration Cockpit.

ℹ Note

SAP recommends using exposed destinations for inbound connections and consumed destinations for outbound connections. For more information, see Connecting with External Systems.

Classes

The following classes are available:

Abstract Destination

Destination

Attribute Name	Type	Mandatory	Description
`id`	String	x	ID of the destination
`active`	Boolean	x	Determines whether the destination is active. It can be used in the following situations: • Exposed Destination If this value is set to `true`, APIs are exposed to the target system. This option does not activate or deactivate the services. If you deactivate this option, your services continue to run and are accessible. • Consumed Destination You deactivate this option if the endpoint is not available for a certain amount of time, and SAP can turn it off. Therefore, the system recognizes that the consumed destination is not available and it doesn't have to try again. You deactivate this option, if you switch from one version to another and the two versions are not compatible.
`url`	String	x	URL of the endpoint that this destination is based on
`endpoint`	Endpoint	x	ID of the endpoint that this destination is based on **Note** This is optional for consumed destinations.
`destinationTarget`	DestinationTarget	x	The destination target that is associated with the `AbstractDestination`
`credential`	AbstractCredential	-	(Optional) ID of the credential used for the destination
`additionalProperties`	String2StringMapType	-	(Optional) Extra configuration parameters

Subclasses

Exposed Destinations

ℹ Note

This subclass inherits from the `AbstractDestination` class.

ExposedDestination

Attribute Name	Type	Mandatory	Description
`targetId`	String	no	Stores target system ID of the registered service

Consumed Destinations

ℹ Note

This subclass inherits from the `AbstractDestination` class and has no additional fields. The "Endpoint" field is optional.

Configuring Consumed Destinations

Configure consumed destinations. Consumed destinations allow SAP Commerce to connect with external systems.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to System > API > Destinations > Consumed Destinations.
3. Click +.
4. Specify the following mandatory values:
   - ID – A unique identifier for the destination.
   - URL – The destination URL.
   - Active – Indicates if the destination is active or not.
   - Destination Target – The external system whose destination is consumed.
5. (Optional) Select Next, then specify the following additional values:
   - Endpoint – Reference to an endpoint object.
   - Credential – The destination credential type should match the destination type.
6. Click Save.

   In the list view, the destination is visible. If it is not, refresh your browser.
7. In the list view, select the destination.

Configuring Exposed Destinations

Configure exposed destinations. Exposed destinations allow external systems to connect to SAP Commerce.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to System > API > Destinations > Exposed Destinations.
3. Click +.
4. Specify the following mandatory values:
   - Destination Target – The external system whose destination is consumed.
   - Active – Indicates if the destination is active or not.

ℹ Note

You have to activate the destination before you can register it.

- Endpoint – Reference to an endpoint object.
- URL – The destination URL.
- ID – A unique identifier for the destination.

5. (Optional) Select **Next**, then specify the following optional values to configure an Integration API as an exposed destination:
   - Integration Object Configuration – The Integration Object to be exposed.
   - Credential – The destination credential should be the same credential type for the assigned Integration Object.
6. Click **Save**.

   In the list view, the destination is visible. If it is not, refresh your browser.
7. In the list view, select the destination.

Next Steps

Register your exposed destination and connect an external system to the destination.

Exposed Destination Registration

You can register exposed destinations to allow external systems to access your system. You can also deregister them.

Registering Exposed Destinations

You can register destinations to which the system sends SAP Commerce events and webservices.

Context

You want to register a destination in the Backoffice Administration Cockpit.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Navigate to **System** > **API** > **Destinations** > **Exposed Destinations** and, in the list view, select a destination that you want to register.
3. In the editor area, choose  (**Register Exposed Destination**).

   If the **Active** option is set to **False**, the button is inactive. To register the destination, change the option to **True**.
4. Confirm your decision.
5. After registration, refresh the editor area.

   The **TargetId** field for the destination populates with a unique ID.

Deregistering Exposed Destinations

You can deregister a destination that is already registered in a target system.

Context

You want to deregister a destination in Backoffice Administration Cockpit, for example, to disable or clean up older versions of an API.

ℹ Note

To delete destination, you first have to delete it.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Navigate to System > API > Destinations > Exposed Destinations and select the destination you want to deregister.
3. In the editor area, choose  (Unregister Exposed Destination).
4. Confirm your decision.

   A success message appears at the top of the page.
5. After deregistration, refresh the editor area.

   The TargetId field is empty.

Related Information

Destination Configuration

Pinging an Exposed Destination

You can verify if Omni Commerce Connect (OCC) or the Integration API associated with an Integration Object's exposed destination is running.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Navigate to System > API > Destinations > Exposed Destinations and, in the list view, select a destination.
3. In the editor area, choose  (Ping Destination).

> **Note**
> If neither OCC nor the Integration API is running, you might have to configure and assign the credentials for the exposed destination.

Standard Host Address for Inbound Connections

SAP provides a parameter that you can use to change the URL that you use for exposed destinations and their endpoints.

You configure a standard host address in the `local.properties` file by editing the parameter `ccv2.services.api.url.0`. For more information about configuring the standard host address, see Configuring the Standard Host Address for Exposed Destinations.

Credentials

This feature allows you to manage stored credentials that you assign to exposed or consumed destinations.

Use Case

An administrator wants to create, edit, or delete credentials.

When you assign a credential to an exposed destination, you define the authentication requirements for accessing the destination. For example, you can define the user name and password required for basic authentication.

When you assign a credential to a consumed destination, you enter the required authentication to access that destination. For example, you can enter the OAuth client ID and secret for accessing a specific API that SAP Commerce consumes.

Features

Credential Configuration

You can create the following types of credentials:

- Basic Credential
- Consumed Certificate Credential
- Consumed OAuth Credential
- Exposed OAuth Credential

Dependencies

After creating credentials, you assign them to destinations.

Credential Configuration

You can access credentials in Backoffice by navigating to System > API > Credentials.

In Backoffice Administration Cockpit, the Credentials node consists of the following credential types:

- Basic Credential
- Consumed Certificate Credential
- Consumed OAuth Credential
- Exposed OAuth Credential

ℹ Note

SAP recommends using consumed credentials for consumed destinations and exposed credentials for exposed destinations.

Classes

The following classes are available:

Abstract Credentials

AbstractCredential

Attribute Name	Type	Mandatory	Description
id	String	x	Unique ID of the `AbstractCredential`
`additionalProperties`	String2StringMapType	-	Optional, additional configuration parameters

Sub Classes

ℹ Note

The sub classes all inherit the `ID` attribute from `AbstractCredential`.

Basic Credentials

BasicCredential

Attribute Name	Type	Mandatory	Description
username	String	x	Username
password	String (encrypted)	x	Password

Consumed Certificate Credentials

ConsumedCertificateCredential

Attribute Name	Type	Mandatory	Description
privateKey	String (encrypted)	x	Certificate key
certificateData	String	x	Attribute to store the certificate content. It doesn't have to be encrypted.

Exposed OAuth Credentials

ExposedOAuthCredential

Attribute Name	Type	Mandatory	Description
oAuthClientDetails	OAuthClientDetails	x	Reference to an `OAuthClientDetails` object.
password	String (encrypted)	-	De-encryptable clone version of the `OAuthClientDetails.clientSecret`

Consumed OAuth Credentials

ConsumedOAuthCredential

Attribute Name	Type	Mandatory	Description
clientId	String	x	Client ID for a target system
clientSecret	String (encrypted)	-	Client secret for a target system
oAuthUrl	String	-	Authorization endpoint URL

Configuring Credentials

You create credentials that you assign to destinations.

Context

When you configure an exposed credential and assign it to an exposed destination, you define the credential that the external system requires to access your SAP Commerce system.

When you configure a consumed credential and assign it to a consumed destination, you define the credential that the SAP Commerce system requires to access an external system.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to **System** > **API** > **Credentials** and choose the type of credential you want to create.
3. Depending on the type of credential you want to create, proceed as follows:

Credential Type	Steps
Basic Credential **User name and password**	a. Choose +. b. Enter an ID for the credential. c. Enter the username and password required for access. d. Click Done.
Consumed Certificate Credential	**Note** This feature supports the SAP BTP Extensions Integration module, which uses this credential automatically. You do not need to create consumed certificate credentials.
Consumed OAuth Credential	a. Choose +. b. Enter an ID for the credential. c. Enter the OAuth client ID. d. Enter the OAuth URL. e. Enter the OAuth Secret. f. Click Done.
Exposed OAuth Credential **Reference OAuth clients**	a. Choose +. b. Select an OAuth client. **Note** You create OAuth clients in System > OAuth > OAuth Clients and reuse them here by referencing them. c. Enter an ID for the credential. d. Click Done.

Events

This feature allows you to define events and activate or deactivate event exporting.

Use Case

An administrator wants to define events that can be exported to other systems. They can use this system, for example, to extend the core functions of their SAP Commerce implementation by building custom logic that runs when specific events occur. For example, when one of their customers makes a payment or places an item in the shopping cart.

Features

Event Configuration

To define events, you create event configurations, which contain the details of the event. SAP delivers most events. However, you can create your own events as a Java class. You assign the java classes to event configurations in Backoffice Administration Cockpit.

Event Exporting

When events occur, they're exported to the target system. You can change the event exporting settings for a specific node or node cluster. You can also change the event exporting for individual events. Additionally, there is a dead letter queue for events that were not correctly exported, which provides an error log for debugging.

Delivered Events

SAP releases SAP Commerce with preconfigured events.

Dependencies

You have defined destination targets for your events.

Event Configuration

Before the events are exported to a target system, you have to register them in SAP Commerce. The details of the events are stored in the event configuration.

You can access the event configuration in Backoffice Administration Cockpit to create new and edit existing event configurations by navigating to System > API > Event Configuration.

After choosing the specific event configuration from the list view, the editor area appears. It consists of Event Configuration and Administration tabs, both with the attributes, which you can edit.

The event specification is sent to a target system only when Export Flag is set to True.

Classes

SAP Commerce events are simple Java classes without further metadata, which extend SAP Commerce `AbstractEvent`. To send the events to a target system, there must be a detailed event description stored in a special configuration. The description is also the base for generating the event specification, which is sent to a target system to register the exportable events.

The following types are available:

EventConfiguration Model

EventConfiguration

Attributes	Type	Mandatory / Optional	Comment
`eventClass`	String	x	SAP Commerce event class with full classpath. For example: `de.hybris.platform.commerceservices.event.Regis`
`version`	Integer	x	
`exportFlag`	Boolean	x	Defines if we want to send this event through the event channel - if to send it. • It turns on the event sending to a target system. • It enables the validation of event mapping in Backoffice Ad and you cannot import the impex with broken mapping if y
`priority`	Enum	x	
`exportName`	String	x	A code for a target system system to map the event data
`mappingType`	Enum	x	Can be `Generic` or `Bean`. It defines how the event mapping is do Converter bean.
`converterBean`	String	-	It must be defined if the `Bean` `mappingType` is set. Spring refere more complicated cases, when we should retrieve any additional d `de.hybris.platform.kymaintegrationservices.popu` **⚠ Caution** This is a sample implementation to show how a custom bean imp `SubmittingOrderEventPopulator` extends `AbstractEve PublishRequestData>` It can populate `publishRequestData`'s data field using any oth address info, but `AddressModel` is a list in `CustomerModel`, so

Attributes	Type	Mandatory / Optional	Comment
`eventPropertyConfigurations`	List<EventPropertyConfigutration>	-	The list of property configurations applied to `eventClass`.

Event Property Configurations

EventPropertyConfiguration

Attributes	Type	Mandatory	Comment
`propertyName`	String	x	Name/key of property, where to extract the value from original event with `propertyMaping path`. For example: `customerUid`
`propertyMapping`	String	x	Path of the event property, the value of which to extract and put under `propertyName` in result data. For example: `event.customer.uid`
`type`	String	x	Type of extracted value and property. You can choose from the following: • 'boolean' • 'integer' • 'number' • 'string'
`title`	String	x	A short name for a property. For example, **Base Store ID**
`required`	Boolean	x	It is a property required to be populated or not.
`description`	String	-	A description. For example: Unique identifier of the base store where the customer is registered.
`examples`	Map<String,String>	-	The examples of a property: • `code` - 000123 • `customerUid` - brian.customer@hybris.com • `baseSite` - electronics
`eventConfiguration`	EventConfiguration	-	The `EventConfiguration` to which the `EventPropertyConfiguration` belongs to.

Creating Event Configurations

Backoffice Administration Cockpit allows you to set up a new event configuration.

Prerequisites

You have created a Java class which contains the details of your event. For more information, see the `Event` class in Event Configuration.

Context

You want to create an event configuration in back office.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Go to System > API > Event Configuration.
3. Click +.
4. In the Event Class field, enter the full class path of your event Java class.
5. Fill in the other fields as required.
6. Add the event configuration properties.

   For more information, see Creating Event Property Configurations.
7. Choose Done.

Results

Your configuration is displayed on the list page. Select it to see the details.

Note

`EventMappingConstraint` checks the configuration before you save it. It tries to find the event class in a running system and the fields from `Mapping Configuration`.

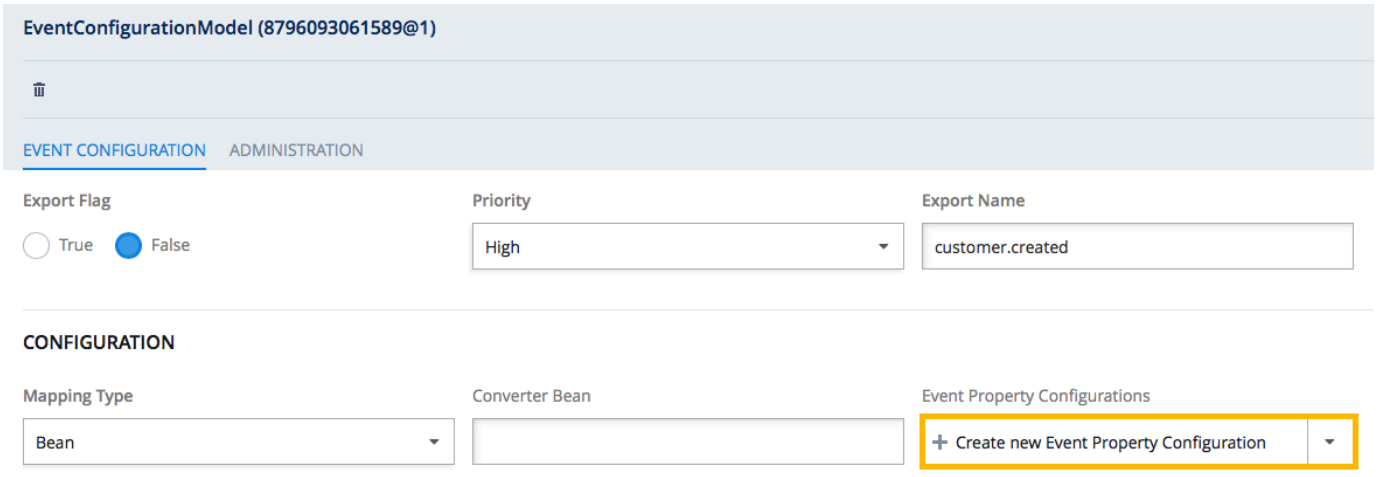
If you select the `Bean Mapping Type`, it looks up the spring bean based on the `Converter Bean` property and checks whether its interface is a converter. It does not check the source and target types of the `Converter Bean` property.

Creating Event Property Configurations

You can create a new event property configuration in Backoffice Administration Cockpit.

Procedure

1. Log in to Backoffice Administration Cockpit.
2. Navigate to System > API > Event Configuration.
3. Choose the event configuration from the list view to open an editor area.
4. In the Event Configuration editor area, under Event Property Configuration, choose Create new Event Property Configuration, to open the Create New Event Property Configuration wizard.



5. Fill in all required fields and choose Done.

Results

The newly created property is added to your event configuration.

Event Exporting

You can change the event exporting settings for a specific node or node cluster and see events that the system could not export.

Activating and Deactivating Event Exporting

In theBackoffice Administration Cockpit, you start and stop the system from exporting events.

Dead Letter Queue

The API Registry module provides a dead letter queue for exported events that cannot be delivered. You access the dead letter queue in Backoffice under System > API > Event Dead Letter Queue. In the dead letter queue, you see the details of all events that could not be exported, for example, the payload and the error that occurred.

Classes

The following classes are available:

Dead Letter Queue

Note

You cannot write to this class.

EventExportDeadletter

Name	Type	Description
`id`	`String`	The ID of the event
`eventType`	`String`	The type of the event
`timestamp`	`Date`	The time of the event
`destinationTarget`	`DestinationTarget`	The `destinationTarget` value of the `EventConfiguration`
`destinationChannel`	`DestinationChannel`	The `destinationChannel` value of the `EventConfiguration`
`payload`	`String`	The payload of the event export request. This is in the JSON format.
`error`	`String`	The payload of the event export response. This is in the JSON format.

Changing Event Exporting Settings

You can activate or deactivate event exporting for nodes or node clusters in the Backoffice Administration Cockpit.

Context

You change the settings in the **Event Configurations** list view. You can select one of the following settings:

Event Export Settings

Setting	Description

Setting	Description
Enable for Current Node	Allows the SAP Commerce node that you are currently logged into to export events
Disable for Current Node	Prevents the SAP Commerce node that you are currently logged into from exporting events
Enable for Whole Cluster	Allows the whole SAP Commerce node cluster that you are currently logged into to export events
Disable for Whole Cluster	Prevents the whole SAP Commerce node cluster that you are currently logged into from exporting events

Procedure

1. Open the Backoffice Administration Cockpit.
2. Choose System > API > Event Configurations.
3. Choose  (Event Exporting Settings).
4. In the Event Exporting screen, choose the setting that you would like to activate.

Delivered Events

You can find the events that are delivered with SAP Commerce in this table.

This table displays the content that is available in the following impex: `bin/modules/api-registry/apiregistryservices/resources/impex/essentialdata-event-configuration.impex`.

> **Note**
> To see additional information, click Show/hide columns.

Events Reference

Extension	Event Name	Export Class	Property M
commerceservices	RegisterEvent	customer.created	event.baseS event.custo event.custo
platformservices	SubmitOrderEvent	order.created	event.order event.order event.busin
commerceservices	ForgottenPwdEvent	password.forgotten	event.baseS event.custo event.custo
core	BeforeSessionCloseEvent	session.beforesessionclose	
core	AfterSessionCreationEvent	session.aftersessioncreation	
core	AfterSessionAttributeChangeEvent	session.aftersessionattributechange	event.attrib

Extension	Event Name	Export Class	Property M
core	AfterItemCreationEvent	item.creation	event.typeC
core	AfterItemRemovalEvent	item.removal	
core	AfterSessionUserChangeEvent	session.aftersessionuserchange	event.previ
core	AfterTenantRestartClusterAwareEvent	system.aftertenantrestartclusteraware	event.tenar
core	AfterTenantInitializationClusterAwareEvent	system.aftertenantinitializationclusteraware	event.tenar
processing	RepollEvent	system.repoll	event.nodel event.node(
addonsupport	AddonSampleDataImportedEvent	system.addonsampledataimported	
commerceservices	CoreDataImportedEvent	system.coredataimported	
acceleratorservices	UploadDataEvent	system.uploaddata	
backoffice	ProcessFinishedEvent	cronjob.processfinished	event.proce event.proce event.proce event.proce event.proce event.proce
backoffice	ProcessStartEvent	cronjob.processstart	event.b2BA event.b2BA event.proce event.proce event.proce event.proce
basecommerce	OrderProcessingEvent	order.processing	event.proce event.proce event.order
basecommerce	OrderFraudEvent	order.fraud	event.proce

Extension	Event Name	Export Class	Property M
			event.proce event.order
basecommerce	ConsignmentProcessingEvent	order.consignmentprocessing	event.proce event.proce event.proce event.proce
ticketsystem	SessionEvent	session.event	
sapcustomerb2b	B2BRegistrationEvent	b2bcustomer.registration	event.baseS event.custo event.custo
sapcustomerb2c	CustomerReplicationEvent	b2ccustomer.replication	event.custo
eventpublisher	DefaultSecDeleteAddressEvent	cecaddress.deleted	event.custo event.addre
eventpublisher	DefaultSecDeleteCustomerEvent	ceccustomer.deleted	event.custo
ruleengine	KieModuleSwappingEvent	ruleengine.kiemoduleswapping	event.deplo event.rulesI
ruleengine	RuleEngineInitializedEvent	ruleengine.initialized	event.deplo event.rulesI
ruleengine	RuleEngineModuleSwapCompletedEvent	ruleengine.moduleswapcompleted	event.rulesI event.previ event.rulesI event.failed event.failur
ruleengine	RuleUpdatedEvent	ruleengine.ruleupdated	event.ruleC
ruleengineservices	RulesCompilationInProgressQueryEvent	ruleengine.rulescompilationinprogressquery	event.modu
ruleengineservices	RulesCompilationInProgressResponseEvent	ruleengine.rulescompilationinprogressresponse	event.modu
saprevenuecloudcustomer	SapRevenueCloudCustomerUpdateEvent	revenuecloudcustomer.update	event.custo

Extension	Event Name	Export Class	Property M
sapcustomerb2c	SAPRFCDestinationJCoTraceEvent	rfcdestination.jcotrace	event.jcoTra event.sourc
sapcustomerb2c	SAPRFCDestinationPingEvent	rfcdestination.ping	event.rfcDe event.mess
sapcustomerb2c	JCoConnectionsSnapshotClusterEvent	jcoconnections.snapshot	event.snaps
sapcustomerb2c	JCoConnectionsSnapshotClusterResultEvent	jcoconnections.snapshotresult	event.snaps event.cluste event.snaps
datahubadapter	DatahubAdapterImportEvent	datahub.adapterimport	event.poolN event.statu
commerceservices	SampleDataImportedEvent	system.sampledataimported	
eventtrackingmodel	CartAbandonedEvent	cart.abandoned	event.cartIc event.userI event.baseS
eventtrackingmodel	ProceedToCheckoutEvent	cart.proceedtocheckout	event.cartIc event.userI event.baseS
eventtrackingmodel	SuccessfulCheckoutEvent	cart.successfulcheckout	event.cartIc event.userI event.baseS
eventtrackingmodel	AddToCartEvent	product.addtocart	event.cartIc event.produ event.userI event.baseS event.userI event.baseS
eventtrackingmodel	RemoveFromCartEvent	product.removefromcart	event.cartIc event.produ event.userI event.baseS

Extension	Event Name	Export Class	Property M
eventtrackingmodel	ProductDetailPageViewEvent	product.detailpageview	event.produ event.userI event.baseS
eventtrackingmodel	ProductMediaViewEvent	product.mediaview	event.produ event.userI event.baseS
eventtrackingmodel	ProductReviewsViewEvent	product.reviewsview	event.produ event.userI event.baseS
eventtrackingmodel	BannerClickEvent	tracking.bannerclick	event.banne event.userI event.baseS
eventtrackingmodel	CartViewEvent	tracking.cartview	event.userI event.baseS
eventtrackingmodel	CategoryPageViewEvent	tracking.categorypageview	event.userI event.categ event.baseS
eventtrackingmodel	CategoryBrowseEvent	tracking.categorybrowse	event.userI event.categ event.baseS
eventtrackingmodel	FindStoresNearMeEvent	tracking.findstoresnearme	event.userI event.baseS
eventtrackingmodel	ForgottenPasswordEvent	tracking.forgottenpassword	event.userI event.baseS
eventtrackingmodel	PageThroughSearchResultsEvent	tracking.pagethroughsearchresults	event.userI event.baseS
eventtrackingmodel	PageViewEvent	tracking.pageview	event.produ event.userI event.baseS event.userI event.categ event.baseS event.userI event.baseS
eventtrackingmodel	PaymentDetailsEnteredEvent	tracking.paymentdetailsentered	event.userI event.baseS

Extension	Event Name	Export Class	Property M
eventtrackingmodel	RefineSearchEvent	tracking.refinesearch	event.userI event.baseS
eventtrackingmodel	SearchEvent	tracking.search	event.userI event.baseS event.userI event.baseS
eventtrackingmodel	SearchNoResultsEvent	tracking.searchnoresults	event.userI event.baseS
eventtrackingmodel	ShipmentDetailsEnteredEvent	tracking.shipmentdetailsentered	event.userI event.baseS
eventtrackingmodel	StoreLocationEnteredEvent	tracking.storelocationentered	event.userI event.baseS
eventtrackingmodel	UnsuccessfulAddToCartEvent	tracking.unsuccessfuladdtocart	event.userI event.baseS
eventtrackingmodel	UnsuccessfulCheckoutEvent	tracking.unsuccessfulcheckout	event.userI event.baseS
eventtrackingmodel	UnsuccessfulLoginEvent	tracking.unsuccessfullogin	event.userI event.baseS
eventtrackingmodel	UnsuccessfulPaymentEvent	tracking.unsuccessfulpayment	event.userI event.baseS
eventtrackingmodel	UnsuccessfulRegistrationEvent	tracking.unsuccessfulregistration	event.userI event.baseS
sapc4ccustomerb2c	SapC4cCustomerUpdateEvent	c4ccustomer.update	event.custo
acceleratorservices	SendReadyForPickupMessageEvent	order.readyforpickup	event.proce event.proce event.proce
acceleratorservices	SendPickedUpMessageEvent	order.pickedup	event.proce event.proce event.proce event.proce
acceleratorservices	FraudErrorEvent	order.frauderror	event.proce event.proce event.order

Extension	Event Name	Export Class	Property M
acceleratorservices	ExportDataEvent	cronjob.dataexport	event.baseS event.code
acceleratorservices	SavedCartFileUploadEvent	cart.fileupload	event.fileMe event.baseS event.custo
acceleratorservices	PaymentFailedEvent	order.paymentfailed	event.proce event.proce event.order
acceleratorservices	SendNotPickedUpConsignmentCanceledMessageEvent	order.notpickedcancelconsignment	event.proce event.proce event.proce event.proce
acceleratorservices	AuthorizationFailedEvent	order.paymentauthfailed	event.proce event.proce event.order
acceleratorservices	SendOrderPartiallyRefundedMessageEvent	order.partiallyrefunded	event.proce event.proce event.proce modificatio event.proce -> orderCod
acceleratorservices	SendOrderPartiallyCanceledMessageEvent	order.partiallycanceled	event.proce event.proce event.proce modificatio event.proce -> orderCod
acceleratorservices	PickupConfirmationEvent	order.pickupconfirmation	event.proce event.proce event.proce event.proce
acceleratorservices	SendDeliveryMessageEvent	order.deliverysent	event.proce event.proce event.proce event.proce
acceleratorservices	OrderFraudCustomerNotificationEvent	order.fraudcustomernotified	event.proce event.proce event.order
acceleratorservices	OrderCompletedEvent	order.completed	event.proce

Extension	Event Name	Export Class	Property M
			event.proce event.order
b2bacceleratorservices	ReplenishmentOrderConfirmationEvent	b2border.replenishmentorderconfirmation	event.order event.order
b2bacceleratorservices	OrderPendingApprovalEvent	b2border.pendingapproval	event.proce event.proce
b2bacceleratorservices	OrderApprovalRejectionEvent	b2border.approvalrejected	event.proce event.proce
b2bacceleratorservices	ReplenishmentOrderPlacedEvent	b2border.replenishmentorderplaced	event.cartT event.baseS event.custo
b2bapprovalprocess	ApprovalProcessStartEvent	b2border.approvalprocessstarted	event.b2BA event.b2BA
b2bapprovalprocess	QuoteRejectedEvent	b2border.quoterejected	event.order event.order event.mana
b2bapprovalprocess	ApprovalProcessCompleteEvent	b2border.approvalprocesscomplete	event.b2BA event.b2BA
b2bapprovalprocess	OrderRejectedEvent	b2border.orderrejected	event.order event.order event.appro
b2bapprovalprocess	MerchantRejectedEvent	b2border.merchantrejected	event.order event.order event.mana
b2bapprovalprocess	OrderApprovedEvent	b2border.approved	event.order event.order event.appro
b2bapprovalprocess	MerchantApprovedEvent	b2border.merchantapproved	event.order event.order event.mana
b2bapprovalprocess	QuoteApprovedEvent	b2border.quoteapproved	event.order event.order event.mana

Extension	Event Name	Export Class	Property M
basecommerce	CancelFinishedEvent	order.cancellationfinished	event.cance event.cance baseSiteUi event.cance orderCode
basecommerce	OrderFraudEmployeeNotificationEvent	order.fraudemployeenotified	event.order event.order
basecommerce	CancelPendingEvent	order.cancellationpending	event.cance event.cance baseSiteUi event.cance orderCode
basecommerce	OrderPlacedEvent	order.placed	event.cartT event.baseS event.custo event.proce event.proce event.order event.quote event.order event.order event.order event.quote event.order event.order event.order event.order event.quote
c4cquote	C4CQuoteBuyerOrderPlacedEvent	c4cquote.orderplaced	event.quote event.order event.order event.order
c4cquote	C4CQuoteBuyerSubmitEvent	c4cquote.submitted	event.quote event.userM event.quote event.baseS event.custo
c4cquote	C4CQuoteCancelEvent	c4cquote.cancelled	event.quote event.userM event.quote

Extension	Event Name	Export Class	Property M
			event.baseS event.custo
commerceservices	QuoteBuyerOrderPlacedEvent	quote.orderplaced	event.quote event.order event.order event.order event.quote event.order event.order
commerceservices	QuoteExpiredEvent	quote.expired	event.quote event.baseS event.custo
commerceservices	QuoteToExpireSoonEvent	quote.expiresoonevent	event.quote event.baseS event.custo
commerceservices	OrderCancelledEvent	order.cancelled	event.proce event.proce event.order
commerceservices	ChangeUIDEvent	customer.changeuid	event.baseS event.custo
commerceservices	QuoteSalesRepSubmitEvent	quote.salesrepsubmit	event.quote event.userM event.quote event.baseS event.custo
commerceservices	QuoteBuyerSubmitEvent	quote.buyersubmit	event.quote event.userM event.quote event.baseS event.custo event.quote event.userM event.quote event.baseS event.custo event.quote event.userM
commerceservices	QuoteSellerApprovalSubmitEvent	quote.sellerapproved	event.quote

Extension	Event Name	Export Class	Property M
			event.userM event.quote event.baseS event.custo
commerceservices	QuoteCancelEvent	quote.cancelled	event.quote event.userM event.quote event.baseS event.custo event.quote event.userM event.quote event.baseS event.custo event.quote event.userM event.quote event.quote
commerceservices	OrderRefundEvent	order.refund	event.proce event.order
processing	BeforeCronJobStartEvent	cronjob.beforestart	event.cronJ event.job ->; event.jobTy event.cronJ
core	AfterInitializationEndEvent	system.initializationend	event.times
commerceservices	CreateReturnEvent	order.returncreated	event.returı event.returı event.returı
processing	AfterCronJobFinishedEvent	cronjob.finished	event.cronJ event.job ->; event.jobTy event.cronJ event.result event.statu
core	AfterInitializationStartEvent	system.initializationstart	event.times

Extension	Event Name	Export Class	Property M
core	AfterTenantRestartEvent	system.tenantrestart	event.times event.tenar
processing	AfterCronJobCrashAbortEvent	cronjob.aborted	event.cronJ event.job ->; event.jobTy event.cronJ
sapcustomerb2c	SAPRFCDestinationRemoveEvent	rfcdestination.removed	event.rfcDe
sapcustomerb2c	SAPRFCDestinationUpdateEvent	rfcdestination.updated	event.rfcDe
warehousing	SendReturnLabelEvent	order.sendreturnlabel	event.retur event.retur event.retur event.retur
sapquoteintegration	SapCpiQuoteBuyerSubmitEvent	cpiquote.submitted	event.quote event.userM
sapquoteintegration	SapCpiQuoteCancelEvent	cpiquote.cancelled	event.quote event.userM event.quote event.quote
sapquoteintegration	SapCpiQuoteOrderPlacedEvent	cpiquote.orderplaced	event.order event.order event.quote
eventpublisher	DefaultSecDeleteB2BUnitEvent	cecb2bunit.deleted	event.b2bU
eventpublisher	DefaultSecDeleteOrderEvent	cecorder.deleted	event.order event.custo
eventpublisher	DefaultSecValidateCustomerEvent	ceccustomer.validated	event.custo
apiregistryservices	DynamicProcessEvent	dynamicprocess.executed	event.busin event.busin

Extension	Event Name	Export Class	Property M
apiregistryservices	EventExportDisabledEvent	eventexport.disabled	event.targe
apiregistryservices	EventExportEnabledEvent	eventexport.enabled	event.targe
apiregistryservices	EventExportFailedEvent	eventexport.failed	event.mess event.retry
commerceservices	AnonymousConsentChangeEvent	anonymousconsent.changed	event.conse event.curre event.oldCc event.other
commerceservices	ClosedAccountEvent	account.closed	event.custo event.site.u
commerceservices	ConsentGivenEvent	consent.given	event.conse event.conse
commerceservices	ConsentWithdrawnEvent	consent.withdrawn	event.conse event.conse
commerceservices	DeletedAddressEvent	address.deleted	event.custo event.site.u
commerceservices	LoginSuccessEvent	login.succeeded	event.custo event.baseS event.site.u
commerceservices	SavedAddressEvent	address.saved	event.custo event.site.u
commerceservices	UpdatedProfileEvent	profile.updated	event.custo event.site.u
core	LegacyLoginFailureEvent	legacylogin.failed	event.uid ->
core	LegacyLoginSuccessfulEvent	legacylogin.succeeded	event.uid ->
core	InvalidateModelConverterRegistryEvent	modelconverterregistry.invalidated	event.refres event.comp
apiregistryservices	InvalidateCharonCacheEvent	charoncache.invalidate	event.cache
kymaintegrationservices	InvalidateCertificateCredentialsCacheEvent	credentialcache.invalidated	event.consu

Extension	Event Name	Export Class	Property M
personalizationyprofile	InvalidateConsumptionLayerUserSegmentsProviderCacheEvent	segmentprovidercache.invalidated	

Charon Client Integration

This feature allows you to integrate your application with the Charon client, which provides a framework for composing asynchronous HTTP transactions.

Use Case

A customer wants to implement the APIRegistry module in SAP Commerce and enable Charon client integration. For a sample implementation, see Charon Client Sample Implementation.

Features

Charon Client Integration

You can enable Charon clients in your own implementation.

Dependencies

You must activate and configure Charon clients. For more information, see Secure HTTP Transactions.

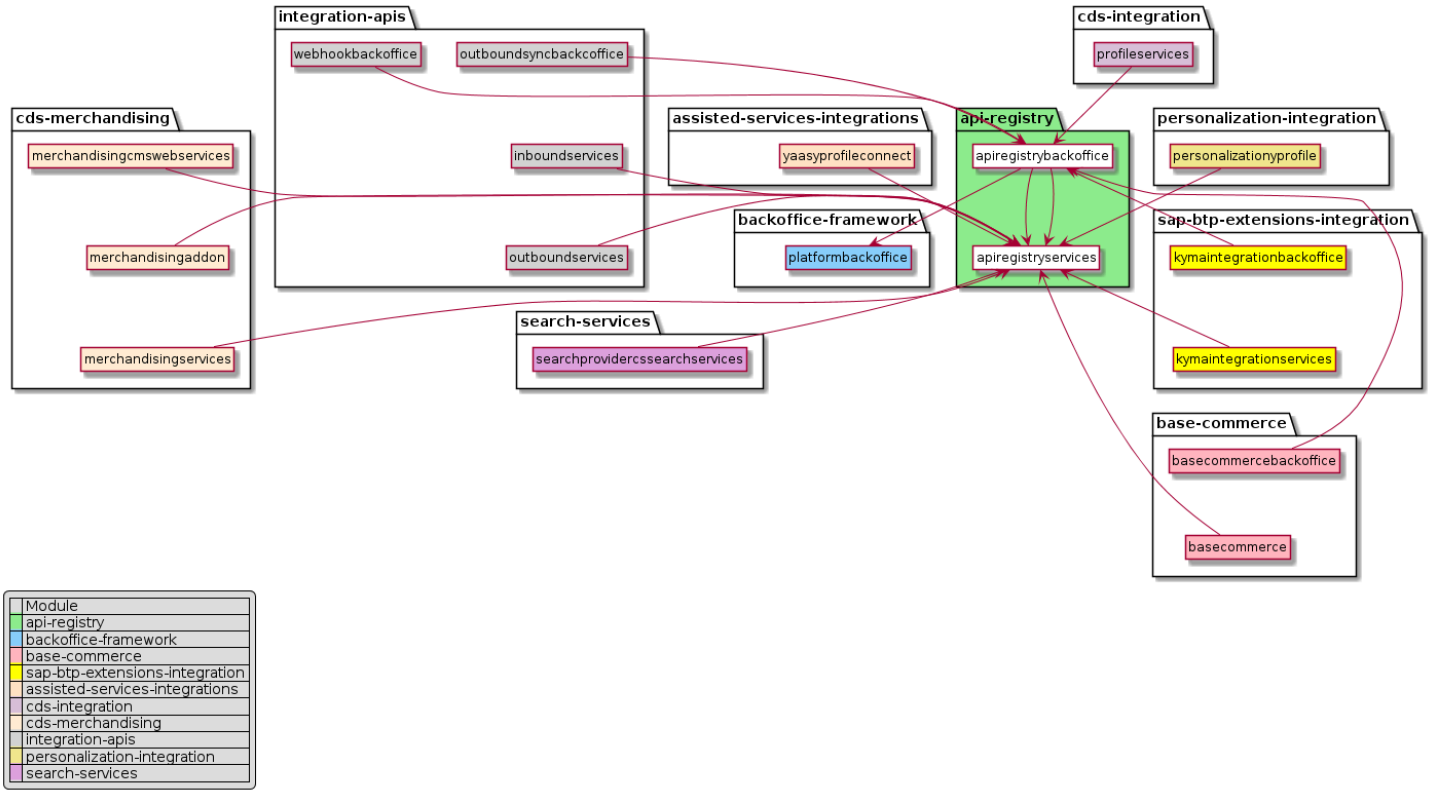
API Registry Module Architecture

The API Registry module is a set of extensions providing generic functionality for storing event configurations, endpoint configurations, and credentials.

Dependencies

→ Recommendation

For a better viewing experience, right-click the diagram and select one of the available browser options, such as **Open Image in New Tab**.



Recipes

For a complete list of SAP Commerce recipes that may include this module, see Installer Recipes.

For a complete list of the SAP Commerce Cloud, integration extension pack recipes that may include this module, see Installer Recipe Reference.

Extensions

The API Registry module consists of the following extensions:

apiregistryservices Extension
The `apiregistryservices` extension contains event configuration and endpoint configuration core functionality.
apiregistrybackoffice Extension
The `apiregistrybackoffice` extension is a Backoffice extension that contains widget settings for the new item types

apiregistryservices Extension

The `apiregistryservices` extension contains event configuration and endpoint configuration core functionality.

ℹ Note

An SAP Commerce extension may provide functionality that is licensed through different SAP Commerce modules. Make sure to limit your implementation to the features defined in your contract license. In case of doubt, please contact your sales representative.

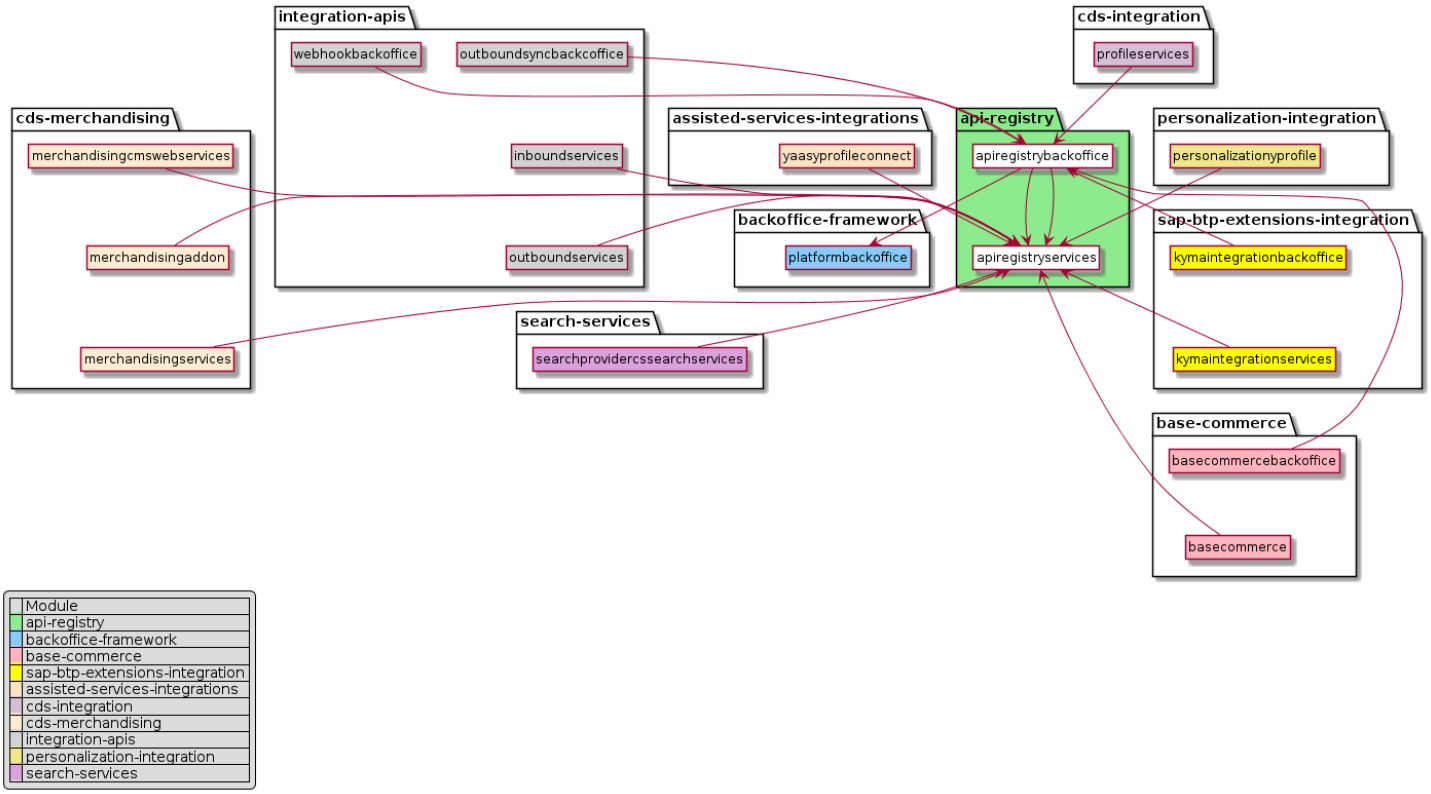
About the Extension

Name	Directory	Dependency	Related Module
`apiregistryservices` extension	`hybris/bin/api-registry`	`platform` extension	API Registry Module

Dependencies

→ Recommendation

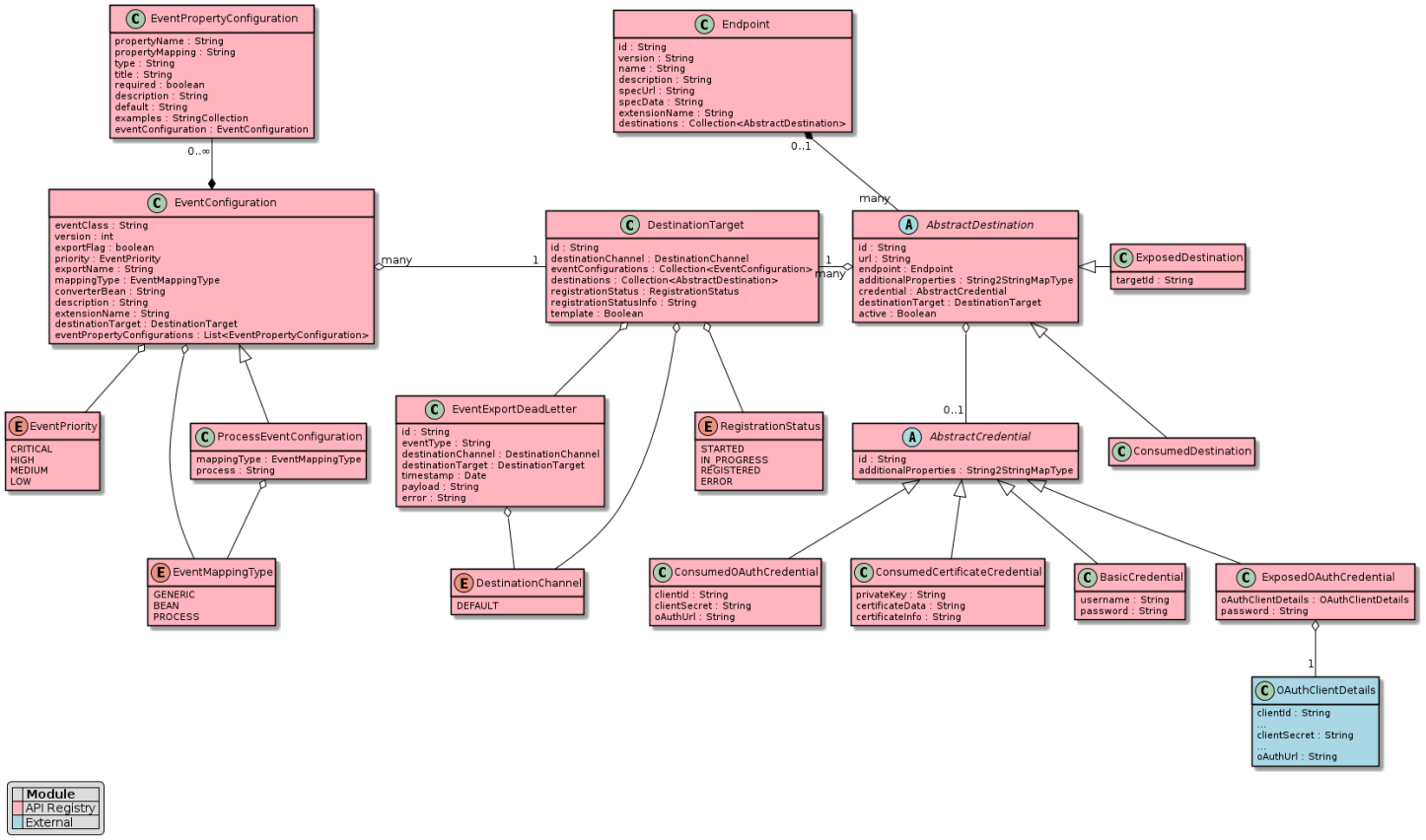
For a better viewing experience, right-click the diagram and select one of the available browser options, such as Open Image in New Tab.



Service Layer Items

→ Recommendation

For a better viewing experience, right-click the diagram and select one of the available browser options, such as Open Image in New Tab.



Data Access Objects

The `apiregistryservices` extension contains the following DAOs:

Name	Methods
`DestinationDao`	• `getDestinationsByChannel(DestinationChannel channel): List<AbstractDestinationModel>` • `getDestinationById(String id): AbstractDestinationModel`
`EventConfigurationDao`	• `findActiveEventConfigsByClass(String eventClass): List<EventConfigurationModel>` • `findActiveEventConfigsByChannel(DestinationChannel channel): List<EventConfigurationModel>`

Services and Strategies

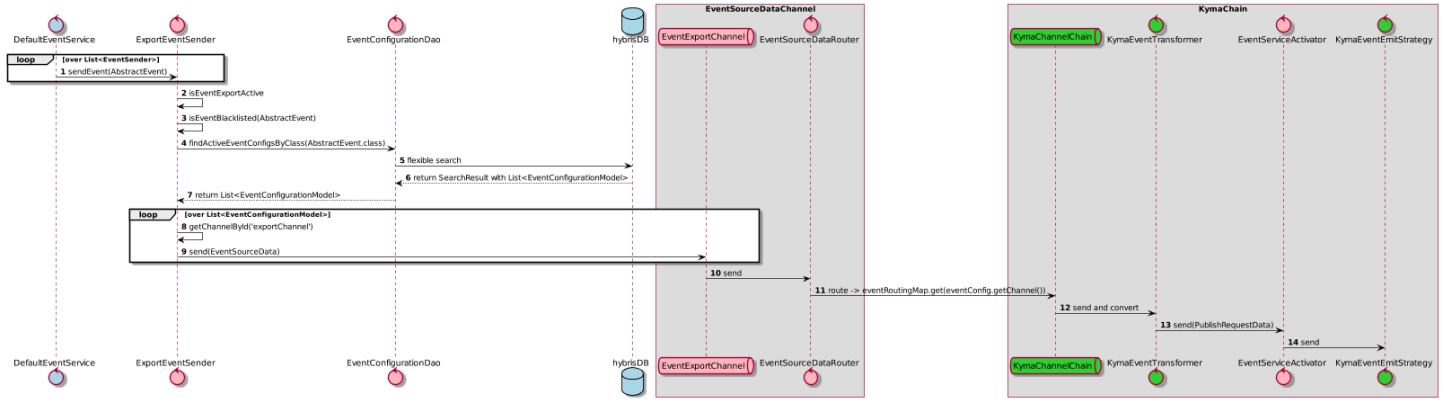
Name	Description	Methods
`ApiRegistrationService`	Service for unregistering and registering APIs.	• `registerExposedDestination(ExposedDestinationModel destination): void` • `unregisterExposedDestination(ExposedDestinationModel destination): void`
`DestinationService`	Service layer interface for `DestinationModels`.	• `getDestinationsByChannel(DestinationChannel channel): List<AbstractDestinationModel>` • `getDestinationById(String id): AbstractDestinationModel`
`EventEmitStrategy`	Strategy interface for sending event payload. `EmitStrategy` used as last point of integration chain in `EventServiceActivator`.	• `sendEvent(Object payload): void`

Name	Description	Methods
`ApiRegistrationStrategy`	Service for exporting and registration of webservices specifications. Strategy wired with specific channel by `apiRegistrationStrategyMap` in spring xml.	• `registerExposedDestination(ExposedDestinationModel destination): void` • `unregisterExposedDestination(ExposedDestinationModel destination): void`

Event Emitting

→ Recommendation

For a better viewing experience, right-click the diagram and select one of the available browser options, such as Open Image in New Tab.



The `ExportEventSender` is implemented as one of additional `EventSenders` to Platform. It is designed to prepare export configuration for any event fired in the system and propagating it to the specific event export service.

For every event that is exported there is an `EventConfigurationModel`. There can be more more than one `EventConfigurationModel` to one event. `ExportEventSender` simply moves all `EventConfigurationModel` to `eventSourceDataChannel`.

`EventSourceDataRouter` is responsible for mapping `EventConfigurationModel` to concrete spring integration channels. It compares `EventConfigurationModel.channel` with key from `eventRoutingMap`. The `eventRoutingMap` is an integration point where customer can add his own channel mappings. By default, the `EventConfigurationModel` is moved to `defaultChannel`, which ends in `MockEventEmitStrategy`, and logs `EventConfigurationModel`. `kymaintegrationservices` exension is a good example on how to add new router mapping and channel.

For more information, see Event Configuration.

API Registration

The `apiregistryservices` extension provides basic functionality for exposing and consuming Web APIs.

`ApiRegistrationService` is used for registering SAP Commerce Web APIs in an external systems. `ApiRegistrationStrategies` are channel-specific implementations, stored in `apiRegistrationStrategyMap`. It delegates the API registration to the channel strategy.

Module Properties

Property	Description
`apiregistryservices.eventPropertyConfiguration.delimiter`	Global delimiter for extracting the property value in `EventPropertyConfiguration.propertyMapping` field. For example: `event.order.code`
`apiregistryservices.events.blacklist`	Property which is used in `ExportEventSender` to filter events which should filter either by a class or a whole package.

Property	Description

	→ Recommendation
	Sending the following events cause errors during the initialization. It is recon in the blacklist:
	
	`de.hybris.platform.servicelayer.event.events.AfterInitial`
	`de.hybris.platform.servicelayer.event.events.AfterInitial`
	`de.hybris.platform.servicelayer.event.events.AfterSession`
	`de.hybris.platform.servicelayer.event.events.BeforeSessio`
	`de.hybris.platform.servicelayer.event.events.BeforeCronJo`
	
	You can edit the blacklist after initialization at your own risk.

`apiregistryservices.events.exporting`	Global cutout for event exporting. This property is set to false by default in `cx`
`apiregistryservices.allowedUrlProtocols`	Specifies the allowed protocol for API URLs. Multiple protocols can be split by The default value is https. For example, `apiregistryservices.allowedl`

Essential Data

The following data files are located under `apiregistryservices/resources/impex`:

- `essentialdata-api-configuration.impex`: It contains SAP Commerce API configurations.

  All of the webservices have a placeholder for their host address, `{ccv2.services.api.url.0}/rest/v2/api-docs`. Update this parameter before registration in an external system. For more information, see Configuring the Standard Host Address for Exposed Destinations. The impex also contains two `ConsumedDestinations` and one `ConsumedCertificateCredential` item for communicating with SAP BTP Extensions.

- `essentialdata-event-configuration.impex`: It contains data for `EventConfigurations` with `EventPropertyConfigurations` and configurations for more than 100 events. The `EventConfiguration` has extension name field, so the impex imports only `EventConfigurations` that exist in the classpath.

This data is the basis for the delivered destination target default template.

apiregistrybackoffice Extension

The `apiregistrybackoffice` extension is a Backoffice extension that contains widget settings for the new item types

ℹ Note

An SAP Commerce extension may provide functionality that is licensed through different SAP Commerce modules. Make sure to limit your implementation to the features defined in your contract license. In case of doubt, please contact your sales representative.

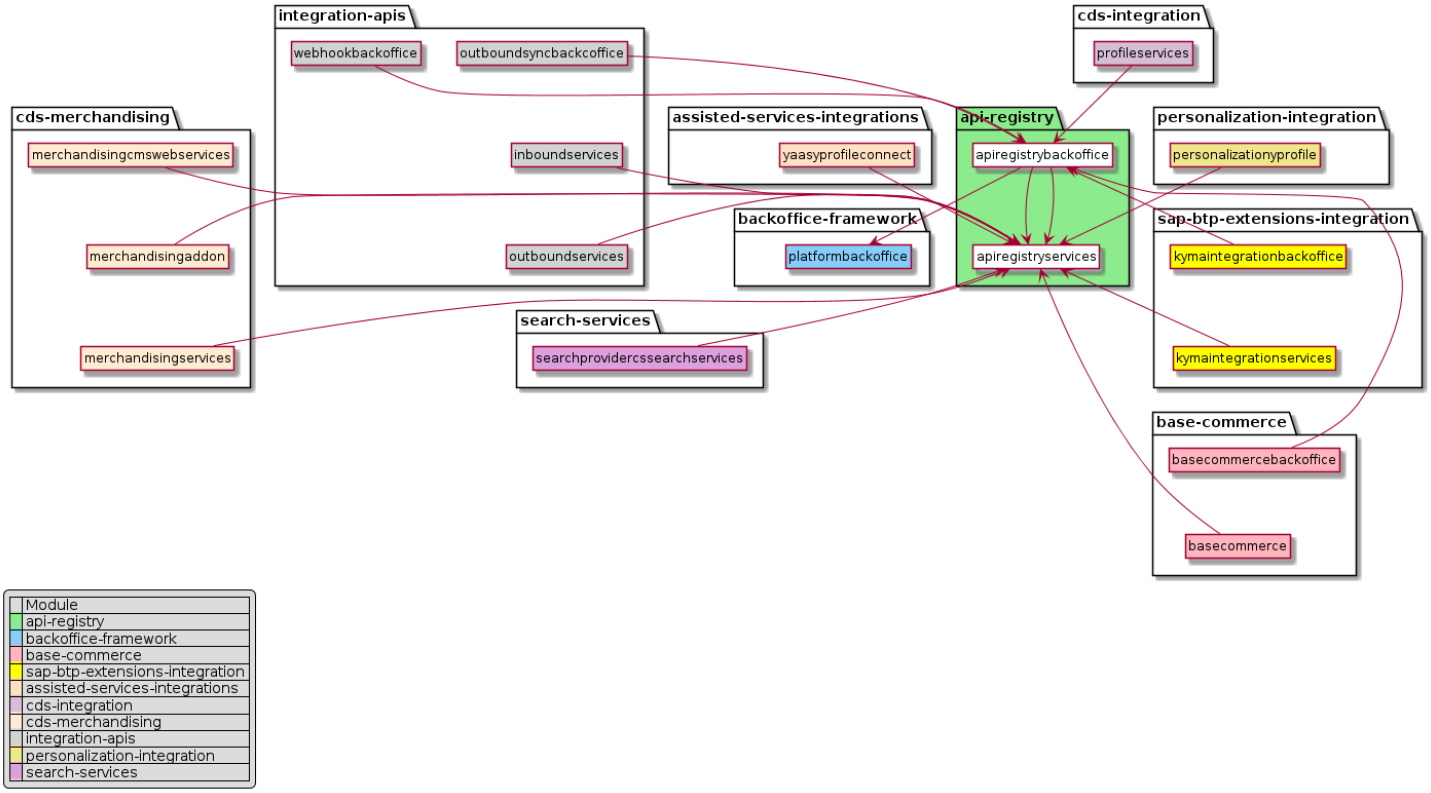
About the Extension

Name	Directory	Related Module
`apiregistrybackoffice` extension	`hybris/bin/api-registry`	API Registry Module

Dependencies

→ Recommendation

For a better viewing experience, right-click the diagram and select one of the available browser options, such as **Open Image in New Tab**.



API Registry Module Implementation

There are specific requirements for implementing the API Registry module.

[Migrating from Yaas Configuration to API Registry]

The `yaasconfiguration` extension is not available in SAP Commerce. You must migrate your data to the API registry module.

[Configuring the Standard Host Address for Exposed Destinations]

When implementing the API Registry module, you configure the standard host address for your exposed destinations.

[Configure Consumed Destination Connection Test]

In Backoffice Administration Cockpit, you can test the connection to consumed destinations.You configure this test by changing certain properties in the `local.properties` file.

[Charon Client Sample Implementation]

A sample integration including Charon client integration

Migrating from Yaas Configuration to API Registry

The `yaasconfiguration` extension is not available in SAP Commerce. You must migrate your data to the API registry module.

All functions available in the Yaas Configuration module are available in the API Registry module.

The following explains how the data in the Yaas Configuration module compares to the data in the API registry module. You can use this topic as the basis for your migration. For an overview of the API Registry data module, see the service layer items diagram in the `apiregistryservices` architecture reference.

Item Types Changes and Equivalents

Migrate your Yaas Configuration data to the API Registry item types.

Yaas Configuration	API Registry	Comments
• `YaasService`	• `Endpoint`	**Description**

Yaas Configuration	API Registry	Comments
• `BaseSiteServiceMapping`	• `ConsumedDestination`	The `Endpoint` class stores the general definition of a service, for example, the name, version, and API specification. `ConsumedDestination` holds the information about how the service is used by a specific project and channel. **Changes** `ConsumedDestination` provides all of the attributes that `YaasService` has, except for a `serviceScope` attribute. The `serviceScope` value should now be stored in `additionalProperties` with the `oauth.scope` key. The `YaasService.identifier` attribute value should now be stored in the `additionalProperties` with the `clientClassName` key. Unlike `BaseSiteYaasProjectRelation`, the relation between `ConsumedDestionation` and `ConsumedOAuthCredential` is stored in the `credential` attribute as a `ConsumedOAuthCredential` reference. The `BaseSiteServiceMapping.baseSite` value should be stored in `additionalProperties` with the `baseSite` key.
`YaasClientCredential`	`ConsumedOAuthCredential`	**Changes**`ConsumedOAuthCredential` has attributes for storing `clientId`, `clientSecret`, and `oAuthUrl`.
`YaasProject`	`DestinationTarget`	**Changes** `DestinationTarget` has attributes for storing `id`, `destinationChannel`, `destinations` (one-many relation), and `eventConfigurations` (one-many relation).
`YaasOrganisation`	`DestinationChannel`	**Changes** Unlike `YaasOrganisation`, `DestinationChannel` is an `enum` type. It's necessary to add an `enum` value for your project.

The following Yaas Configuration item types and relations don't have equivalents in API Registry module:

- `AbstractYaasServiceMapping`
- `BaseSiteYaasProjectRelation`

Service Changes and Equivalents

Service Lookup

Make the following code changes wherever they occur. The method parameters in both modules are the same. The new implementation throws `CredentialException` if there is no credential found for the given client type.

Deprecated (Yaas Configuration)

```
YaasServiceFactory.lookupService(final Class<T> serviceType)
```

New (API Registry)

```
ApiRegistryClientService.lookupClient(final Class<T> clientType) throws CredentialException
```

Building Configuration

Make the following code changes wherever they occur. The only difference is the credential attribute type, which is changed to `ConsumedOAuthCredentialModel`.

Deprecated (Yaas Configuration)

```
YaasConfigurationService.buildYaasConfig(final YaasClientCredentialModel clientCredential, final Class serviceTyp
```

New (API Registry)

```
ApiRegistryClientService.buildClientConfig(final ConsumedOAuthCredentialModel credential, final Class clientType,
```

Constants

The constants for the following, can be found in the service class `DefaultApiRegistryClientService` in the API Registry module. In the Yaas Configuration module, they are implemented in `YaasconfigurationConstants`.

```
YAAS_OAUTH_URL = "oauth.url";
YAAS_OAUTH_CLIENTID = "oauth.clientId";
YAAS_OAUTH_CLIENTSECRET = "oauth.clientSecret";
YAAS_CLIENT_URL = "url";
YAAS_CLIENT_SCOPE = "oauth.scope";
YAAS_TENANT = "tenant";
```

Example

The following example is from the `yaasyprofileconnect` extension:

```
public AsmProfileDataServiceClient getAdaptee()
{
    return yaasServiceFactory.lookupService(AsmProfileDataServiceClient.class);
    }
```

The following example is from the `apiregistryservices` extension:

```
public AsmProfileDataServiceClient getAdaptee()
{
  try
  {
    return getApiRegistryClientService().lookupClient(AsmProfileDataServiceClient.class);
  }
  catch (final CredentialException e)
  {
    throw new SystemException(e);
  }
}
```

Impex Changes

The following examples highlight the differences between impex files in the deprecated and new modules:

This is an example of a YaaS Configuration impex:

```
INSERT_UPDATE YaasOrganisation;identifier[unique=true];basePath[unique=true];
;ecpyaasmicrodemoorg;demobasepath;

INSERT_UPDATE YaasProject;identifier[unique=true];yaasOrganisation(identifier);baseSite(uid)
;devproject;ecpyaasmicrodemoorg;electronics

INSERT_UPDATE YaasClientCredential;identifier[unique=true];clientId;clientSecret;pubsubClient;yaasProject(identi
;basepath.devapplication;clientIdcredential;clientSecret;hybris.order;devproject;https://api.yaas.io/hybris/oauth
```

```
INSERT_UPDATE YaasService;identifier[unique=true];serviceURL;serviceScope
;ProductClient;https://api.yaas.io/hybris/product/v2;hybris.product_read_unpublished

INSERT_UPDATE BaseSite;uid[unique=true];yaasProjects(identifier)
;electronics;devproject

INSERT_UPDATE BaseSiteServiceMapping;baseSite[unique=true];yaasService(identifier)[unique=true];yaasClientCredent
;electronics;ProductClient;basepath.devapplication
```

This is an example of an API Registry module impex:

```
INSERT_UPDATE ConsumedOAuthCredential;id[unique=true];clientId;clientSecret;oAuthUrl
;basepath.devapplication;clientId;clientSecret;https://api.yaas.io/hybris/oauth2/v1;

INSERT_UPDATE Endpoint;id[unique=true];version[unique=true];specUrl;name
;ProductClient;v1;"empty";"Product Client v1"

INSERT_UPDATE ConsumedDestination;id[unique=true];url;additionalProperties(key,value)[map-delimiter=|];endpoint(i
;ProductClient;https://api.yaas.io/hybris/product/v2;oauth.scope->hybris.product_read_unpublished|baseSite->elect

INSERT_UPDATE DestinationTarget;id[unique=true];destinationChannel(code)
;devproject;ASM
```

Configuring the Standard Host Address for Exposed Destinations

When implementing the API Registry module, you configure the standard host address for your exposed destinations.

Context

SAP provides a project property that you can use to change the host address that you use for exposed destinations. This allows you to change multiple exposed destination URLs without having to change each host address separately. Reregister the URLs for the changes to take effect.

Procedure

1. Go to the `hybris/config` directory.
2. Open the `local.properties` file.
3. Set the `ccv2.services.api.url.0` property to the host address of your SAP Commerce system.

   For example:

   ```
   ccv2.services.api.url.0=https://localhost:9002
   ```

4. Go to the `hybris/bin/api-registry/apiregistryservices` directory.
5. For each of the exposed destinations that you want to use this host address, enter the parameter in the **URL** field in curly braces as follows: `{ccv2.services.api.url.0}`.

   SAP recommends performing this step for any custom exposed destinations you later create aside from the default ones.

   You can add any specific elements of the URL as usual.

   Example

   `{ccv2.services.api.url.0}/rest/v2`.

6. For each of the exposed destinations respective endpoints, enter the parameter in the **SpecURL** field in curly braces as follows: `{ccv2.services.api.url.0}`.

   You can add any specific elements of the URL as usual.

   Example

   `{ccv2.services.api.url.0}/rest/v2/api-docs`.

Configure Consumed Destination Connection Test

In Backoffice Administration Cockpit, you can test the connection to consumed destinations.You configure this test by changing certain properties in the `local.properties` file.

Context

You do not have to configure these properties as they are delivered with default values, but if you want to customize this process, you can.

For the test, you can configure how long the test takes to time out if there is no response from the destination. You can also define which status codes cause the test to return an error.

Procedure

1. Go to the `hybris/bin/api-registry/apiregistryservices` directory.
2. Open the `local.properties` file.
3. Edit the following properties if required:

Property	Description
`apiregistry.testConsumedDestination.url.timeout`	This specifies the timeout value for the test. After the period of time defined in this property, the test returns an error, informing the user that the test has timed out. Enter the value in seconds. By default, the property is set to 5 seconds.
`apiregistryservices.testConsumedDestinationUrl.httpstatus.error.codes`	This specifies the status codes that pings returns an error for, this might be useful if you want to include, for example, server errors in the list of status codes that count as errors. By default the peoperty contains the values `400,404,403,407,401`. You can add more code by adding them to the list separated by commas. All other status codes return a success message.

Charon Client Sample Implementation

A sample integration including Charon client integration

Context

ℹ Note

This is a sample implementation and may not entirely suit your needs. You can use this implementation as an example of how to implement Charon clients in your solution.

Procedure

1. Provide the configuration using ImpEx:

```
INSERT_UPDATE ConsumedOAuthCredential;id[unique=true];clientId;clientSecret;oAuthUrl
;basepath.devapplication;clientId;clientSecret;https://api.yaas.io/hybris/oauth2/v1;

INSERT_UPDATE Endpoint;id[unique=true];version[unique=true];specUrl;name
;ProductClient;v1;"empty";"Product Client v1"

INSERT_UPDATE DestinationTarget;id[unique=true];destinationChannel(code)
;devproject;ASM
```

```
INSERT_UPDATE ConsumedDestination;id[unique=true];url;additionalProperties(key,value)[map-delimiter=|];endp
;ProductClient;https://api.yaas.io/hybris/product/v2;oauth.scope->hybris.product_read_unpublished|baseSite-
```

2. Now, assume that you have a simple client that retrieves a product listing:

```
import java.util.List;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;

import com.hybris.charon.annotations.OAuth;


/**
 * The class of ProductClient.
 */
@OAuth
//retries,retriesInterval,timeout are configured as additionalConfigurations.
@Control(retries = "${retries}", retriesInterval = "${retriesInterval}", timeout = "${timeout}")
public interface ProductClient
{
    @GET
    @Path("/${tenant}/products")
    List<Object> getProducts();

}
```

3. Create a simple test class to create the service proxy for communication:

```
@Test
public void testProduct() throws Exception
{
   //ConsumedDestination should be configured so that the baseSite refers the currentBaseSite

    //ApiRegistryClientService instance refers to DefaultApiRegistryClientService
    final ProductClient productClient = apiRegistryClientService.lookupClient(ProductClient.class);
    System.out.println("Product :" + productClient.getProducts().get(0));

}
```

Results

In the log output from this test class, you see something like the following:

```
INFO  [main] (junit) [CharonHandler] baseUrl:https://api.yaas.io/hybris/oauth2/v1, method:POST, consumes:Optional
INFO  [nioEventLoopGroup-2-1] [CharonHandler] baseUrl:https://api.yaas.io/hybris/product/v2, method:GET, consumes
Product from the Yaas :{id=55f6a0606b8ef026f803eb9f, yrn=urn:yaas:hybris:product:product:devproject;55f6a0606b8ef
```

Search and Navigation Module

The Search and Navigation module functionality helps your customers to browse through the pages of your web stores and view the search results based on the facet settings. This can contribute to higher conversions, larger orders and more page views from people who use the search and navigation features.

Features	Architecture	Implementation

Search Query Templates

Manage Search Configurations

solrfacetsearch Extension

solrserver Extension

Solr Server Installation

Solr Facet Search Configuration

Solr Security

Multiple Solr Version Support

Search and Navigation Module Features

The Search and Navigation module provides a range of features related to creating and managing search configurations. For example, you can create search query templates for your search configurations or manage

Search Query Templates

The search query templates introduce numerous customization options for getting improved search results for the customers. When creating the templates you can use various attributes, such as page size, grouping or sorting with respect to various channels or even storefront parts.

Manage Search Configurations

Create and manage your search, indexing, and server configurations using the Backoffice Administration Cockpit.

Search Query Templates

The search query templates introduce numerous customization options for getting improved search results for the customers. When creating the templates you can use various attributes, such as page size, grouping or sorting with respect to various channels or even storefront parts.

Use Case

So far the user has received the same results while searching by keyword or using suggestions, because the results were governed by the storefront settings. The user wants to easily adjust the query depending on what they want to achieve. For example, remove fields like images from the suggestions template to improve the performance or change the sorting to draw the attention to certain products.

Features

Creating Templates

Create customized templates to accommodate your search configurations and improve performance.

Variants Grouping

Use the template settings to group the variants for a specific item, to save your customers' time.

Keyword Highlighting

Configure keyword highlighting for terms, to make them more visible on the storefront.

Configuring Suggested Terms

Create a template for suggested terms and customize it by adding sorting and restricting fields.

Dependencies

There are no specific dependencies for using search query templates.

Search Query Templates in Backoffice Administration Cockpit

Get to know how to manage the search query templates in Backoffice Administration Cockpit.

The Backoffice Administration Cockpit helps you to create the templates and define their settings so that they suit your needs. Follow the links below to get to know the functionality in detail.

Search Query Template Types

The following two types are supported out of the box: DEFAULT and SUGGESTIONS.

DEFAULT

The DEFAULT template is available on the template list right from the start for you to work with. In further sections you find information about what it holds and how it works. For details, see Search Query Template Features.

SUGGESTIONS

The SUGGESTIONS template is provided with the solution, however not available on the list right from the start, however you can add it at any time. The SUGGESTIONS query template allows you to define and configure the fields to be included in the results in the suggestions list. For details, see SUGGESTIONS Template.

Storefront

Whenever a storefront is mentioned it means the following: `https://apparel-uk.local:9002/yacceleratorstorefront/en`.

Access Search Query Templates

Location of the Search Query Templates.

DEFAULT template

The DEFAULT template is the default template that comes together with search configuration. It is predefined to include certain query properties and settings, but can be easily customized.

SUGGESTIONS Template

The SUGGESTIONS is supported out of the box and allows you to configure the settings for the suggested items that appear when you type in the term in the search box.

Access Search Query Templates

Location of the Search Query Templates.

Context

The steps below show you how to navigate to the Search Query Templates in the Backoffice Administration Cockpit.

Procedure

1. In the Backoffice Administration Cockpit navigate to **System** > **Search and Navigation** > **Solr Facet Search Configuration** > **Indexed Types**. You can also use the search field to filter the tree entires.

   A list of available (if any) search configurations appears. Select the one you have created the search query template for and click it to open an editor.

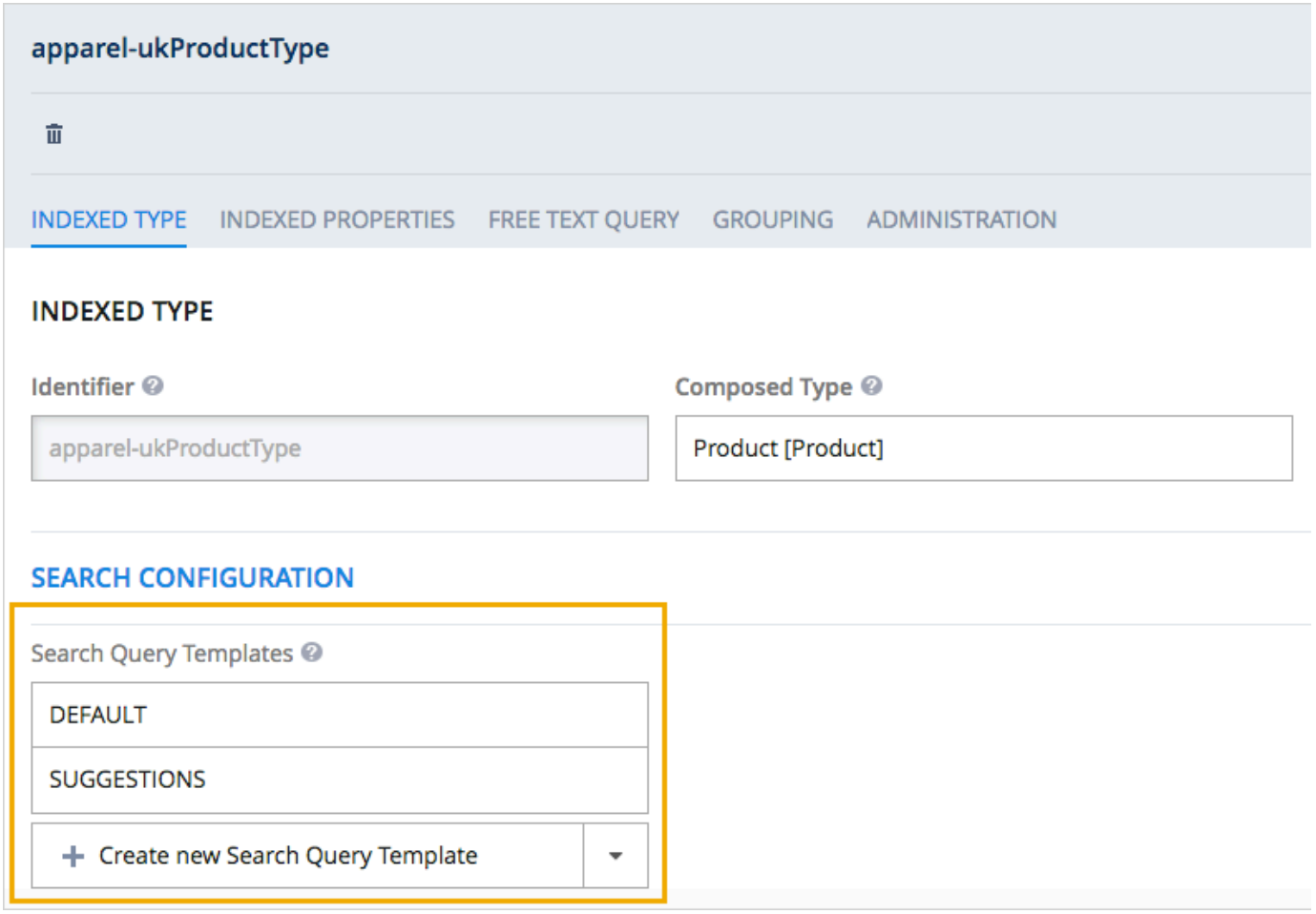
2. Click the **Indexed Type** tab and navigate to **Search Configuration** section.

   You can see the templates.

   Note

   The DEFAULT template is always present.

   Double-click the template if you want to edit it.

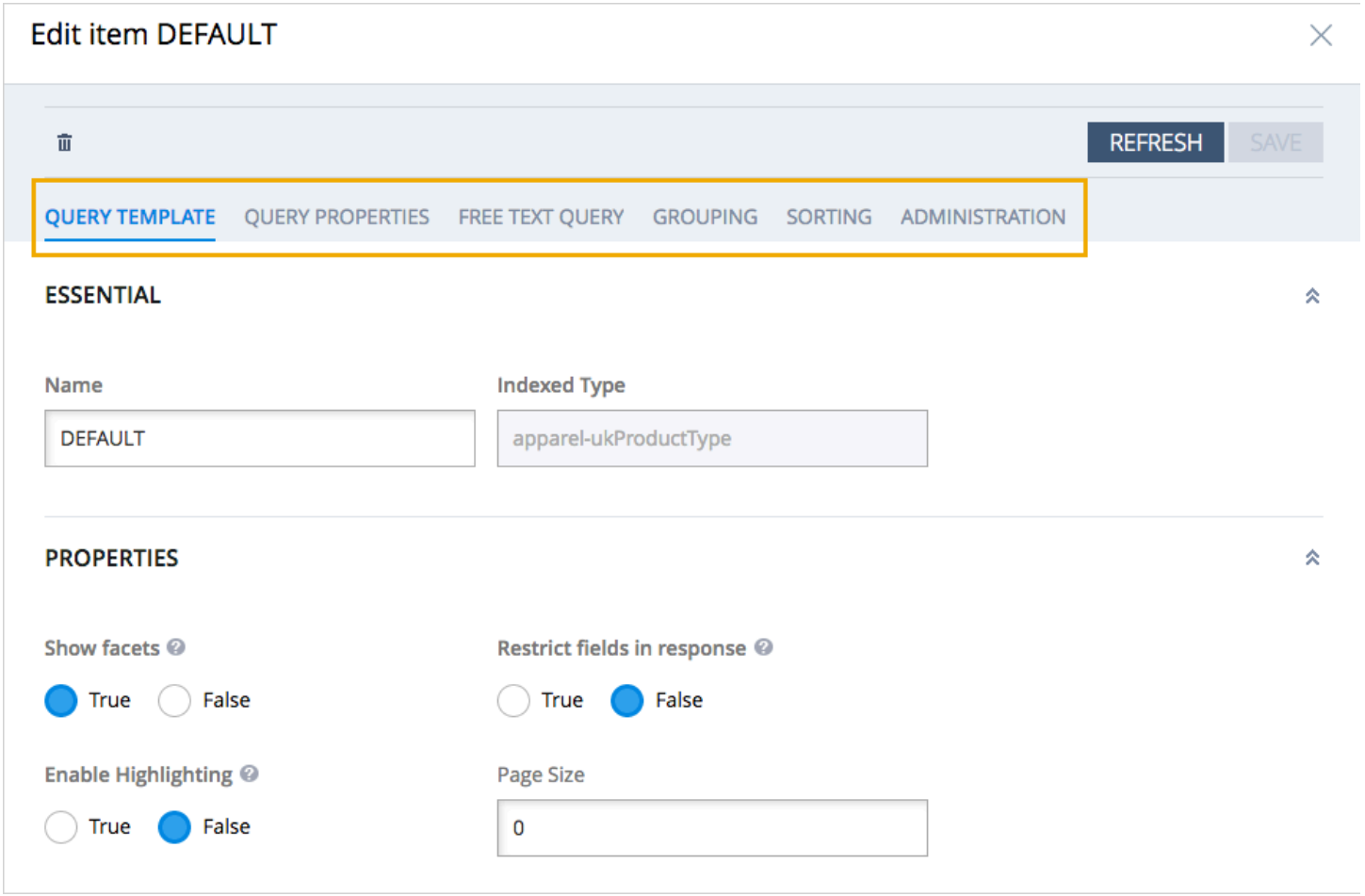


Search Query Template Features

Discover the functionality of search query templates using the example of the DEFAULT template.

The DEFAULT template is available for you right from the start. It provides the following functionality:

- **Query Template Essential Information**: Basic information about the template and main properties that apply to the entire template.
- **Query Properties**: A list of query properties set for a given index type. For each query property, you can additionally define the free text query and facet settings.
- **Free Text Query:**: An advanced free text query builder.
- **Grouping**: Settings that allow grouping of different variants of the same item.
- **Sorting**: Settings that allow sorting of the items.



Next Steps

If you want to learn more about the configuration of the DEFAULT template, see DEFAULT template.

Follow the instructions in SUGGESTIONS Template to learn more about the features while creating and configuring your own template.

DEFAULT template

The DEFAULT template is the default template that comes together with search configuration. It is predefined to include certain query properties and settings, but can be easily customized.

Configuration Scenario

- On the example of the DEFAULT template, get to know more about the basic settings: Basic Settings
- Define the free text query for Query Properties to improve your search experience: Query Properties
- Configure the grouping settings to group the variants of one item together and make browsing through the results faster and more efficient: Configuring Variant Grouping.
- Learn how to highlight the keywords win your search to make them easy to notice: Keyword Highlighting

Basic Settings

The essential settings of the search query template provide you with the following information:

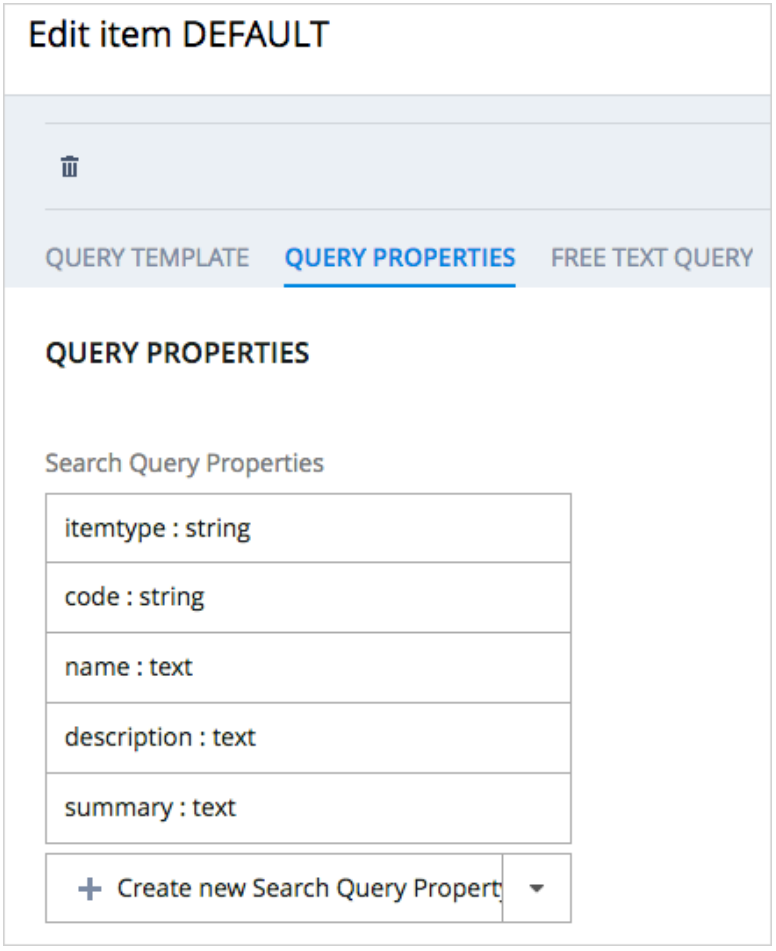
- Name: the name of your template. The name needs to be capitalized, because otherwise it won't match the naming convention and the template won't work.
- Indexed Type: the indexed type the template is created for.

The **Properties** section includes the global settings for the template.

- **Show facets**: If set to **true**, the query properties selected as facets will be visible.
- **Restrict fields in response**: If set to **true**, only the query properties set to be included in response will be displayed. For details, see Restrict Fields in Response.
- **Enable Highlightling**: If set to true, the query properties with highlighting option selected will be highlighted in the search result. For details, see Keyword Highlighting.

Query Properties

The query properties are the properties of an indexed type. They contain their own search configurations (like free text query settings or facet settings) and are also indexed in the process.



Once you double-click a property, you will come across the following set of tabs:

- **Essential Settings**: allow you to decide if this property is included in the response (while limiting the nuber of fields) or used for highlighting.
- **Free Text Query**: allows for defining settings that make the search more efficient and precise:
  - **Free Text Query**: facilitates finding particular terms,
  - **Free Text Fuzzy Query**: facilitates finding a term even if it is misspelled,
  - **Free Text Wildcard Query**: facilitates finding a term even if only the first, last or middle part of it is known to you,
  - **Free Text Phrase Query**: facilitates finding the entire phrase.

  For details on defining the free text query, see Define the Query Property Details.
- **Facet Settings**: Here you can decide whether the property should be displayed as a facet (if possible), what type of facet it should be, and which providers should apply to it. For details, see Facet Settings.

Define the Query Property Details

If you want to make your search effective and to the point, and at the same time eliminate possible problems stemming from misspelled and incomplete terms, you should properly configure your free text query settings

A wizard helps you to define the phrase query, enable wildcards, and configure fuzziness settings.

Add Free Text Query and Phrase Query

Configure your free text

Prerequisites

For the purpose of this example there are no settings defined for any of the properties. As a result, if you go to the storefront and search for a particular term (such as **belt**, the system shows you all available results, without any limiting to what you actually searched for.

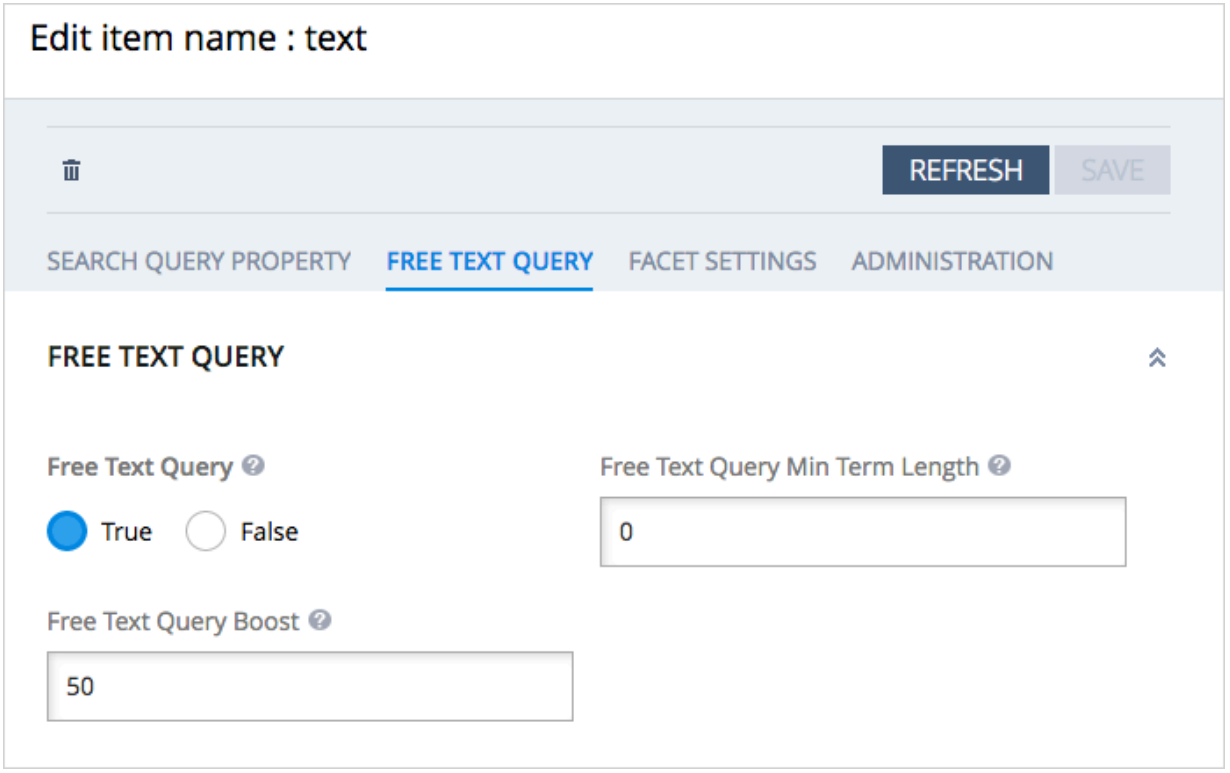
Context

To show you how the free text query and the phrase query work, you will define the settings for the **name** property.

Procedure

1. Navigate to the DEFAULT template. If in doubt, follow the steps from Access Search Query Templates.
2. Click the DEFAULT template and navigate to **Query Properites**.
3. Select the **name** property and double click it.
4. In the new window, navigate to **Free Text Query**.

   This is the base setting for each property. If you don't enable the **Free Text Query** for a property, it won't be included in the search.
5. Enable the **Free Text Query** by setting it to **True**. Remember to save your changes.

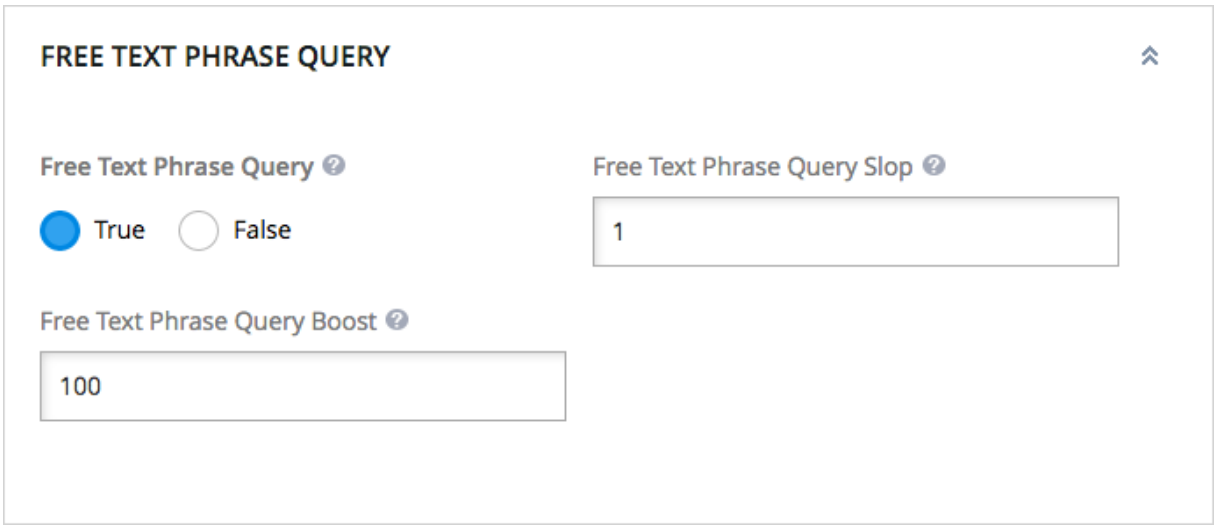


6. If you want the search term to be of a particular length, you can set it in **Free Text Query Min Term Length**. It means that if you enter 4, the term such as cap will be ignored since it has only three characters. For the purpose of this example, let's leave it at 0.
7. You can also boost the settings for the free text query, so that the score is boosted by a specified value. In our example it is 50.

   If you return to the storefront and search for **belt**, the results are limited to items with the word **belt** in their name.

Enabling the free text query leads to the limited results, but most surely you want them limited even more. Try looking for **belt green**. If you try this on the storefront, the results don't change. To be able to search for phrases you need to enable the free text phrase query first.

8. Go to the Free Text Phrase Query section and set it to **True**.



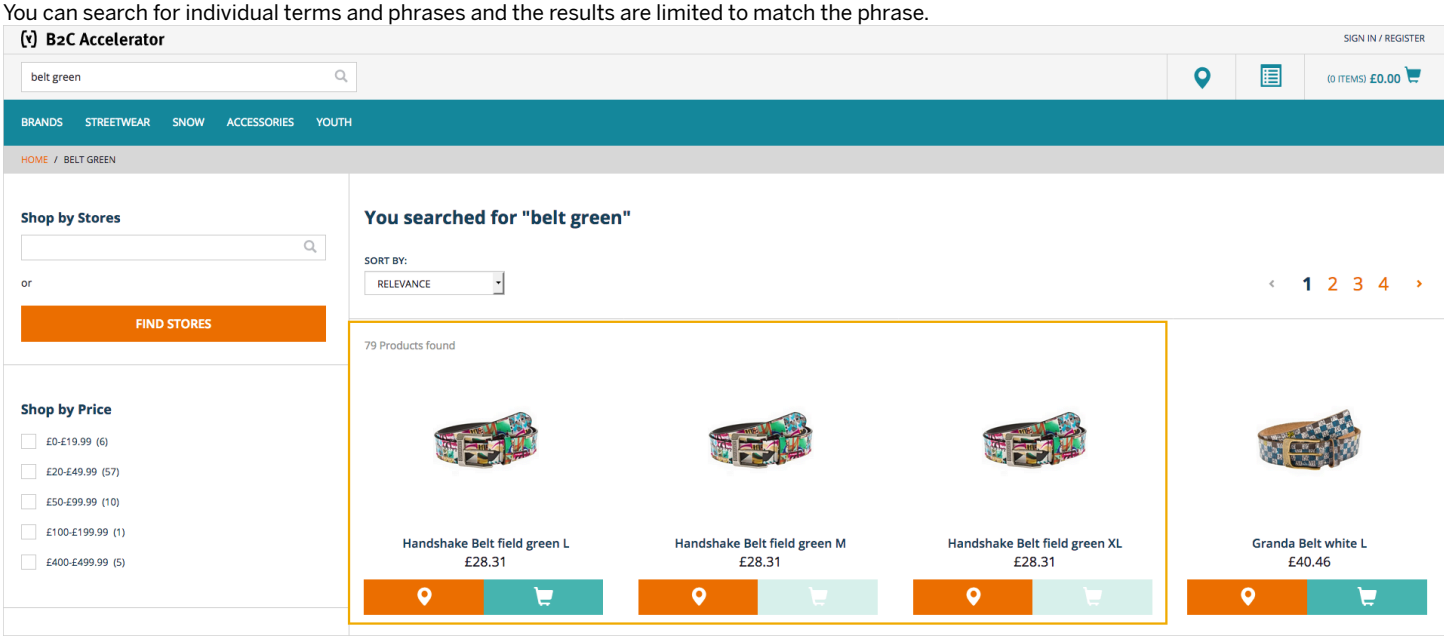
9. Define Query Slop, which is the number of words allowed between the terms in the query phrase. 0 means that the system will search for an exact phrase. Set the value for 1. Remember to **save** the changes.

   Come back to the storefront and search for **belt green**. The term you are looking for should come first. You can notice one word in the middle of the phrase - that's what the query slop was needed for.

10. Boost the phrase query match over the free text query match to make the system look for the phrases first.

Results

You can search for individual terms and phrases and the results are limited to match the phrase.



Add Wildcard Text Query and Fuzzy Query

Phrase query helps you to search for the entire phrases, but it may happen that the entry is mispelled or incomplete. To explain how you can deal with possible problems, you will define the settings for the **code** query property.

Procedure

1. Double-click the DEFAULT template and navigate to Query Properties.
2. Select **code** query property.
3. Enable the Free Text Query. Remember to save your changes.

   Try searching for a specific code, such as `300604640`. The item is found.

Activating the Free Text Query results in the item being found, but you need to know the exact code. Imagine the situation when you only remember a part of the code. Try typing just a part of your code. Nothing is found, or the system returns random results. To avoid this, you need to enable the support for wildcards.

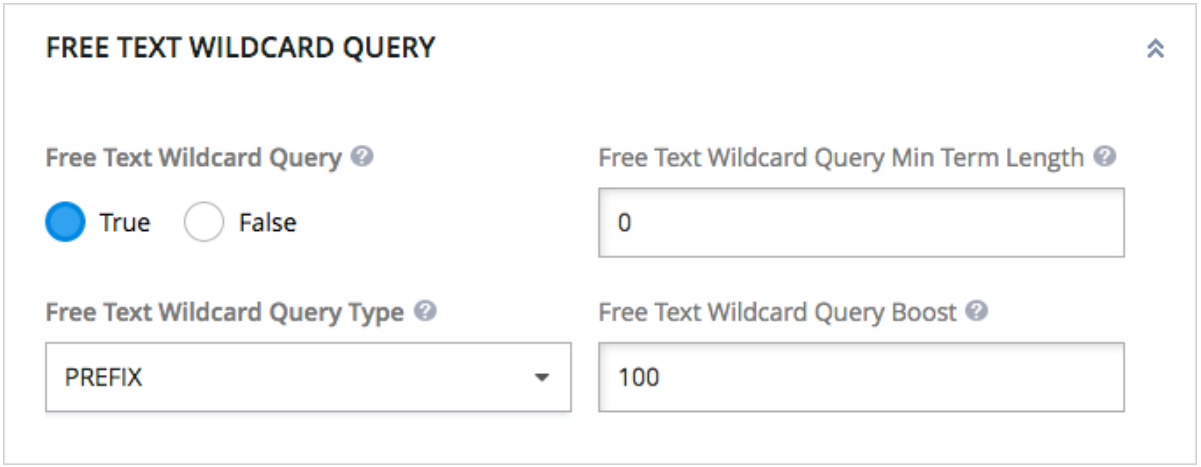
4. Go to Free Text Wildcard Query section and set it to **True**.
5. Select the Wildcard Query Type.

   You have the following options to choose from:

   - Prefix: the multiple character wildcard will be applied at the beginning.
   - Postfix: the multiple character wildcard will be applied at the end.
   - Prefix and Postfix: the multiple character wildcard will be applied both at the beginning and at the end.

   Select Prefix. Save your changes.
6. You can also set the minimum wildcard term length, to ignore the wildcards with character number lower than provided. Leave the value at 0.

   The result should be as follows:



7. Go to the storefront and type `604640` .

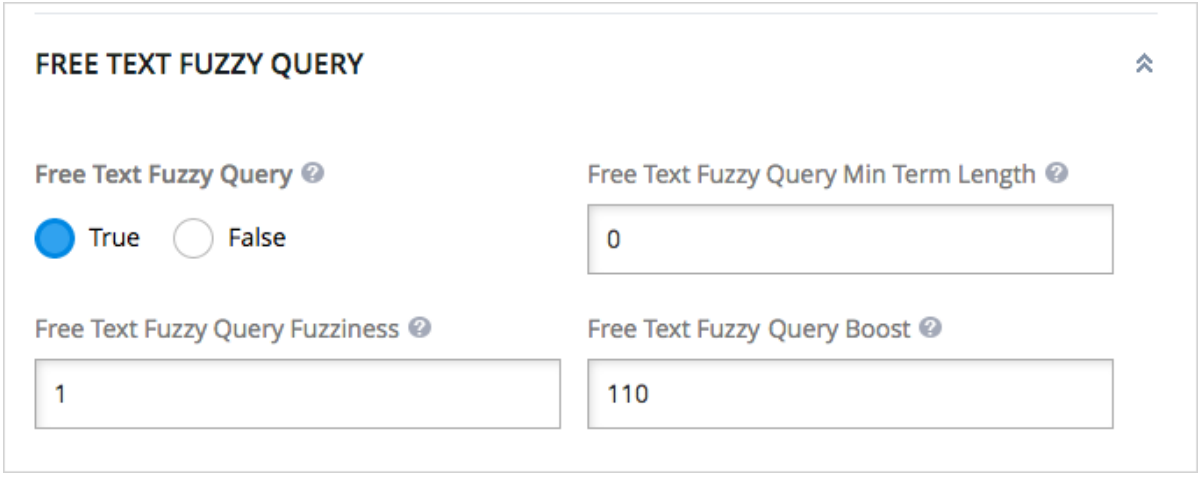
   The item is found. Try using different options to check if the search works properly.

Now you can search for the items even if you enter just a part of the term. Let's look at the case when you misspell an entry, entering 300603640 instead of 300604640. Again, nothing is found. That's why you need the fuzziness support, so the system can suggest you the correct item.

8. Go to Free Text Fuzzy Query and enable it.
9. Set the level of fuzziness to 1.

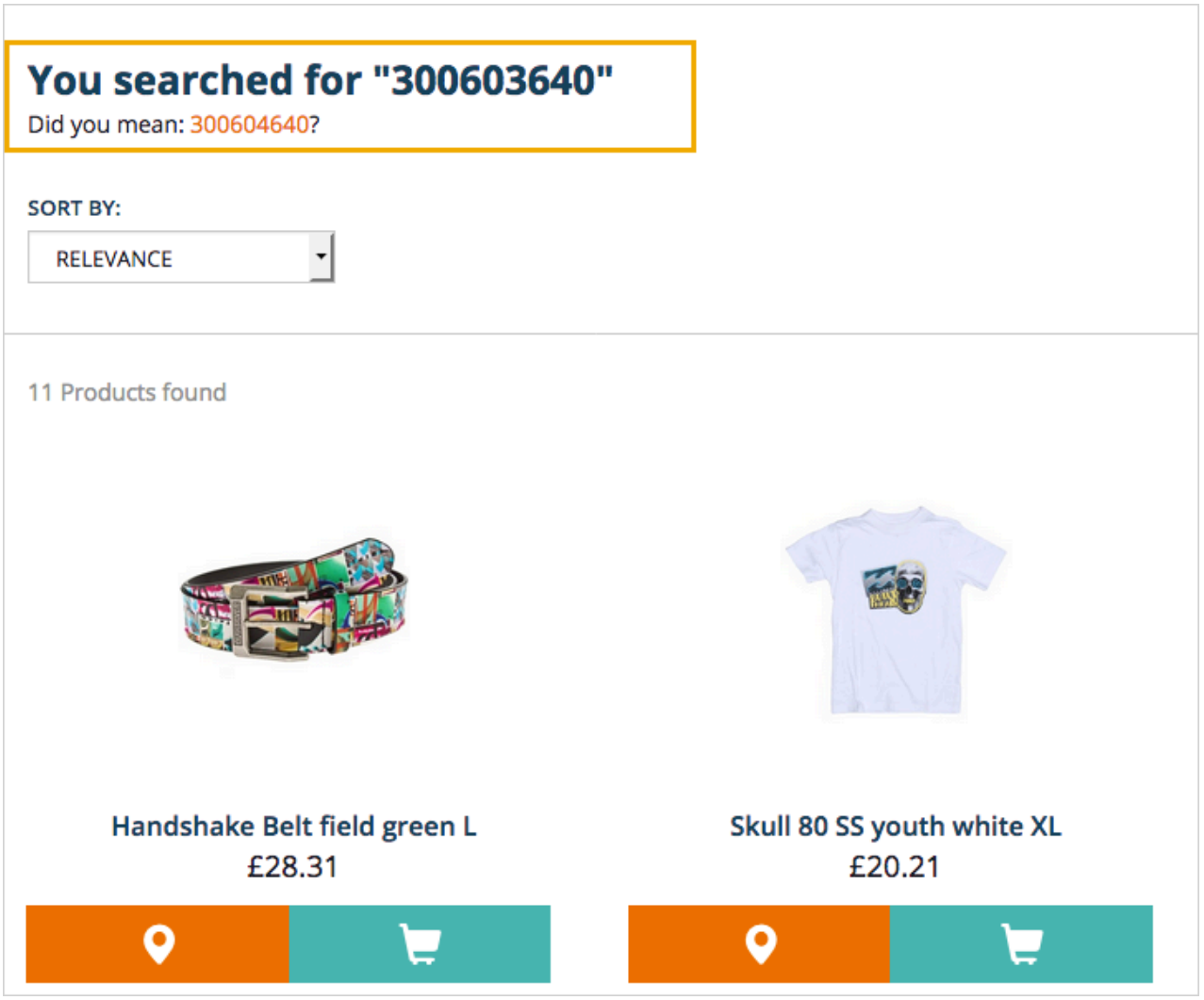
   The entered value means the number of changes between your input and the matched term.
10. Leave the Free Text Fuzzy Query Min Term Length at 0.
11. Boost the query by entering `110` in the Free Text Fuzzy Query Boost.

    The result should be as follows:



12. Go to the storefront and type a code with one wrong digit, for example 300603640

    The result include the item you searched for and a hint about the code:



Results

Thanks to wildcards and fuzzy query you can find your item easily even if you enter a part of the term or misspell it.

Facet Settings

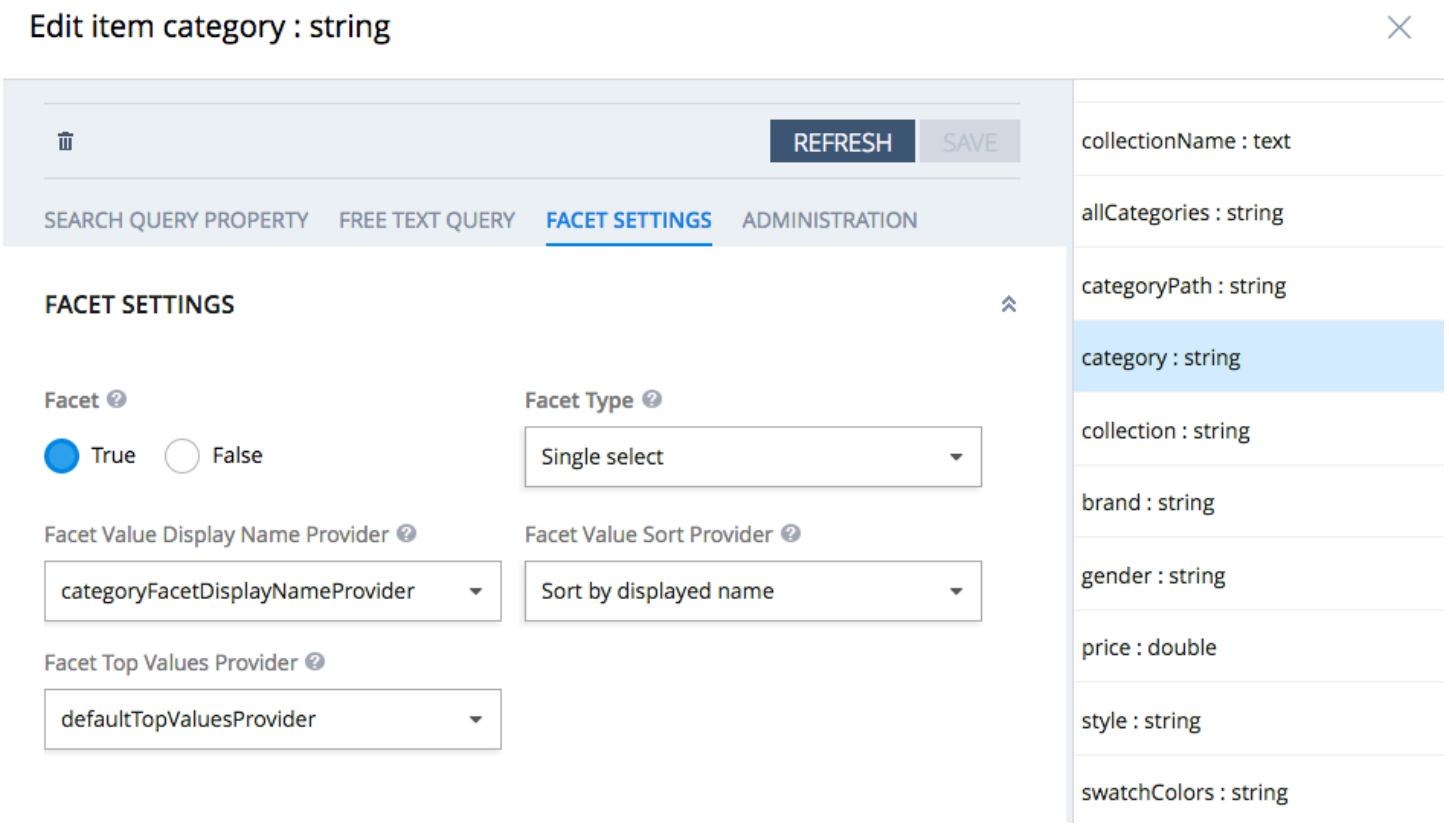
Facet settings allow you to decide if particular query property should be a facet, as well as add additional settings.

You can define the settings for each query property separately, using the **Facet Settings** section of the wizard. If you set a query property as facet, it will be displayed on the storefront.

Additionally, you can use it in further configurations, for example when configuring search settings in Adaptive Search.

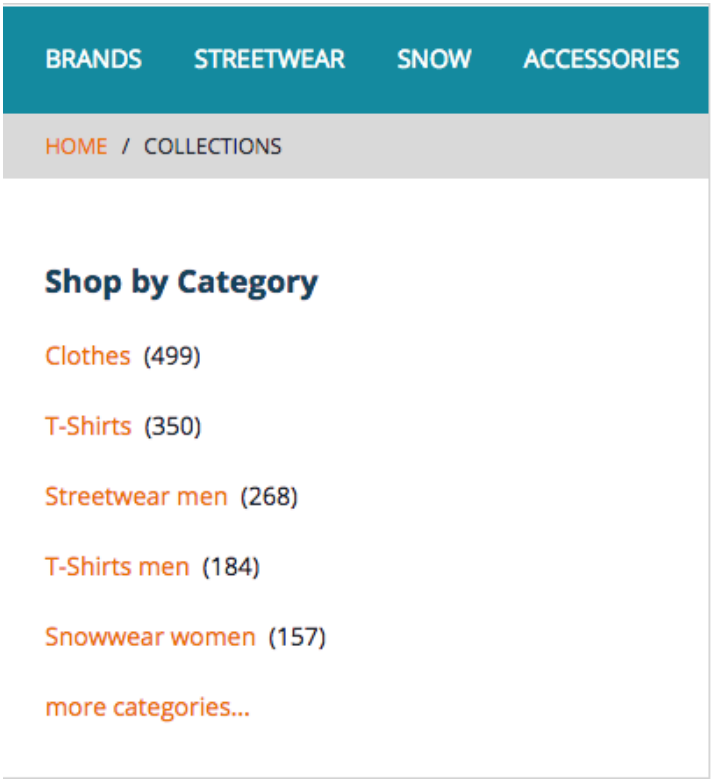
If a property cannot be a facet (for example because of its type) you will see a warning message.

Let's have a look at the facet settings for the **category** property.



Setting	Description
Facet	Determines if the query property should be displayed as facet.
Facet Type	The type of the facet: • Refine: selecting one facet value • Multiselect OR: selecting two or more facet values linked with OR condition • Multiselect AND: selecting two or more facet values linked with AND condition
Facet Value Display Name Provider	Retrieves the display name for a facet value.
Facet Value Sort Provider	Sorts the facet values according to a specified property such as name or the number of products. The sorting applies to the full list of the facet values and it might be different when the top values provider is used.
Facet Top Values Provider	Provides a list of facet values that have the larger number of items, even when the facets have many values. The remaining values are collapsed.

A resulting facet setting should be similar to the example below:



Configuring Variant Grouping

Variant grouping allows you to group similar entries according to a specific value, for example, color.

Context

Note

You can achieve variant grouping configuration using Adaptive Search Module as well.

After you configure your variant grouping, the product and all its variants will appear on the storefront. In the following image, you can see the result of grouping variants according to the color:

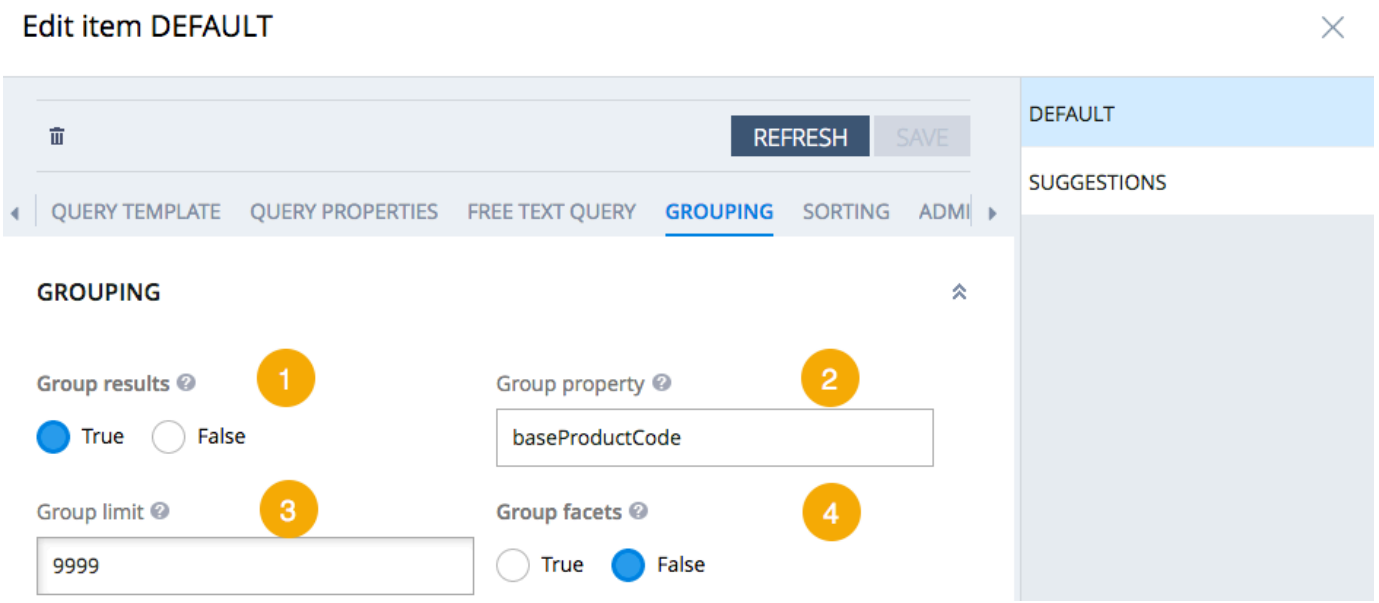
1. A preview of the product.
2. The variants of the same product, grouped using the color value.

Note

If templates are enabled, grouping should be set directly in the template. If it is not, grouping doesn't work properly. If no templates are defined, the fallback mechanism is applied. In this case, the configuration defined in Search and Navigation > Indexed Types is used. You can define it by clicking on an indexed type and opening the Grouping tab.

Procedure

1. Go to System > Search and Navigation > Solr Facet Search Configuration > Indexed Types.
2. Select the type that you want to define the grouping for.
3. In the editor, go to Indexed Type > Search Configuration and double-click the template.
4. Click the Grouping tab and configure the following items:



Item	Name	Description	Example
1.	Group results	Specifies whether the system groups and displays multiple products under a single search result.	True
2.	Group field name	The name of the field that the system uses to group the results. You specify the attribute that is used to identify which products should be grouped.	baseProductCode
3.	Group limit	Determines the number of variants to display.	100
4.	Group facets	If enabled, facets are computed for each group. If not they are computed for each document.	False

5. To define the value, according to which the system allows customers to select variants, in the `local.properties` file, define the `commerceservices.variant.rollup.property.<baseSiteid>=<variant-attribute-qualifier>` property.

   For example, if you want to group variants based on color, enter the property as follows: `commerceservices.variant.rollup.property.apparel=style`.

   This property determines the criteria that are used to show variants under the base product image. This corresponds to number 2 in the previous image. While the group field parameter determines the returned items, this parameter defines which parameter is used to display variants from the returned results.

Keyword Highlighting

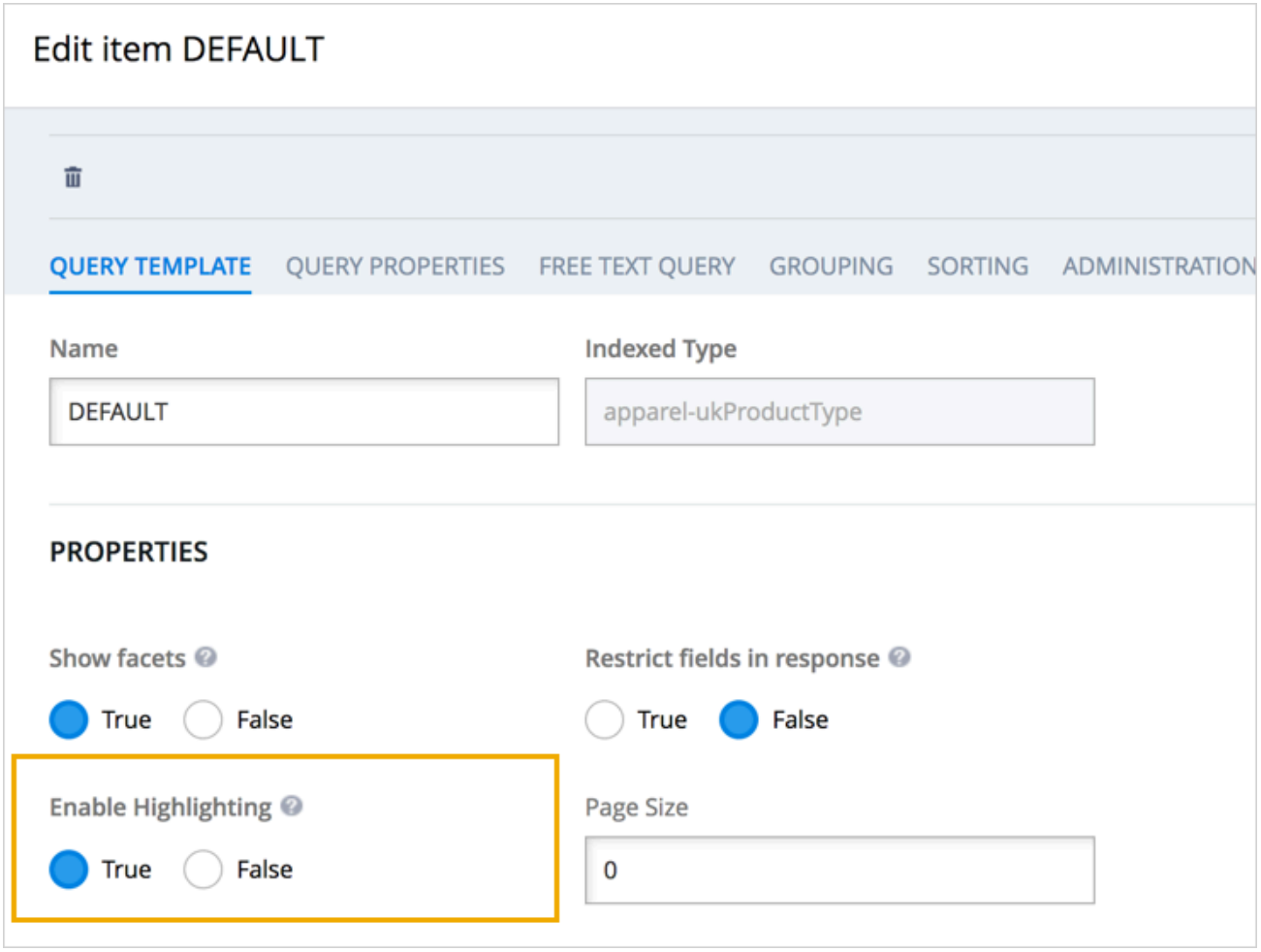
Keyword highlighting makes the keywords easily noticeable while searching.

Context

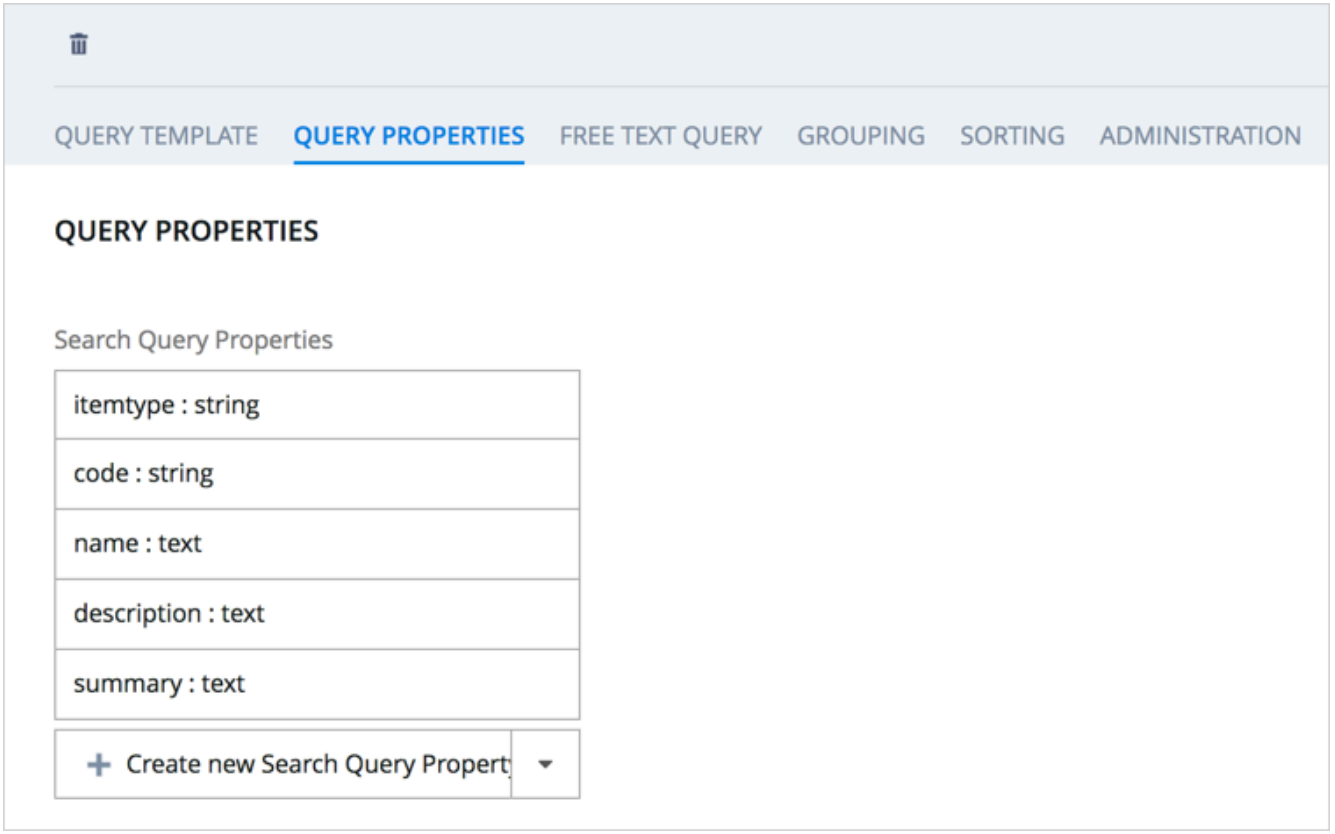
Perform the following steps to enable keyword highlighting.

Procedure

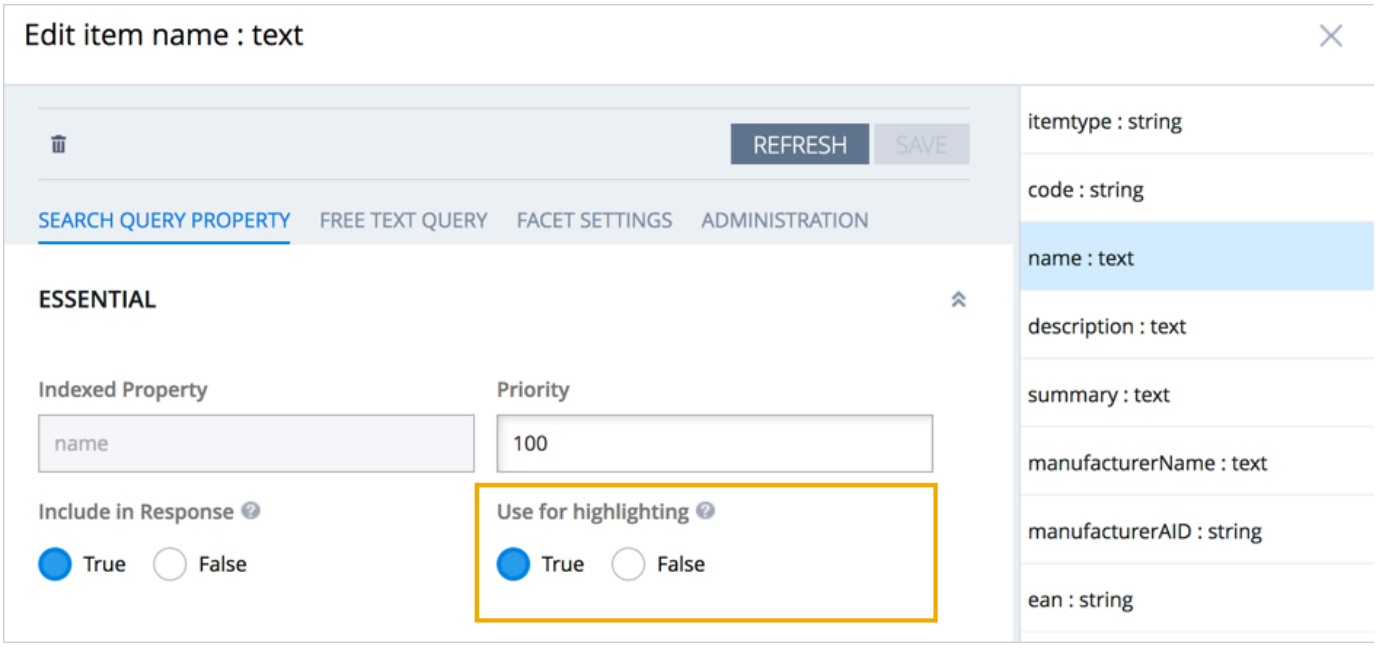
1. Navigate to Search Query Templates in Backoffice Administration Cockpit. When in doubt, follow the instructions in Access Search Query Templates.
2. Open the template.
3. First you need to enable the highlighting for the template. In the Query Template section, set the **Enable Highlighting** to true.



4. Save the settings.
5. Once the global highlighting is enabled, choose which query properties should be used for highlighting. Go to Query Properties section.



6. Let's select the name property, so that the keywords in the item name are highlighted. Double-click the property to edit it.
7. In the Search Query Property section set Use for highlighting to **True**.



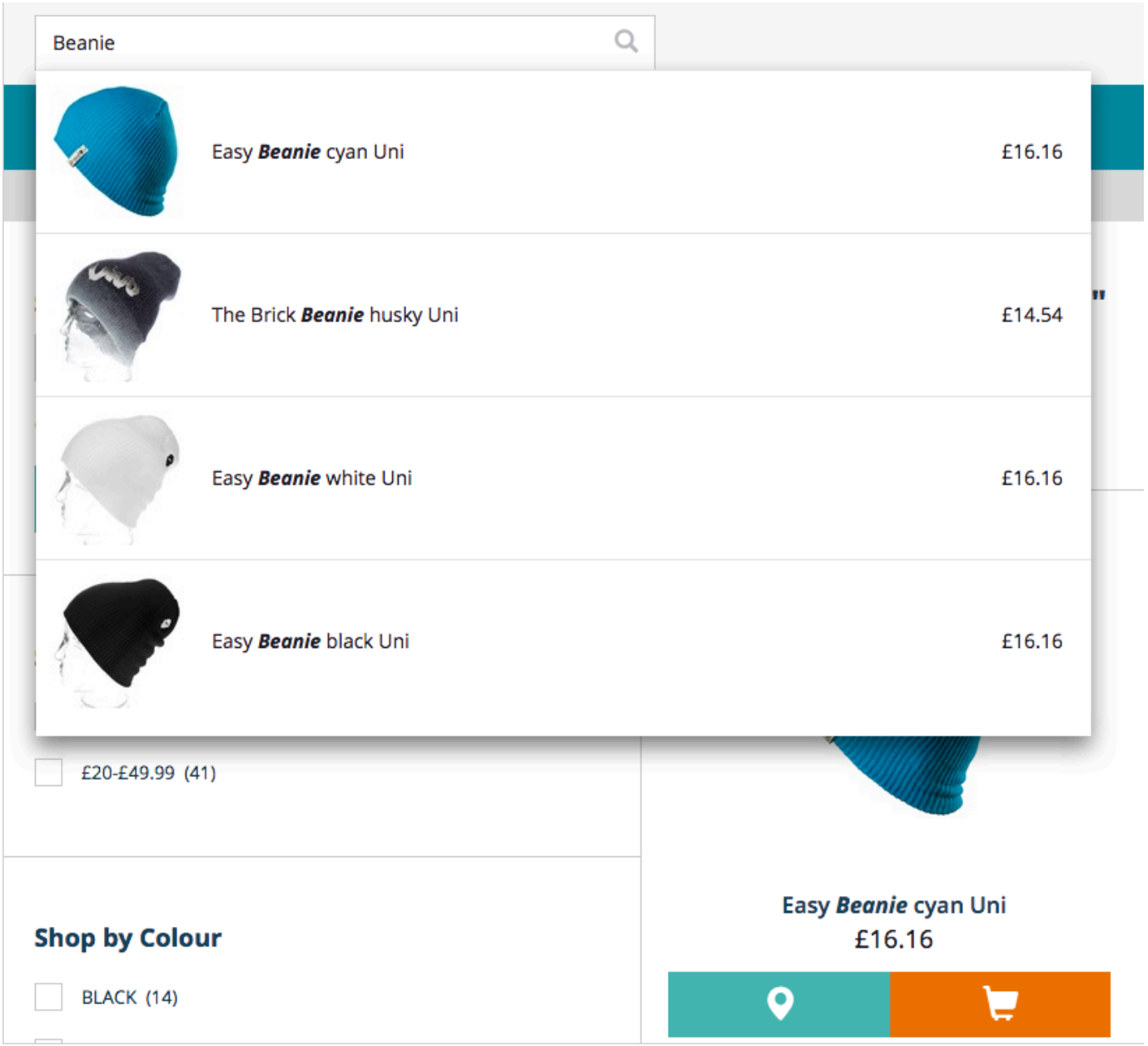
8. Save your settings.

ℹ Note

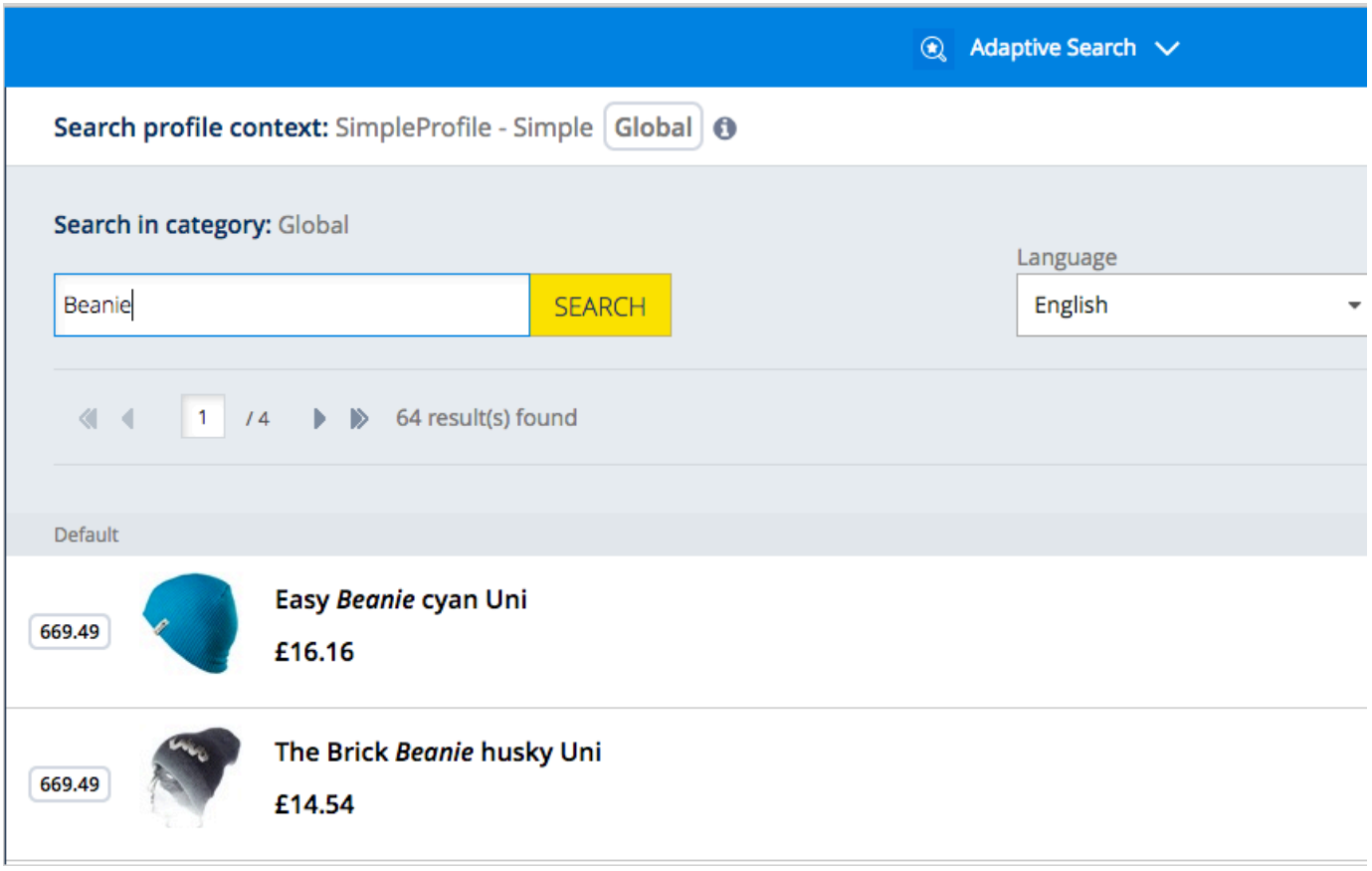
Once you close the current window, you return to the previous wizard windows. Remember to save the settings along the way, otherwise the changes will be lost.

9. Go to the storefront (for example `https://apparel-uk.local:9002/yacceleratorstorefront/en/`) and search for an item.

   The keywords are distinguished.



10. If you go to the Adaptive Search perspective and browse through the search results, you can also see the highlighted keywords.



SUGGESTIONS Template

The SUGGESTIONS is supported out of the box and allows you to configure the settings for the suggested items that appear when you type in the term in the search box.

Context

The template is not predefined. You define it in Backoffice Administration Cockpit. Using the template, you can set the following:

- Query properties that you want to see, such as name, code, or description of an item
- Phrase search, wildcards, and fuzzy queries,
- Sorting
- Highlighting

Procedure

1. Create the template using the Backoffice Administration Cockpit.
2. Configure the template by adding query properties and free text query settings.
3. Add sorting.
4. If you want your results to provided faster, you can restrict the fields in response.

Results

When searching for items in the storefront, the template returns suggested items in the search box, sorted according to price.

Create a Search Query Template

You create a search query template in Backoffice Administration Cockpit and set specific parameters to allow the storefront to recognize it to provide search suggestions.

Procedure

1. In the Backoffice Administration Cockpit, go to System > Search and Navigation > Solr Facet Search Configuration > Indexed Types.

   A list of all search configurations appears.

2. Select a search configuration.

   The editor area opens.

3. Select the Indexed Type tab.
4. In the Search Configuration section, click Create new Search Query Template.

   A new window appears.

5. Enter the name `SUGGESTIONS`.

   > **Note**
   > You must enter the name as above. If you do not, the storefront uses the DEFAULT template to search for suggestions. For more information, see the **Naming Convention** section in: Search Query Templates Technical Details.

6. Click Done.

   The template is created. You can now configure it.

Configure the Template

Context

In order for the template to work properly, you need to add the query properties you wish to be displayed and configure the free text query settings.

Results

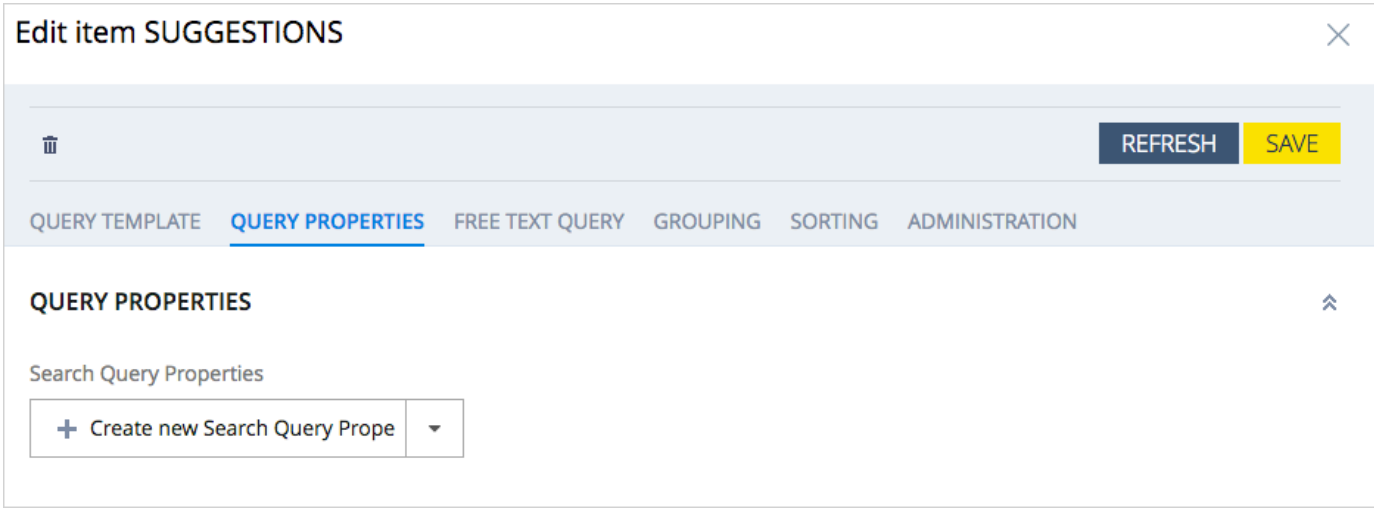
The query properties will be included in the term search. Continue the configuration with defining the Free Text Query settings.

Add Query Properties

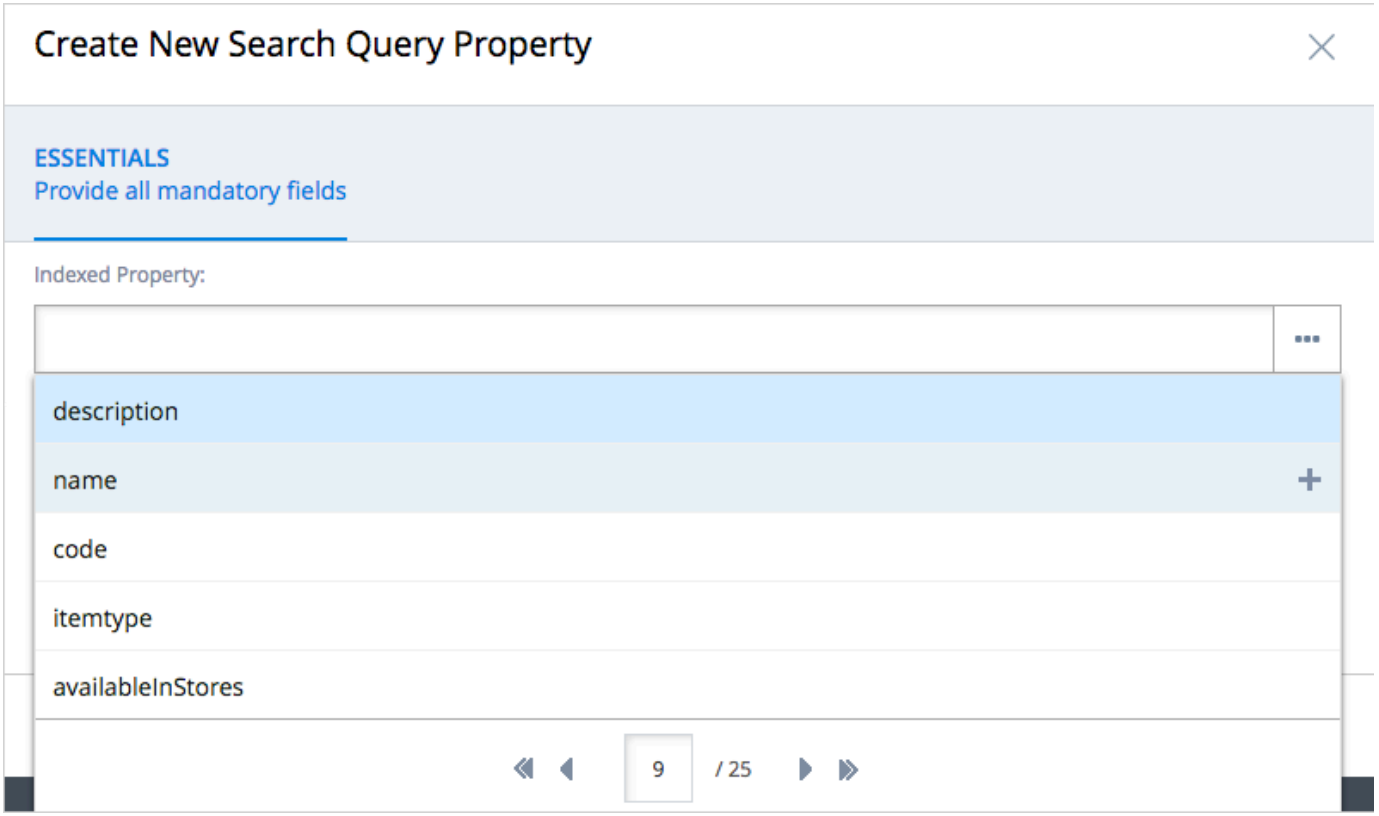
Add the query properties for your template.

Procedure

1. Navigate to the Query Properties and click Create new Search Query Properties



2. Select a desired property from the list or use the reference dialog to add it. For the purpose of this use case, use the name property.



Click Done to finish.

3. Using the same way, add more properties such as: code, description, price.

You can see them all on the list.

Define Free Text Query

To make sure the search is properly executed you need to set the free text query settings.

Procedure

1. In the template, navigate to Query Properties section.
2. Double click the name property.
3. Go to Free Text Query section.
4. Enable the Free Text Query by setting it to **True**.

Enabling the free text query allows the system to search for, locate and display the term.

5. Enable the Free Text Wildcard Query by setting it to **True**, and set the Type to **Prefix and Postfix**.

Because the wildcard query is enabled, the match the term even if you enter just a part of it.

6. Save your configuration. Perform Step 5 and Step 6 again for code and priceValue property. Remember to Save your changes.
7. Leave the other settings as they are. If you want to learn more about the free text query settings, see Define the Query Property Details.
8. Go to the storefront and search for a `belt`.

Suggestions are visible.

Sorting

Sorting enables you to prioritize the results, for example if you want to make some items more prominent, or if you want to order them.

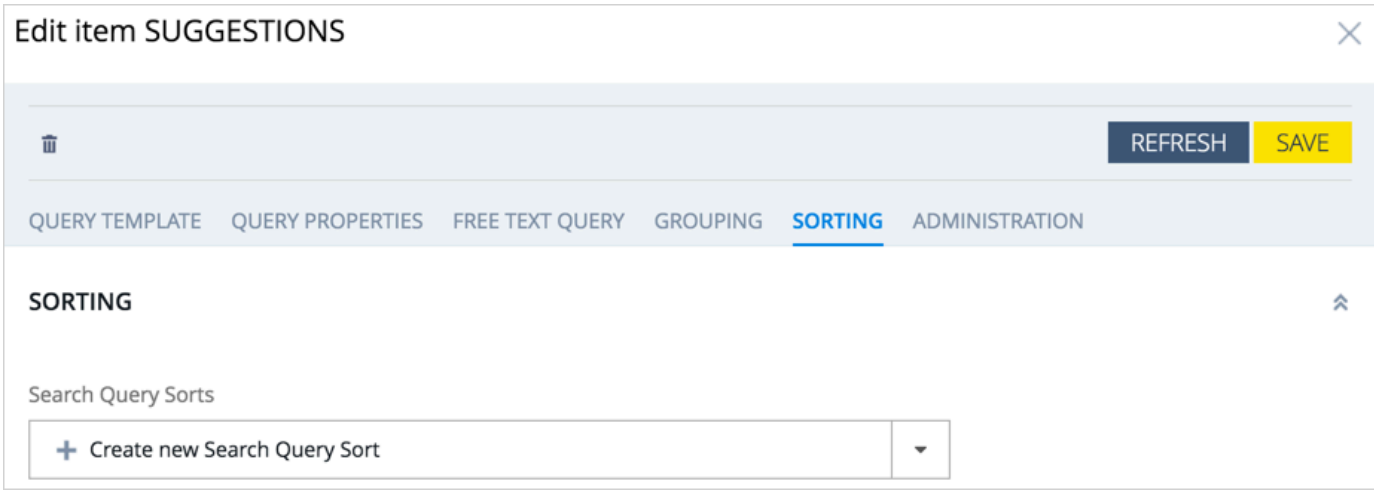
Context

Follow the steps below to sort the results according to price.

Procedure

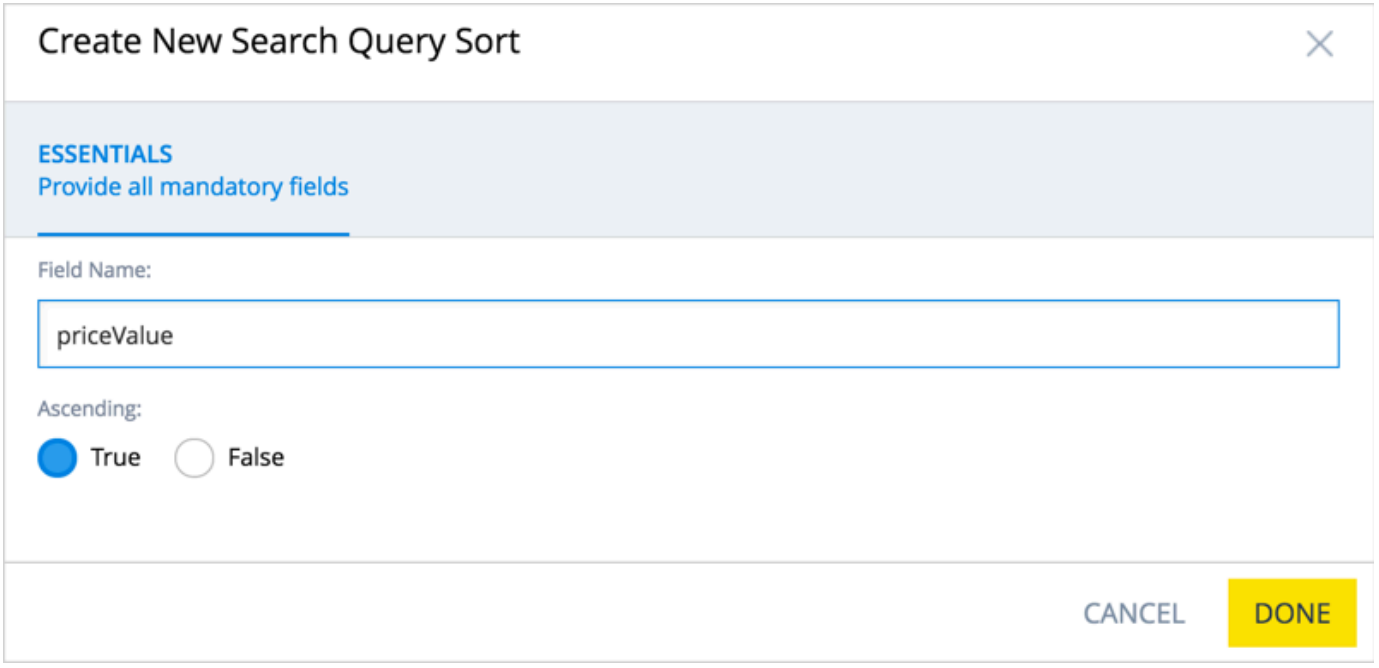
1. Go to Sorting tab.

   In this tab, you can define the sorting settings for your SUGGESTIONS template.



2. Click the Create a New Search Query Sort button.

   In the new window, enter priceValue in the field and set Ascending to **true**.



   Your suggestion list is now sorted according to price.

3. Go to `storefront` and start writing camera in the search field.

   You can see a list of results sorted according to the price value.

Results

You have configured a SUGGESTIONS template, by adding the search query properties and defining the sorting settings.

Restrict Fields in Response

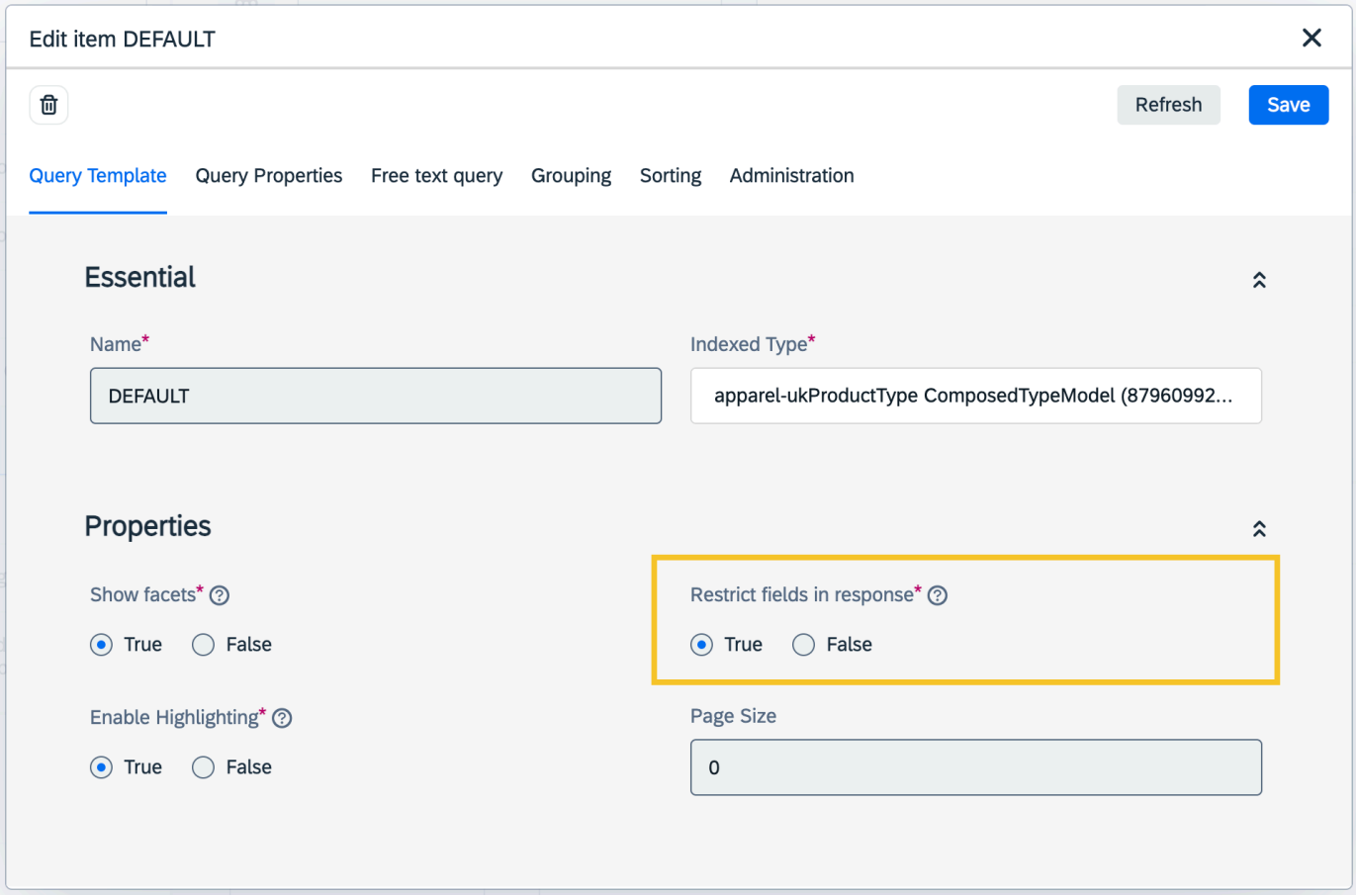
When configuring a Search Query Template, you can easily decide which search query properties should be included in the results visible on the storefront.

Context

For the purpose of this use case, let's assume that you have created a search query template with the following query properties: **name**, **priceValue**, **code** and **img-515Wx515H** to make sure the images are rendered. The results you get when writing **camera** in the search field should look similar to what you can see in the following figure:

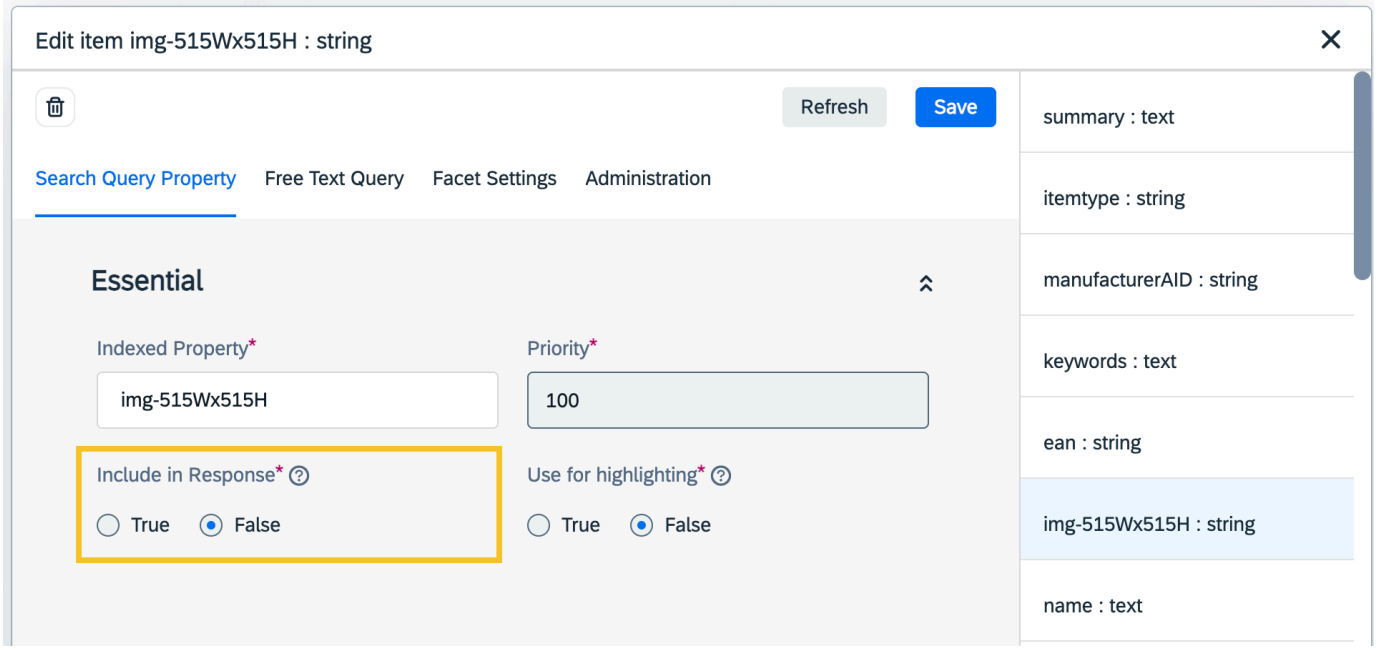
Procedure

1. Navigate to your SUGGESTIONS template. If you need help, see Access Search Query Templates
2. In the **Query Template** tab of the editor set **Restrict fields in response** to **true**. With this setting, only the properties with **Include in response** checked will be included in the response.



Remember to click **Save**, otherwise you changes will be lost.

3. Go to the **Query Properties** tab and double-click the **img-515Wx515H** property.
4. Set **Include in response** to **false**. Remember to **save** your changes after editing.



5. Close the query properties window to return to the main window. Save your changes.
6. Go to the storefront and start writing camera in the search field.

   The images are not included in the results.

Results

You can easily narrow down the number of fields displayed in the response.

Search Query Templates Technical Details

Technical information about the search query templates such as the naming convention or the use of the fallback mechanism.

Currently you can choose from the following two predefined search contexts to use the search query templates: DEFAULT and SUGGESTIONS. You can find the `enumtype` for the new search query context options in the `commerceservices-items.xml` file:

```
<enumtype code="SearchQueryContext" generate="true" autocreate="true" dynamic="false">
                        <description>Different sources of the search query</description>
                        <value code="DEFAULT" />
                        <value code="SUGGESTIONS" />
                </enumtype>
```

Naming Convention

A specific naming convention is used to determine, which query template is used for which context. That is why the same name is used for the search context and search query template. For example, to use a specific query template when searching for suggestions, you have to call it **SUGGESTIONS**, otherwise it won't work. If a template with a specific name is not found, the **DEFAULT** template is used. If there is no **DEFAULT** template, the fallback mechanism is used.

Legacy Mode

The legacy mode for search is set to **false** by default, and the serch query templates are avaialble for you to work with.

The search query is created based on the configuration in `SolrSearchQueryTemplate` and `SolrSearchQueryProperty`. If you want to work in a non-legacy mode, but you **don't want** to use query templates, make sure no templates are defined for a given `indexedType`, so the fallback mechanism can work.

Fallback Mechanism

The fallback mechanism was introduced to maintain compatibility for the users who are working with the previous versions. In such a case, the configuration is taken from the `SolrIndexedType` and `SolrIndexedProperty` types. The fallback mechanism is used when the legacy mode is off and there is no template defined for **any** context. It is possible to define a SUGGESTIONS template and use it for the suggestions, but use the fallback mechanism for DEFAULT context, meaning with respect to all other searches.

Further Steps

Follow the links below to get to know the search query templates data model and browse through the technical guides:

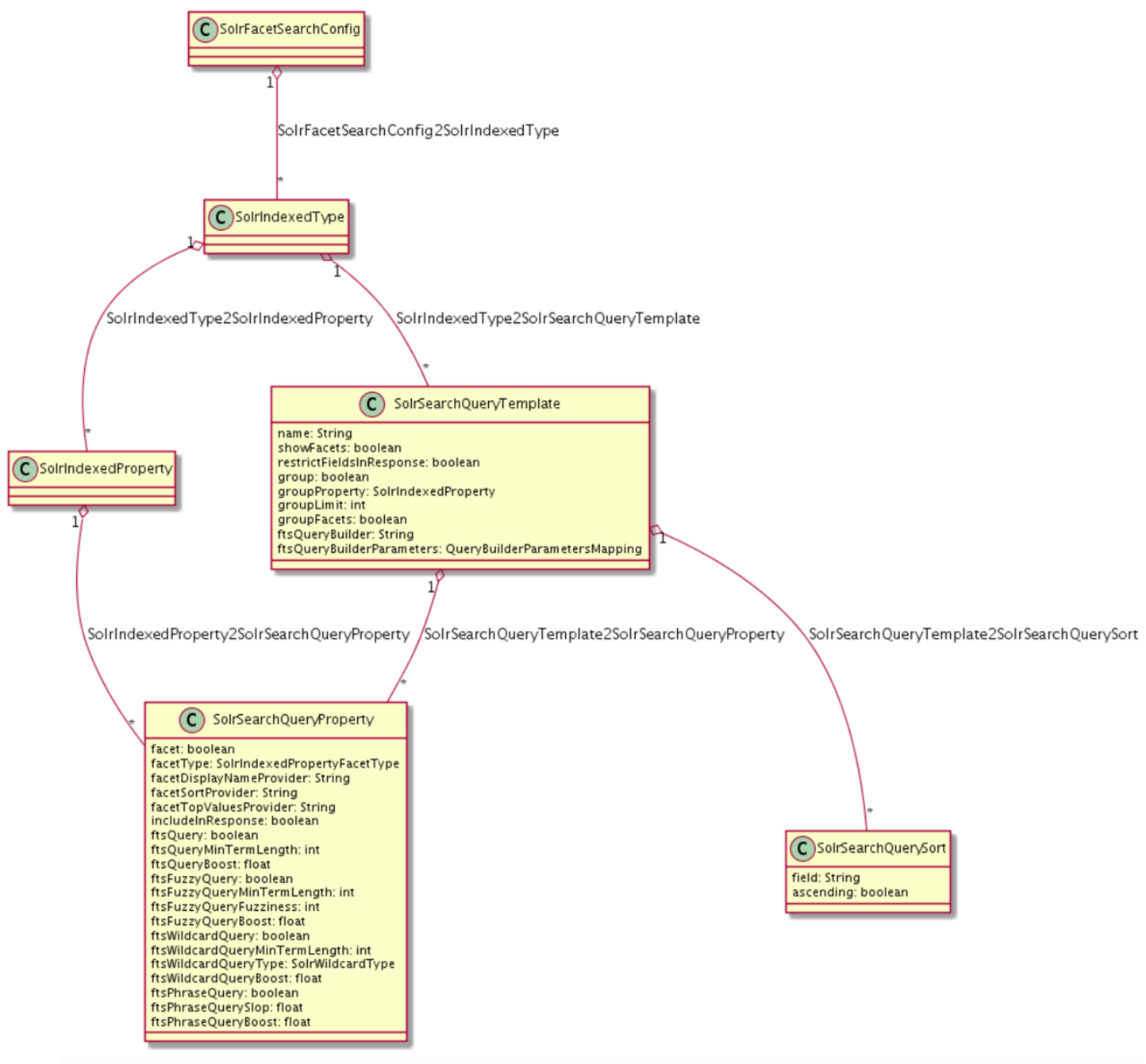
- Search Query Templates Data Model
- Search Query Templates Developer Guides

For details on managing the search query templates using the Backoffice Administration Cockpit see Search Query Templates in Backoffice Administration Cockpit.

Search Query Templates Data Model

Search query templates facilitate the option to have more than one configuration for free text search queries that can be used in different contexts such as storefront suggestions, main results, or the Backoffice Administration Cockpit..

The following figure illustrates the search query template data model:



`SolrSearchQueryTemplate`

Attribute	Name	Description
`name`	Name	A name chosen by the user for the template.
`showFacets`	Show facets	Enables or disables facets in the query or the response.
`restrictFieldsInResponse`	Restrict fields in response	If checked, only properties with `includeInResponse` checked will be included in the response.
`group`	Group results	If enabled, query results will be grouped .
`groupProperty`	Group property	The name of the property by which the results are grouped.
`groupLimit`	Group limit	The number of results to return for each group.
`groupFacets`	Group facets	if enabled, facets are computed per group, otherwise per document.
`pageSize`	Page Size	

Attribute	Name	Description
`ftsQueryBuilder`	Free text query builder	The `id` of a spring bean that implements the `FreeTextQueryBuilder` interface (for example `defaultFreeTextQueryBuilder`, `disMaxFreeTextQueryBuilder`, `multiFieldFreeTextQueryBuilder`).
`ftsQueryBuilderParameters`	Free tex query builder parameters	Parameters of the free text query builder.

`SolrSearchQueryProperty`

Attribute	Name	Description
`priority`	Priority	
`includeInResponse`	Include in Response	If selected, the field will be returned within a query response
`facet`	Facet	Check if the property should be classified as facet
`facetType`	Facet Type	Facet Type (`Refine`, `MultiSelectAnd`, `MultiSelectOr`)
`facetDisplayNameProvider`	Facet Value Display Name Provider	Provider for facet value display names
`facetSortProvider`	Facet Value Sort Provider	Provider for sorting facet values
`facetTopValuesProvider`	Facet Top Values Provider	Provider for selecting top facet values
`ftsQuery`	Free Text Query	If enabled a free text query will be performed on this property
`ftsQueryMinTermLength`	Free Text Query Min Term Length	If a term length is below this parameter, search term will not be taken into consideration
`ftsQueryBoost`	Free Text Query Boost	Multiplicative boost factor to increase or decrease the importance in the query
`ftsFuzzyQuery`	Free Text Fuzzy Query	If enabled a free text fuzzy query will be performed on this property
`ftsFuzzyQueryMinTermLength`	Free Text Fuzzy Query Min Term Length	If a term length is below this parameter, search term will not be taken into consideration
`ftsFuzzyQueryFuzziness`	Free Text Fuzzy Query Fuzziness	The required similarity. The value can be 0, 1 or 2, only terms with a higher similarity will be matched
`ftsFuzzyQueryBoost`	Free Text Fuzzy Query Boost	Multiplicative boost factor to increase or decrease the importance in the query
`ftsWildcardQuery`	Free Text Wildcard Query	If enabled a free text wildcard query will be performed on this property
`ftsWildcardQueryMinTermLength`	Free Text Wildcard Query Min Term Length	If a term length is below this parameter, search term will not be taken into consideration
`ftsWildcardQueryType`	Free Text Wildcard Query Type	Where the multiple character wildcard will be applied
`ftsWildcardQueryBoost`	Free Text Wildcard Query Boost	Multiplicative boost factor to increase or decrease the importance in the query
`ftsPhraseQuery`	Free Text Phrase Query	If enabled a free text phrase query will be performed on this property
`ftsPhraseQuerySlop`	Free Text Phrase Query Slop	The number of other words permitted between words in query phrase, if zero, then this is an exact phrase search
`ftsPhraseQueryBoost`	Free Text Phrase Query Boost	Multiplicative boost factor to increase or decrease the importance in the query

`SolrSearchQuerySort`

Attribute	Name	Description
`field`	Field Name	The field according to which the sorting is executed. For example adding **`priceValue`** as the name results in the result list being sorted according to price.
`ascending`	Ascending	The order according to which the sorting is executed.

Search Query Templates Developer Guides

Learn how to easily create, manage and customize the search query templates.

Implement a Default Search Query Template

Learn how to set up and configure a DEFAULT search query template.

Procedure

1. You can find an example of a DEFAULT template under `/apparelstore/resources/apparelstore/import/coredata/stores/apparel-uk/solr.impex`.

   In order to use it, update the settings in the following way:

   - from `SolrIndexedType` to `SolrSearchQueryTemplate` and
   - from `SolrIndexedProperty` to `SolrSearchQueryProperty`.

   For details on the data model see: Search Query Templates Data Model.

2. For the search query templates to be working properly, you need to disable the search legacy mode for searching.

   You can do it either by modifying the `/apparelstore/resources/apparelstore/import/coredata/stores/apparel-uk/solr.impex` file in the following way:

   ```
   $searchConfigName=apparel-ukPageSize

   INSERT_UPDATE SolrSearchConfig;description[unique=true];legacyMode
   ;$searchConfigName;false
   ```

   or using the Backoffice Administration Cockpit. See: Search Query Templates in Backoffice Administration Cockpit.

3. The DEFAULT template is now ready for you to use. You can edit it using theBackoffice Administration Cockpit

Implement a Suggestions Query Template

Learn how to create and manage a SUGGESTIONS query template.

You can configure the SUGGESTIONS query template in the same way as the DEFAULT template.

You can use the sample data provided below to configure your SUGGESTIONS template. Simply import the impex file using the SAP Commerce Administration Console (Administration Console)

```
$solrIndexedType=apparel-ukProductType

# Search query template
INSERT_UPDATE SolrSearchQueryTemplate;name[unique=true];indexedType(identifier)[unique=true];ftsQueryBuilder
;SUGGESTIONS;$solrIndexedType;defaultFreeTextQueryBuilder

# Non-facet search query properties
INSERT_UPDATE SolrSearchQueryProperty;indexedProperty(name, solrIndexedType(identifier))[unique=true];searchQuery
;itemtype:$solrIndexedType ; ; ; ; ; ; ; ; ; ;
;code:$solrIndexedType ; ; ;TRUE;90 ; ; ;TRUE; ; ;POSTFIX;45;3;
```

```
;name:$solrIndexedType ; ; ;TRUE;100;TRUE;50;TRUE;25 ; ; ; ;
;priceValue:$solrIndexedType ; ; ; ; ; ; ; ; ; ; ;
;img-515Wx515H:$solrIndexedType ; ; ; ; ; ; ; ; ; ; ;
;img-300Wx300H:$solrIndexedType ; ; ; ; ; ; ; ; ; ; ;
;img-96Wx96H:$solrIndexedType ; ; ; ; ; ; ; ; ; ; ;
```

If you want to learn how to create and configure a SUGGESTIONS search query template using the Backoffice Administration Cockpit, see Search Query Templates in Backoffice Administration Cockpit.

Implement a Custom Search Query Template

Context

Because the search query template feature is very flexible and adjustable, you can easily create your own custom template. As an example, you are going to create a search context, which will be used when browsing categories. Next, you will create a **CATEGORY** search query template with grouping products feature enabled. As a result, you will see grouped products in the Category Page and not grouped in the Search Page (because the DEFAULT context is used for the Search Page).

Procedure

1. Create new context by adding the CATEGORY element to the `SearchQueryContext` enum type in the `commerceservices-items.xml` file.

   ```
   <enumtype code="SearchQueryContext" generate="false" autocreate="false" dynamic="false">
       <value code="CATEGORY" />
   </enumtype>
   ```

2. Update the system using the `ant updatesystem` command.
3. You need to call the `getProductSearchFacade().categorySearch` method with the newly created context.

   In the `AbstractCategoryPageController.java` class replace

   ```
   searchPageData = getProductSearchFacade().categorySearch(categoryCode);
   ```

   with

   ```
   searchPageData = getProductSearchFacade().categorySearch(categoryCode, SearchQueryContext.CATEGORY);
   ```

   The result should look like below:

   ```
   ...
   public void doSearch()
           {
               showCategoriesOnly = false;
               if (searchQueryData.getValue() == null)
               {
                   // Direct category link without filtering
                   searchPageData = getProductSearchFacade().categorySearch(categoryCode, SearchQueryContext.C
                   if (categoryPage != null)
                   {
                       showCategoriesOnly = !categoryHasDefaultPage(categoryPage)
                               && CollectionUtils.isNotEmpty(searchPageData.getSubCategories());
                   }
               }
               else
               {
                   // We have some search filtering
                   if (categoryPage == null || !categoryHasDefaultPage(categoryPage))
                   {
                       // Load the default category page
   ```

```
                    categoryPage = getDefaultCategoryPage();
                }

                final SearchStateData searchState = new SearchStateData();
                searchState.setQuery(searchQueryData);

                final PageableData pageableData = createPageableData(page, getSearchPageSize(), sortCode, s
                searchPageData = getProductSearchFacade().categorySearch(categoryCode, searchState, pageabl
            }
        }
...
```

4. Create a new search query template, which will be used in the category context. Bear in mind the naming convention - the name of the template needs to be the same as the name of the search context.

   Use the SAP Commerce Administration Console(Administration Console) to import the following impex file:

```
$solrIndexedType=apparel-ukProductType
# Search query template
INSERT_UPDATE SolrSearchQueryTemplate;name[unique=true];indexedType(identifier)[unique=true];ftsQueryBuilde
;CATEGORY;$solrIndexedType;defaultFreeTextQueryBuilder;true;baseProductCode:$solrIndexedType;999

# Non-facet search query properties
INSERT_UPDATE SolrSearchQueryProperty;indexedProperty(name, solrIndexedType(identifier))[unique=true];searc
;itemtype:$solrIndexedType               ;    ;   ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;code:$solrIndexedType                  ;    ;   ;TRUE;90 ;    ;  ;TRUE;    ;  ;POSTFIX;45;3;
;name:$solrIndexedType                  ;   ;    ;TRUE;100;TRUE;50;TRUE;25 ;  ;       ;  ;
;description:$solrIndexedType           ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;manufacturerName:$solrIndexedType      ;   ;    ;TRUE;80 ;TRUE;40;TRUE;20 ;  ;       ;  ;
;ean:$solrIndexedType                   ;    ;   ;TRUE;100;    ;  ;TRUE;   ;  ;POSTFIX;50;3;
;priceValue:$solrIndexedType            ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;keywords:$solrIndexedType              ;   ;   ;TRUE;40 ;TRUE;20;TRUE;10 ;  ;       ;  ;
;img-515Wx515H:$solrIndexedType         ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;img-300Wx300H:$solrIndexedType         ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;img-96Wx96H:$solrIndexedType           ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;img-65Wx65H:$solrIndexedType           ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;img-30Wx30H:$solrIndexedType           ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;url:$solrIndexedType                   ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;stockLevelStatus:$solrIndexedType      ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;inStockFlag:$solrIndexedType           ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;pickupAvailableFlag:$solrIndexedType   ;   ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
;baseProductCode:$solrIndexedType        ;  ;    ;    ;   ;    ;  ;    ;   ;  ;       ;  ;
# Category search query fields
INSERT_UPDATE SolrSearchQueryProperty;indexedProperty(name, solrIndexedType(identifier))[unique=true];searc
;categoryName:$solrIndexedType                  ;    ;   ;true;40;true;20;true;10;
;brandName:$solrIndexedType                           ;    ;   ;    ;  ;    ;  ;    ;  ;
;collectionName:$solrIndexedType                    ;    ;   ;    ;  ;    ;  ;    ;  ;
# Category search query facets
INSERT_UPDATE SolrSearchQueryProperty;indexedProperty(name, solrIndexedType(identifier))[unique=true];searc
;allCategories:$solrIndexedType  ;;;Refine;
;categoryPath:$solrIndexedType  ;;;Refine;
;category:$solrIndexedType      ;;;Refine;
;collection:$solrIndexedType        ;;;Refine;
;brand:$solrIndexedType             ;;;Refine;
;gender:$solrIndexedType            ;;;Refine;
# Other facet properties
INSERT_UPDATE SolrSearchQueryProperty;searchQueryTemplate(name, indexedType(identifier))[unique=true][defau
;; price:$solrIndexedType             ; ;MultiSelectOr ; 4000;
;; style:$solrIndexedType             ; ;MultiSelectOr ;-1   ;
;; swatchColors:$solrIndexedType      ; ;MultiSelectOr ; 2500;colorFacetDisplayNameProvider
;; size:$solrIndexedType              ; ;MultiSelectOr ; 2000;
```

```
;; allPromotions:$solrIndexedType     ; ;MultiSelectOr ; 0   ;promotionFacetDisplayNameProvider
;; availableInStores:$solrIndexedType ; ;MultiSelectOr ;10000;apparelPointOfServiceFacetDisplayNameProvider
```

5. Go to `https://apparel-uk.local:9002/yacceleratorstorefront/` to verify that products are being grouped while you browse the categories.

Manage Search Configurations

Create and manage your search, indexing, and server configurations using the Backoffice Administration Cockpit.

Use Case

The user wants to define configurations for search, indexing and Solr server. Additionally the user would like to define keywords, synonyms and stopwords for each configuration.

Features

Creating Facet Search Configurations

Create facet search configurations including search and indexer configurations as well as details such as keywords, synonyms, stopwords and keyword redirects.

Defining and Configuring Indexed Types

Define the items you want to be indexed along with their properites and queries.

Creating Ranges

Define the value range sets for given data types.

Dependencies

There are no specific dependencies for using search query templates.

Create Facet Search Configuration

Use the facet search config node in the explorer tree to create a new facet search configuration and customize it according to your needs.

Have a look at the following user guides to successfully create and customize your configurations.

1. First, learn how to create a new facet search configuration and get to know advanced configuration options: Add a Search Configuration.
2. Once you have created your new configuration, or you want to customize the exisiting one that was already defined, you can use the instructions provided in the the following guides:
    - Details on configuring the fallback language: Configure Fallback Language for Solr Indexer.
    - Details on configuring keyword redirects: Configure Keyword Redirects.
    - Details on configuring synonyms: Configure Synonyms.
    - Details on configuring stopwords: Configure Stopwords.
    - Details on defining search and indexer configuration: Define Search and Indexer Configuration.
    - Details on updating solr index: Update Solr Index.

Add a Search Configuration

Learn how to create a new facet search configuration using Backoffice Administration Cockpit.

Procedure

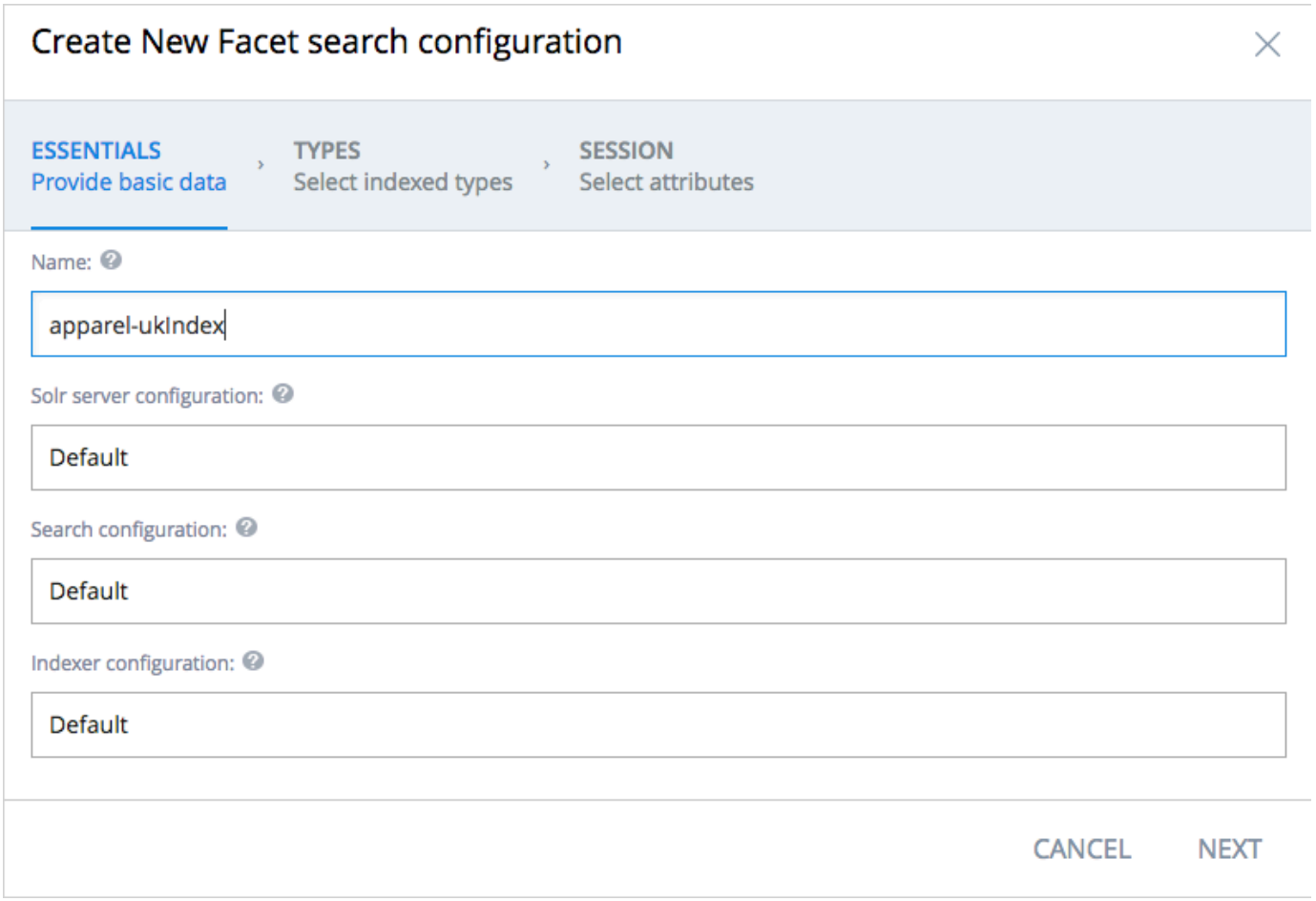
1. Navigate to System > Search and Navigation > Solr Facet Search Configuration > Facet Search Configurations. You can also use the search field to limit the results.

   A list of available configurations (if any) appears on the right.

2. Click the + button at the top of the page.

   A new window opens.

3. In the Essentials section, fill in the following fields



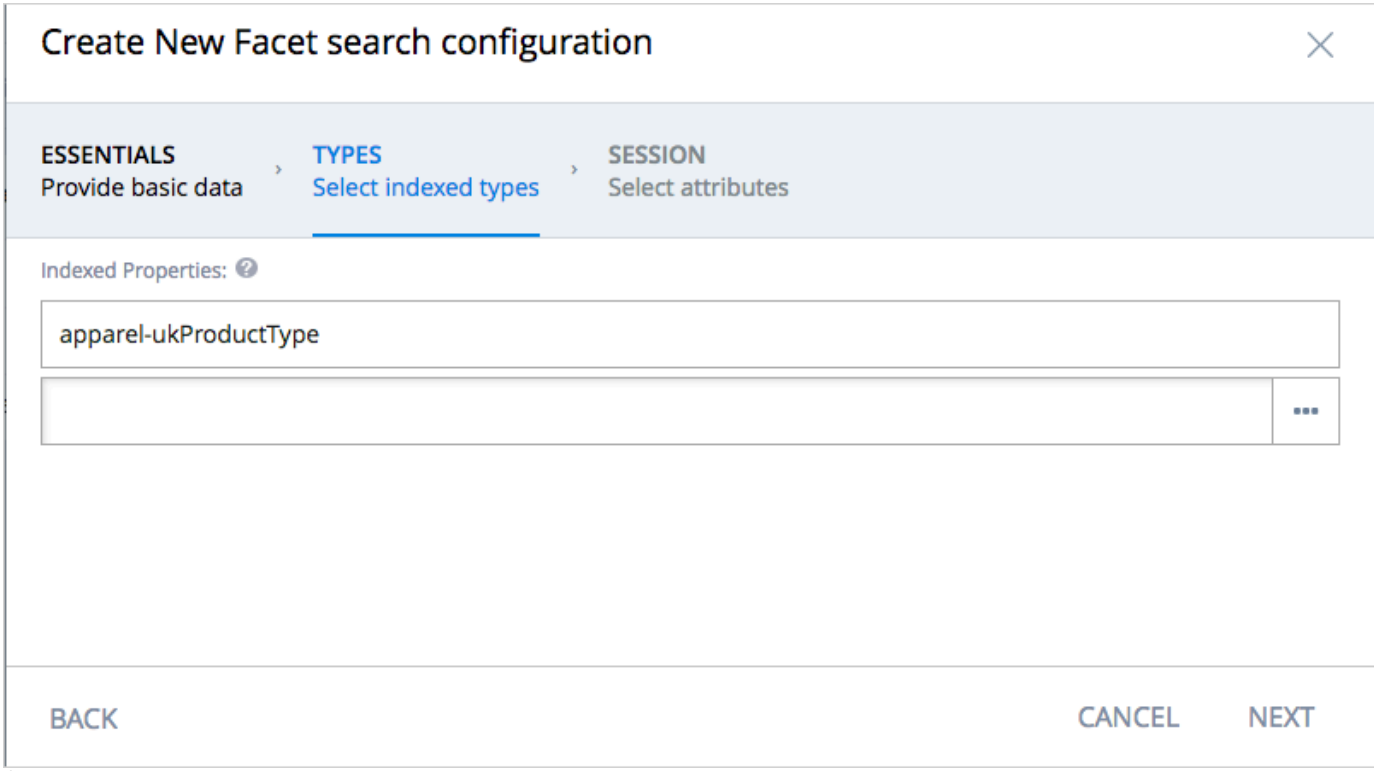
   - **Name**: A **unique** name for your configuration.
   - **Solr server configuration**: A solr server configuration to be used. Available values are **Default**, **Local Cloud**, **Local Standalone**.
   - **Search configuration**: A chosen search configuration. The **Default** configuration is available out of the box.
   - **Indexer configuration**: An indexer configuration to be used. Available values are **Default**, **Direct**, **Two-Phase**.

   > **Note**
   >
   > Default configurations are suitable for most developer environments. For test and production environment you should adjust the default configurations or create new ones.

   For details on creating solr server configuration and indexer configuration see: Define Search and Indexer Configuration.

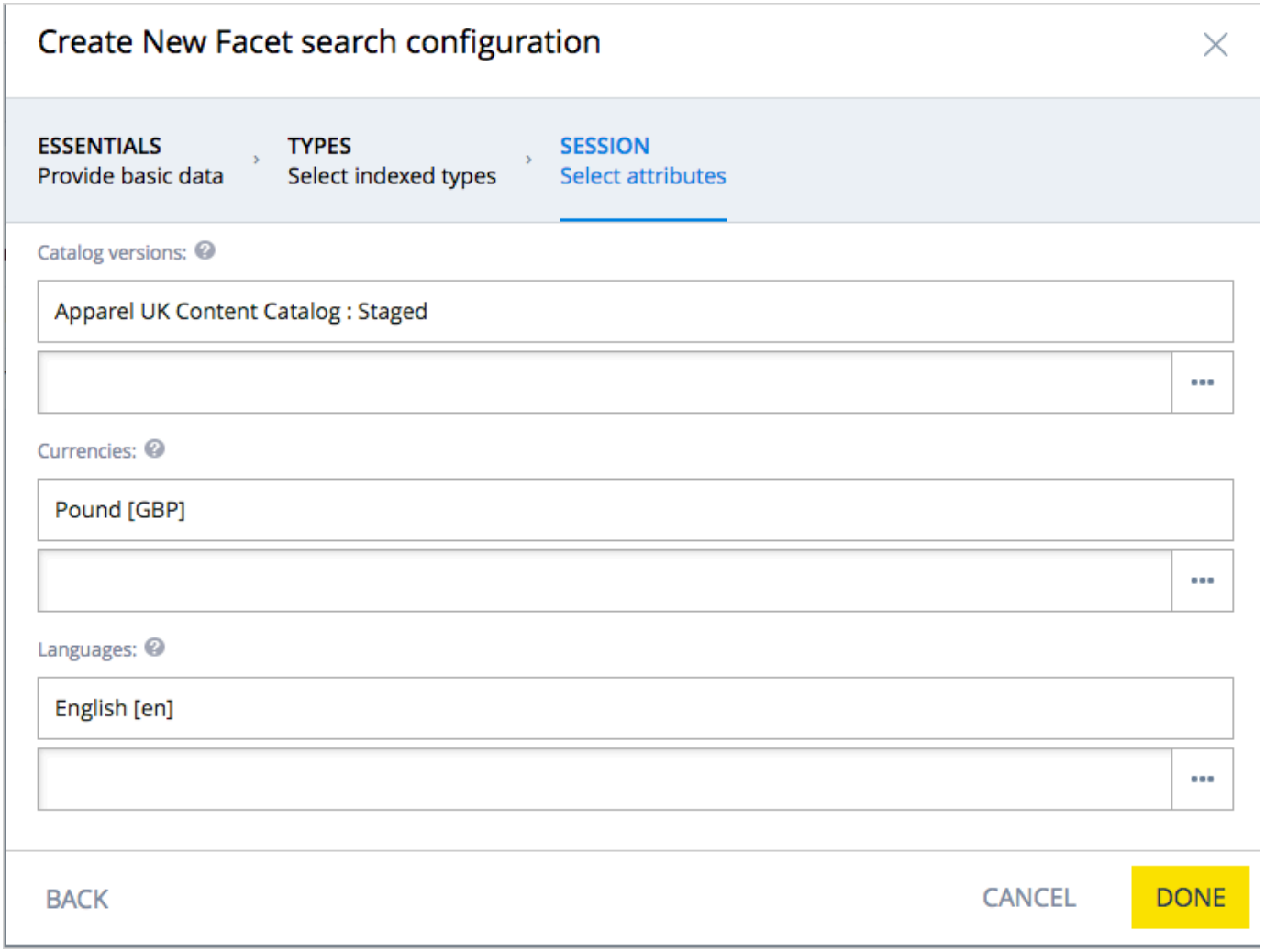
   Click **Next** to proceed.

4. In the Types section, select the item types to be indexed.



Click Next to proceed.

5. In the Session section, fill in the following fields:



   - **Catalog versions**
   - **Currencies**
   - **Languages**

6. Click Done to finish

Your configuration is now added to the list. You can access it at any time to introduce the changes.

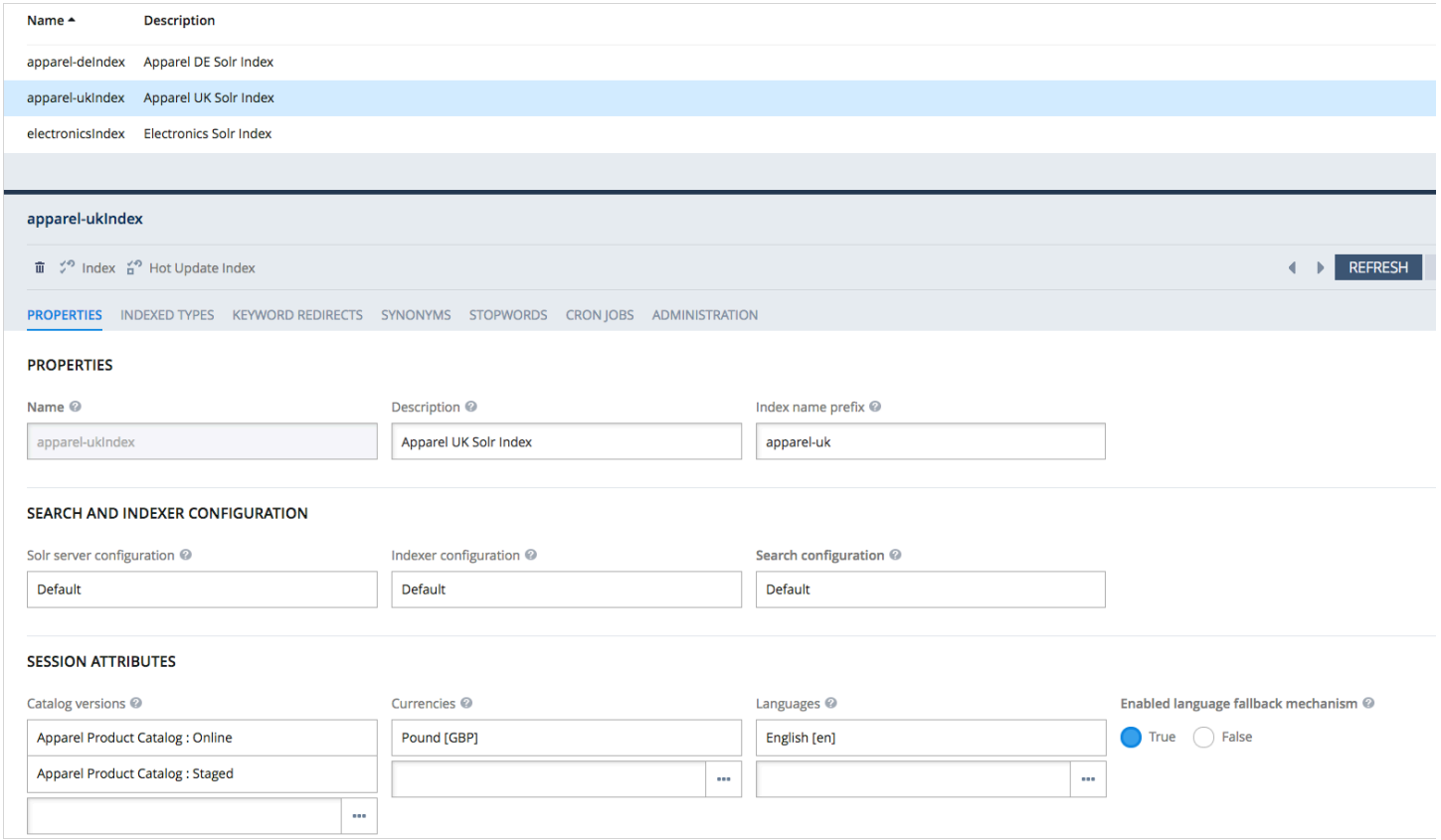
Advanced Customization of Solr Facet Search Configuration

Configurable Items

The data model of the Solr facet search configuration includes the main element - the `SolrFacetSearchConfig` object. It represents a single configuration and stays in relation to other items which, when combined together, allow you to define different aspects of the configuration such as related languages, currencies or catalog versions.

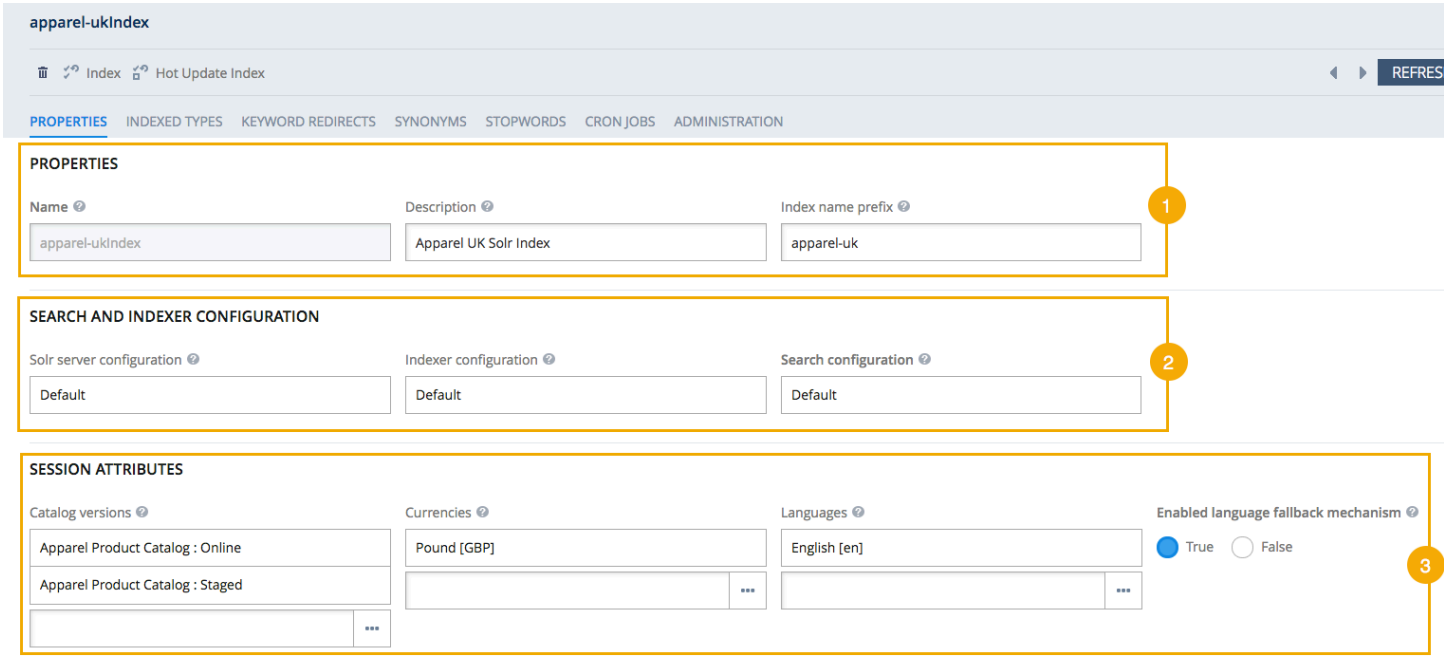
Customization Options

The configuration editor allows you to set up the configuration for the Solr facet search. It contains several tabs with configurable items described in details in the sections below.



Properties

Using the Properties tab you can view your configuration, modify it and define additional settings.



1. **Properties**: Main properties of your configuration, such as name, description, and index name prefix.

2. **Search and Indexer Configuration**: In this section you can define and customize the following configurations:

Solr server configuration

The Solr server configuration editor enables you to set up necessary configuration options.

Indexer configuration

The Indexer configuration editor enables you to configure several options of the Indexer.

Search configuration

The Search configuration allows you to define the search-related settings such as the number of results displayed per page or the sorting.

For details on how to define and customize each of these configurations see: Define Search and Indexer Configuration

3. **Session attributes**: Additional settings for your configuration, including: catalog versions, languages, or currencies. Here you can also enable the fallback mechanism: Manage Solr Ranges.

Indexed types

You can use the Indexed types tab to add or modify the item types to be indexed. Select a predefined Indexed type or create a new type by right-clicking the Item types table and selecting the appropriate action from the context menu. The Item corresponding to the indexed type has its own explorer tree node in Backoffice Administration Cockpit. It's called **Indexed types** and is present under the Facet Search directory.

Keyword redirects

The Keyword redirects tab enables users to define the words used to redirect them to the certain search results like URL targets or items. These keyword redirects are defined for specific facet search configuration and can be created for several languages separately.

For details on defining keyword redirects see: Configure Keyword Redirects

Synonyms

The Synonyms tab enables users to create additional word to be used as a search parameter. These synonyms are defined for specific facet search configuration and can be created for several languages separately.

For details on defining synonyms see: Configure Synonyms.

Stopwords

For indexes with lots of common words like the, a, these words make the index large and slow down phrase queries. A simple solution is to filter common words out of fields where they show up often. The Stopwords tab enables users to add words to be used as filtering parameters. These stop words are defined for specific facet search configuration and can be created for several languages separately.

For details on defining stopwords see: Configure Stopwords.

Cron jobs

Cronjobs assigned to a particular search configuration. For an example of setting a cronjob see: Update Solr Index with CronJob.

Define Search and Indexer Configuration

Learn how to use the Backoffice Administration Cockpit to easily define and modify the Solr configuration.

Context

The Backoffice Administration Cockpit gives you the opportunity to access and edit the Solr server configuration and Indexer configuration. Each of these elements is a separate platform type, hence, they have their own editors. You have access to the following configurations:

- **Solr server configuration**: allows you to define the server-related aspects such as solr standalone server configuration, SolrCloud server configurations or search client configurations. Out of the box, we provide the default, local cloud and local standalone configurations.
- **Indexer configuration**: allows you to set the solr indexer configuration. Out of the box, we provide default, direct and two-phase configurations.
- **Search configuration**: allows you to configure the general search parameters such as result page size or default sort order. A default configuration is available out of the box.

ℹ Note

If the properties for configurations are set in the Backoffice Administration Cockpit, then the value comes from the property, otherwise it comes from the database.

Procedure

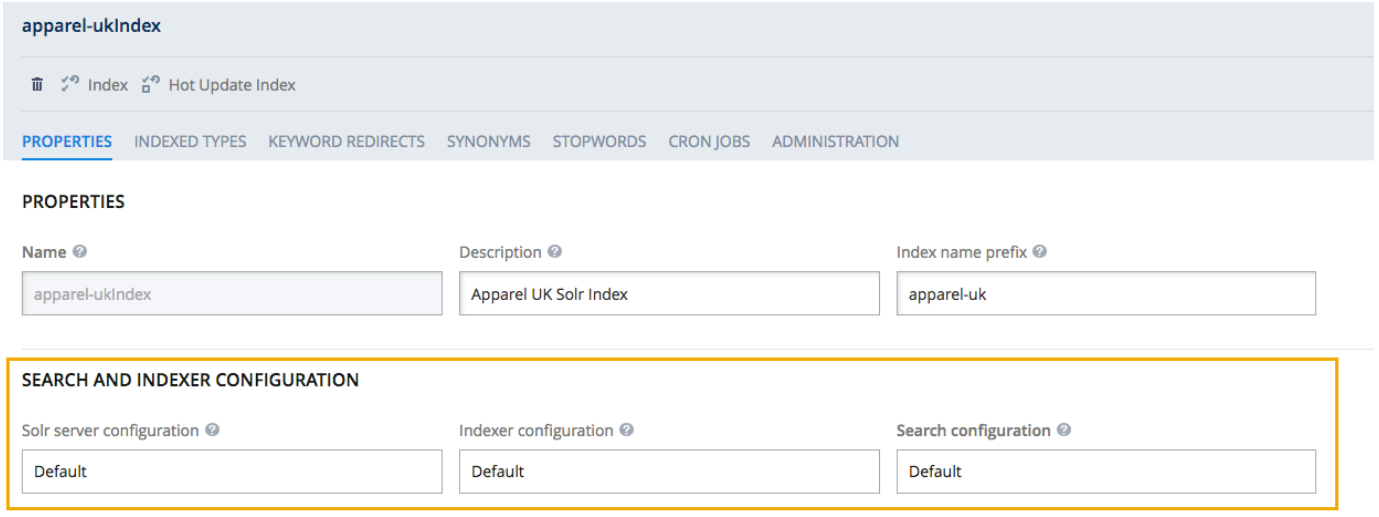
1. Navigate to **System** > **Search and Navigation** > **Solr Facet Search Configuration** > **Facet Search Configurations** (1).

   A list of available configurations (if any) appears.

2. Select an existing configuration. If you want to create a new one, follow the steps in: Add a Search Configuration.

   An editor opens in which you can define additional settings.

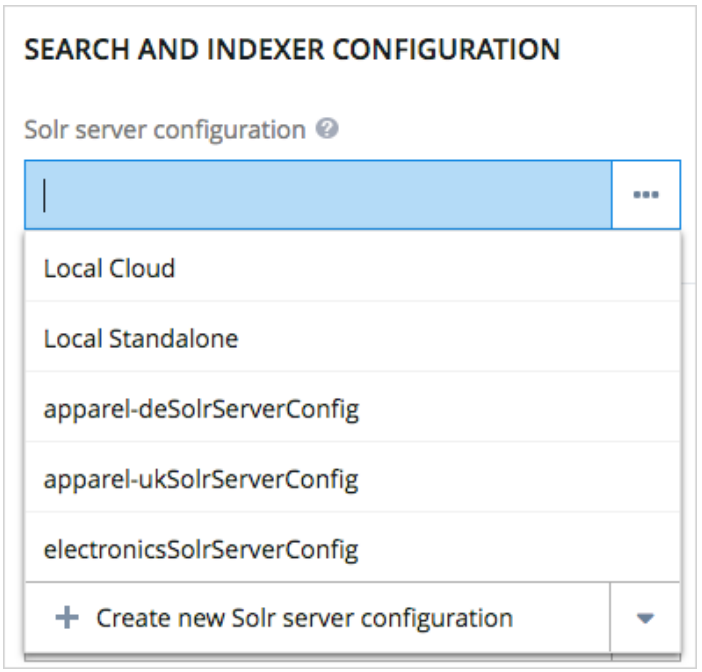
3. Under **Properties** you will find the **Search and Indexer Configuration** section.



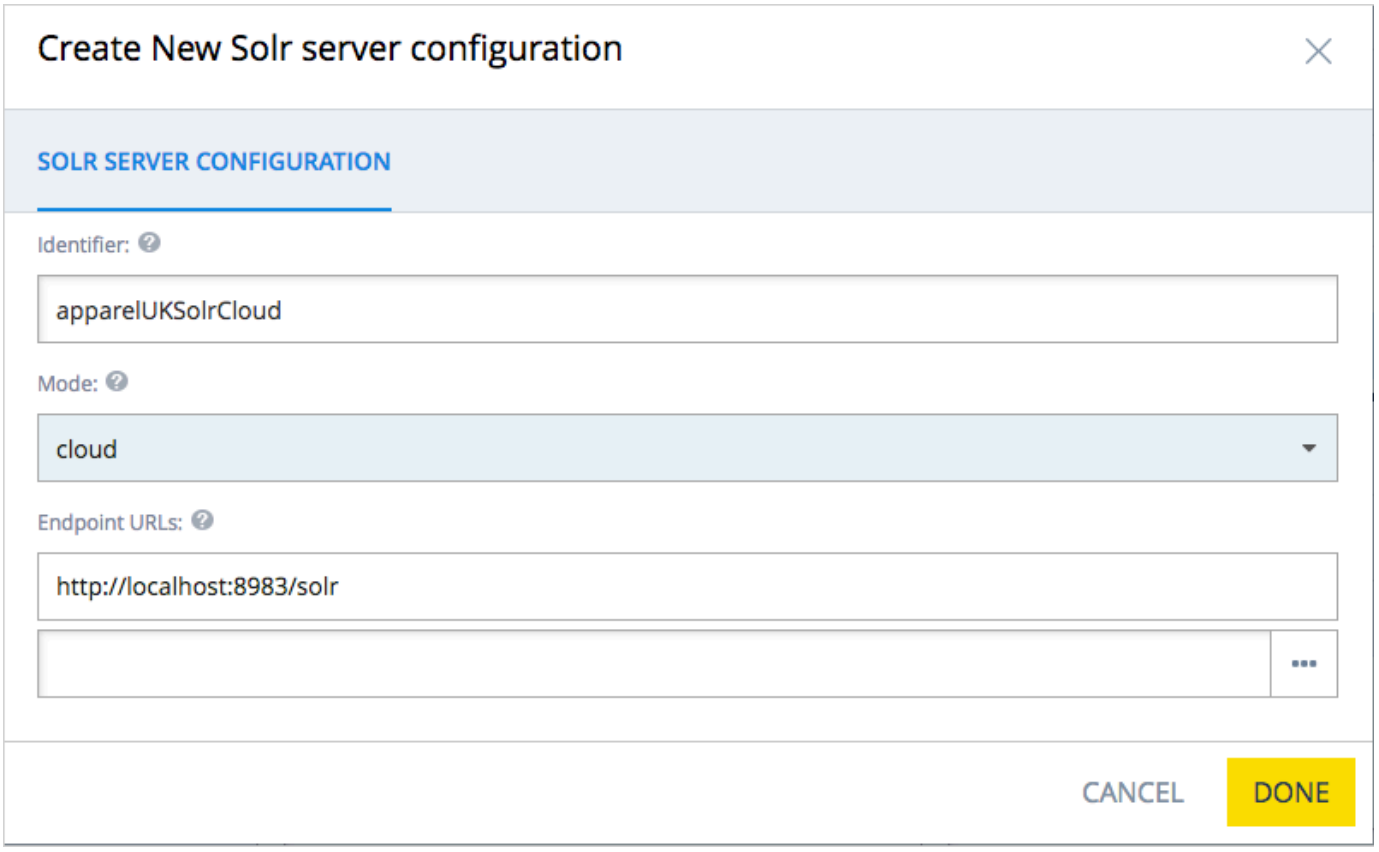
Configure the Solr Server

Procedure

1. Use the drop-down menu to select the **Create new Solr server configuration** option.



2. The Solr server configuration window opens.



3. Fill in the fields to define the basic settings for your configuration.

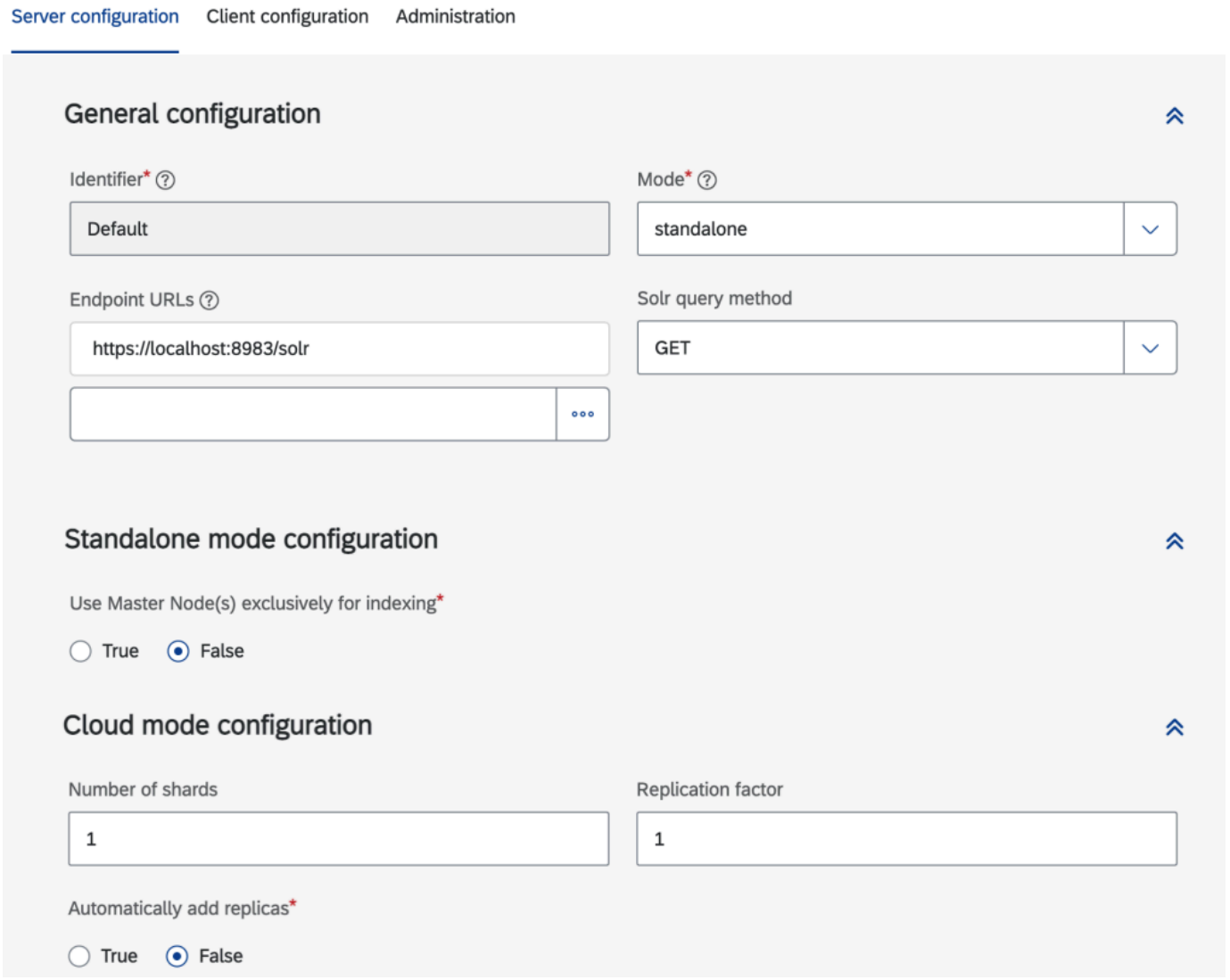
Field Name	Description
Identifier	A unique identifier for your Solr server configuration.
Mode	You can choose a mode of operation from the following values: ○ standalone ○ embedded ○ cloud

Field Name	Description
	○ xml-export
Endpoint URLs	URLs of Solr instances in this configuration. To ensure secure communication to and from Solr, use the **https** protocol. For details on secure communication, see Secure Communication Using SSL.

4. Click DoneSaveSocketTimeoutException. to finish.

   You should be able to find your configuration on the list. Select it to make it active. Remember to click to save your changes.

5. To add information about configurations, click the configuration to open the editor.



   Use the server tab to define additional solr server settings.

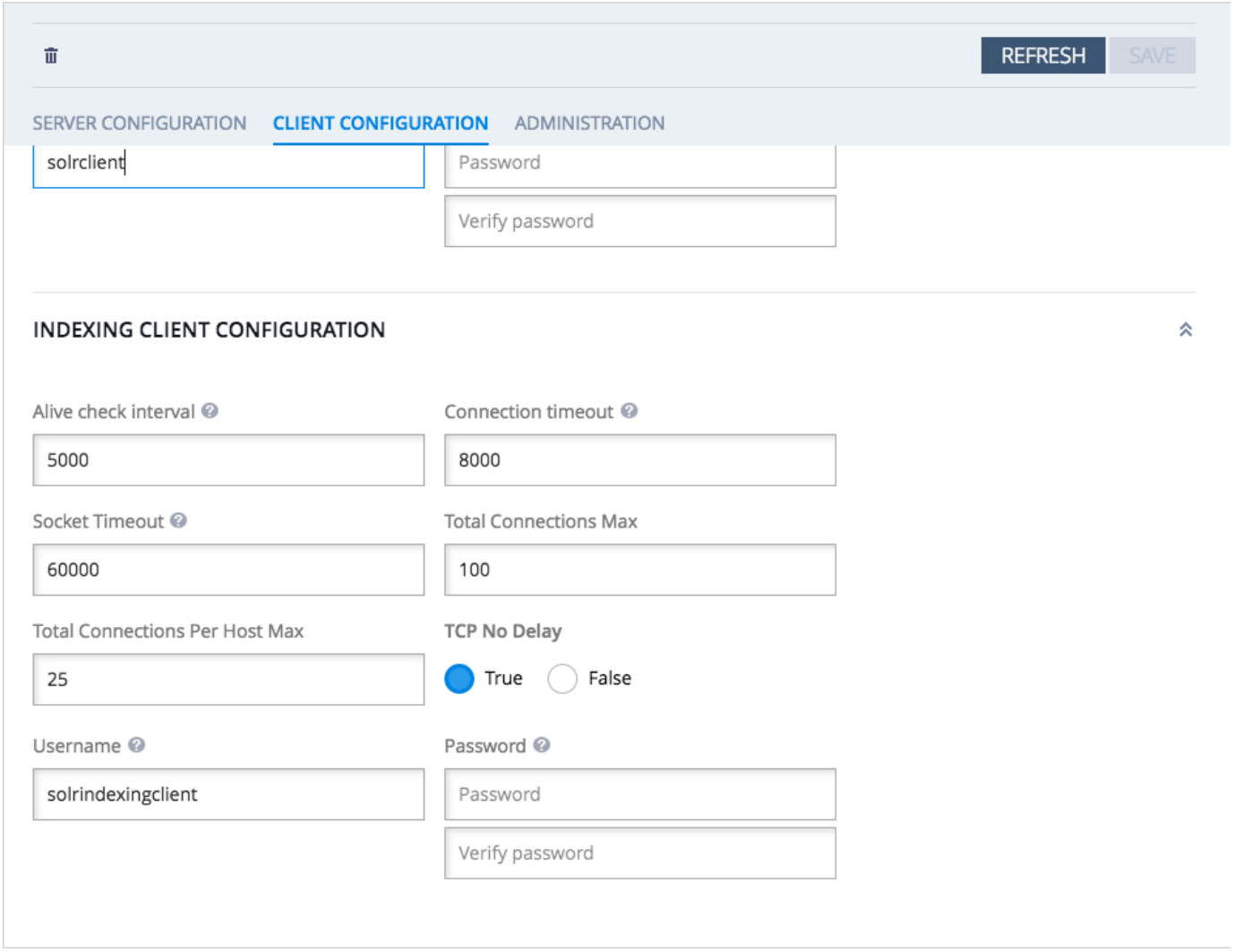
6. For Solr query method, *Get* and *Post* provide the same functionality with almost the same performance. However, *Post* supports a bigger header size.
7. For the Standalone Mode configuration, state if the leading node(s) should be used exclusively for indexing. If you are using standalone Solr with master-client mode, it can be set to True.
8. For the Cloud Mode Configuration you can set the following values:
   - Number of shards: When a collection is too large for one node, it can be broken up and stored in sections by creating multiple shards. A shard is a logical partition of the collection, containing a subset of documents from the collection, such that every document in a collection is contained in exactly one shard. You can keep this value to 1 unless there is a huge volume of data to be processed.

     ℹ Note

     This configuration is valid for all the indexed types, however, it can be overridden and defined for an individual indexed type. For details refer to: Additional Settings for Indexed Types.

- Replication factor: The replication factor defines the number of replicas you will have for each index. It is recommended to keep this value greater than 1 to avoid a single point of failure.
- Automatically add replicas: Use this option to allow Solr to create replicas automatically when the load is high. You can set it to False because automatically created replicas might hamper the full index job completion.

9. In the client configuration tab, you can add and edit the settings for the search client and indexer client.



- Socket Timeout: Timeout happens when the data flow is interrupted. If you see a **SocketTimeoutException** in the application log, you might need to update the value here. However, please be aware that the exception could be a consequence of other exceptions and increasing the timeout length might not help in such cases.

The available fields are the same for both clients and include the timeout settings and the maximum number of connections to be performed. Additionally, this tab allows you to configure the username and password for Solr client and indexing client to ensure authorized and authenticated connection. For details on authorization, see Authentication and Authorization Support for Solr.

Configure the Indexer

Procedure

1. Use the drop-down menu to select the **Create new Indexer configuration** option. The drop-down menu looks similar to the one for the Solr server configuration.
2. A pop-up window appears.
3. Fill in the fields to complete the configuration.

Field Name	Description
Identifier	A unique identifier for your indexer configuration.

Field Name	Description
Batch size	The size of the batch to be indexed. It is recommended to keep it below 1000. Refer 2446013.
Export path	Determines the directory where the indexer is supposed to write the indexing documents in XML_EXPORT mode.
Number of threads	Number of threads you use for indexation, typically two times the number of cores you have.
Indexer mode	You have the following options to choose from: • **DIRECT**: For full index operation, the current index is removed and replaced by a new index. For a certain period of time the index is not fully available for the query. • **TWO_PHASE**: The current index remains active until the new index has been built in the new Solr core. After that, new index replaces the old index and can be queried by users without any shutdown periods. External front-end applications do not need to know which Solr indexer core is active at the current time.
Commit mode	You have the following options to choose from: • **NEVER**: a commit operation will not be executed explicitly • **AFTER_INDEX**: hard commit after indexing; this is the default. Recommended when the batch size is small. • **AFTER_BATCH**: hard commit after each batch • **MIXED**: soft commit after each batch, hard commit after the indexing
Optimize mode	You have the following options to choose from: • **NEVER**: an optimize operation will not be executed explicitly, this is the default • **AFTER_INDEX**: optimize after the indexing It is recommended to set the value as NEVER. This option impacts indexing performance particularly when the data volume is high. If you opt for AFTER_INDEX, please adjust the SocketTimeOut setting on the Solr Server configuration page.
Distributed Indexing	In Distributed indexing mechanism, the items are split into batches and indexed in parallel. To know more on indexing, see Indexing Process.
Ignore errors	If the `ignoreErrors` flag is enabled, this means that some errors (related to data) are ignored during indexing.
Max retries	Maximum number of retries in total. This value can be set to 3 or 4 to workaround certain issues.
Max batch retries	Maximum number of retries per batch.

4. Click **Done** to finish.

   You should be able to find your configuration on the list. Select it to make it active. Remember to click **Save** to save your changes.

Results

You have successfully configured the settings for the Solr server and the indexer. You can edit them at any time by going to the Solr configuration tab and selecting the configuration you want to change.

Configure Search

Procedure

1. Use the drop-down menu to select the **Create new Search configuration** option. The drop-down menu looks similar to the ones presented above.
2. Fill in the fields.

Field Name	Description
Result page size	Defines the number of results per page.
Default sort order	Default sort order.
Description	A description for your search configuration

3. Click Done to finish.

Configure Fallback Language for Solr Indexer

Instructions on assigning the fallback language to the Solr facet search configuration using the **Backoffice Administration Cockpit.**

Context

The language fallback mechanism is responsible for providing a content for a localized attributes if no value is provided for the current localization settings. Products can be searched for based on their identifier defined specifically for each language. However, in case the identifier has no value for a particular language you can configure a fallback language that will be used for finding the indexed item.

Procedure

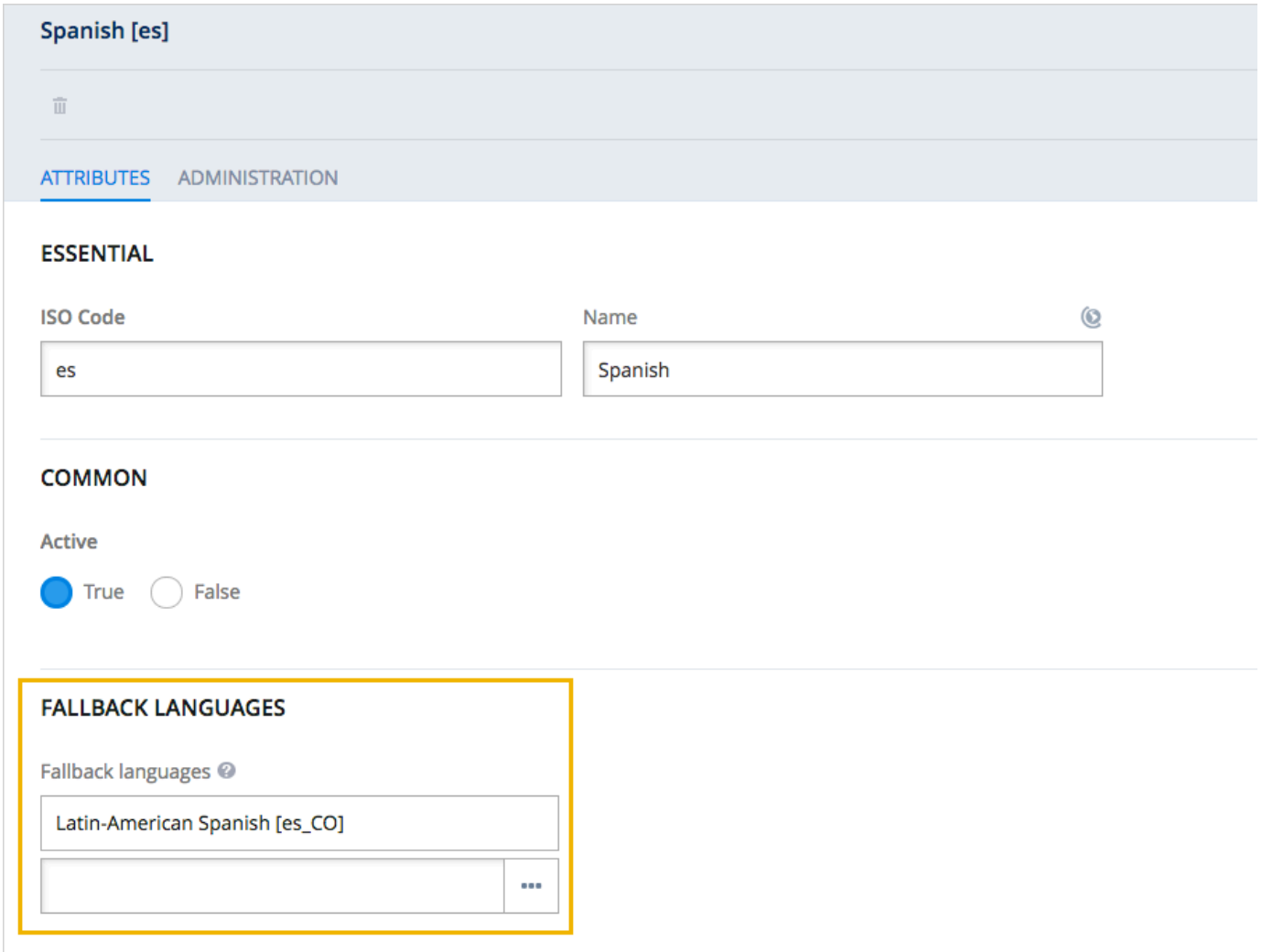
1. Navigate to Internationalization > Languages node in the explorer tree.

   A list of already configured languages (if any) is displayed.

2. Select the language you want to define the fallback language for.

   The editor opens.

3. In the editor go to Fallback languages section. Use the pop-up window or a search reference window to add a fallback language. You can also create a new one by clicking Create new language.



An new language appears on the list. This language can be than used by the Solr facet search configuration as a fallback language in case a certain indexed term does not exist for a product in the main language.

ℹ Note

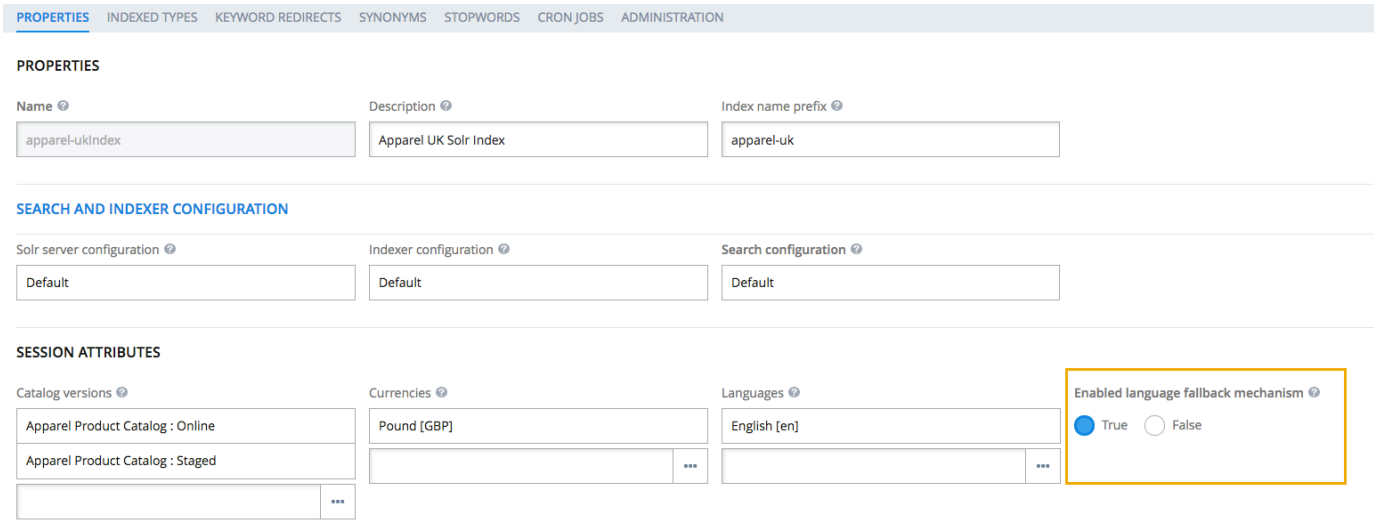
The fallback mechanism has to be enabled separately for each Solr facet search configuration.

4. Navigate to System > Search and Navigation > Solr Facet Search Configuration > Facet Search Configurations

   A list of available configurations appears.

5. Click the selected configuration to open the editor.

6. In the editor go to Properties tab. At the bottom of the page set Enabled language fallback mechanism to **True**.



   It enables the fallback language mechanisms for the current Solr facet search configuration. When done, click Save to keep the changes.

7. Once you have set up the fallback languages and enabled the fallback language mechanism for a particular facet search configuration you need to update your index in order to apply these modifications. For details on how to update a solr configuration see: Update Solr Index.

Configure Keyword Redirects

You can use the **keyword redirects** feature to define the words that will redirect your users to the certain search results like URL targets or items.

Context

Follow the steps listed below to create a new keyword redirect.

Procedure

1. Navigate to System > Search and Navigation > Solr Facet Search Configuration > Facet Search Configurations.

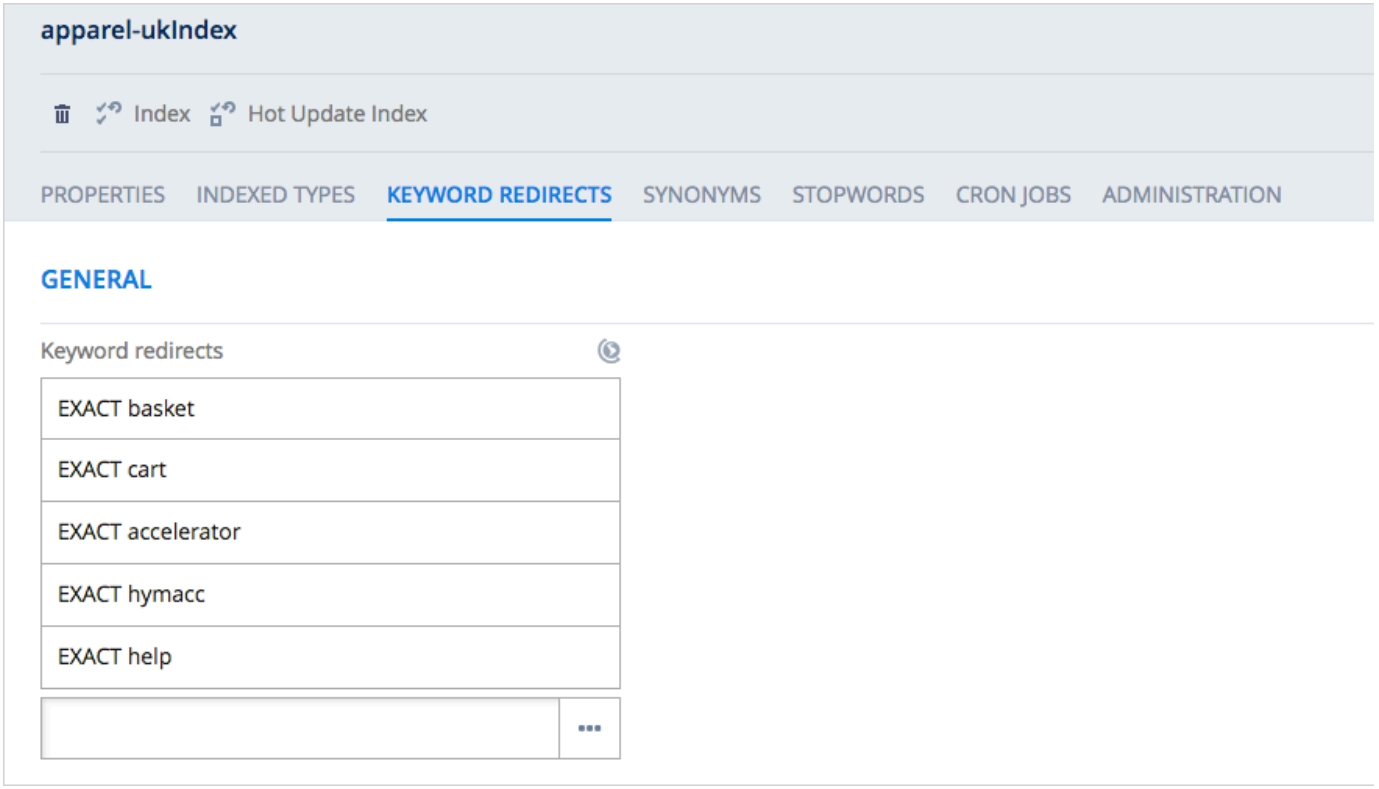
   You can see a list of available configurations (if any) on the right.

2. Click the configuration you want to add the keyword redirect to.

   An editor opens.

3. Go to Keyword redirects tab.

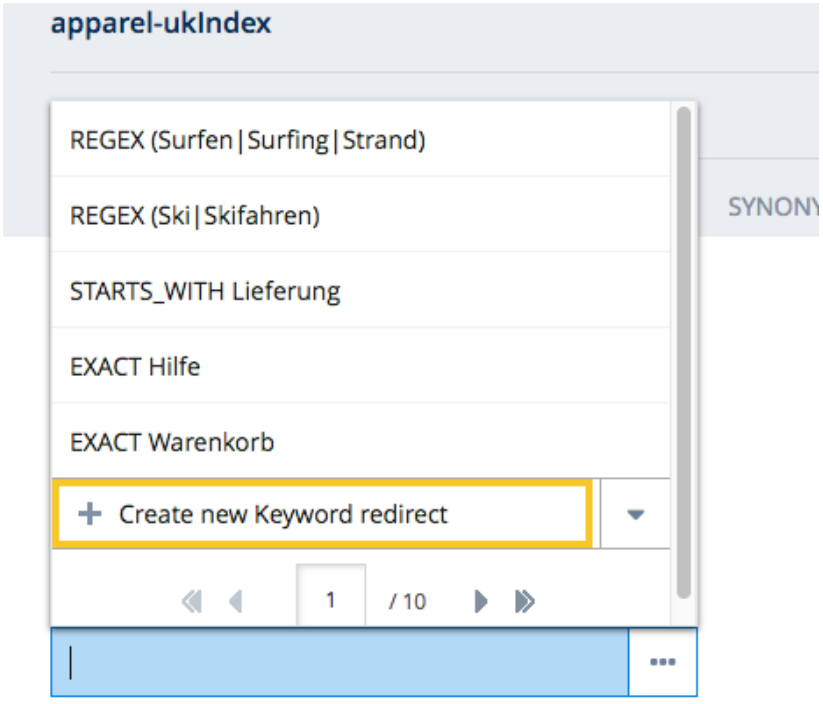
   A list of keyword redirects is displayed (if any of them have already been added).



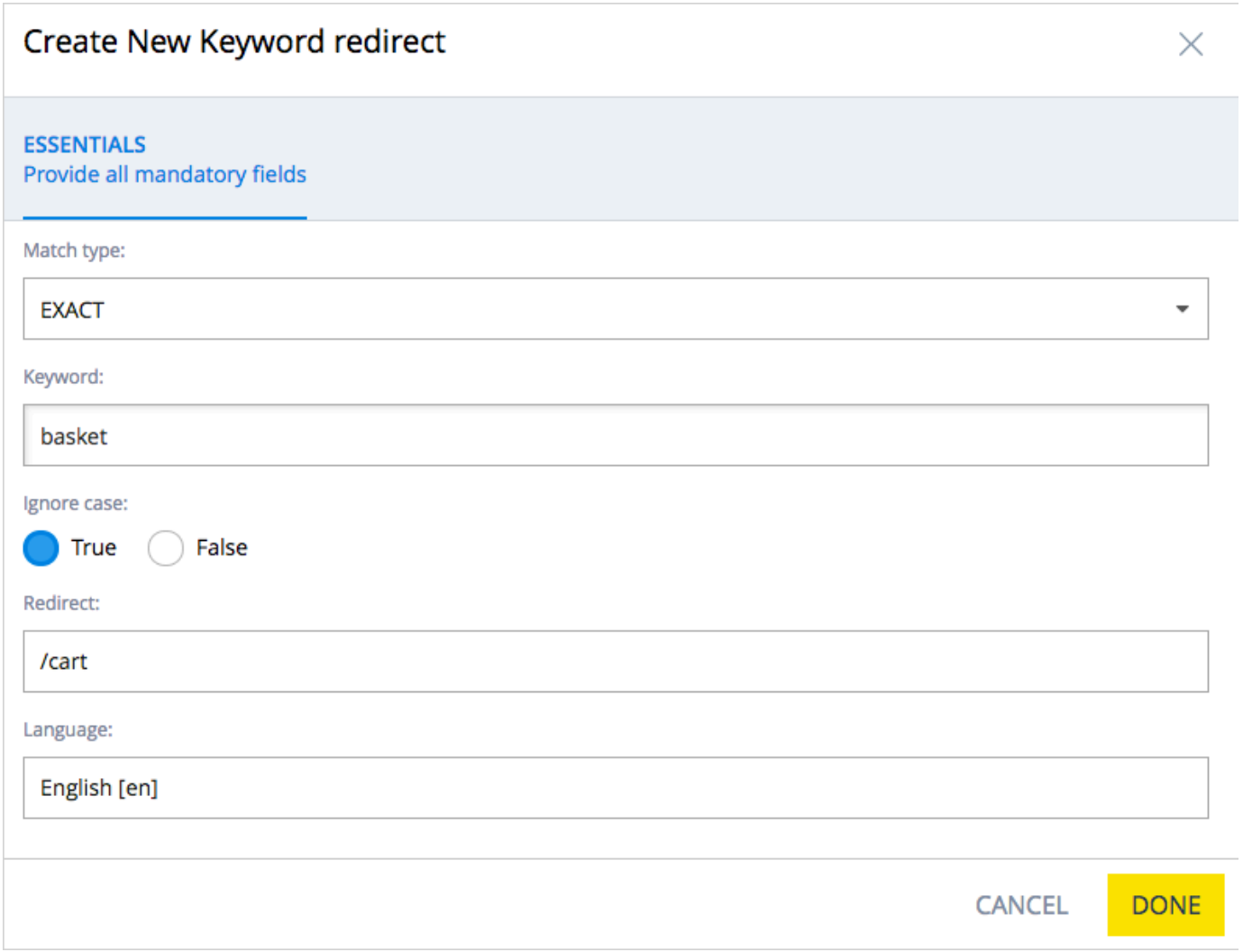
4. Use the more button to search for a new word.

   A search window opens. Use it to select keyword(s) that you want to add. You can select more than one at a time.

5. If you want to create a new keyword redirect use the drop-down menu and click the Create new Keyword redirect button.



6. Fill in the fields in the new window.



Field Name	Description
Match type	Sets the match option determining how the keyword redirect will be handled. Available options: EXACT, STARTS_WITH, ENDS_WITH, CONTAINS, REGEX.
Keyword	A keyword you are creating the redirect for.
Ignore case	Defines whether the keyword is case-sensitive.
Redirect	An item your keyword redirects to. It may be for example a product, a webpage or a directory.
Language	The language of your keyword.

Click Done to finish. Your new keyword is now available for use.

7. If you want to delete the keyword from the list, simply hover over it until the x button appears. Click the button.

   The keyword is removed from the list.

8. If you want to delete the keyword permanently, enter the editor mode by clicking the keyword entry.

   The editor appears.

9. Use the bin icon in the left-top corner of the editor to remove the keyword redirect.

   The keywords has been deleted.

Configure Synonyms

Synonyms in the Solr facet search let you define an additional word you can use as a search parameter.

Context

Using synonyms doesn't mean that you will get the same results searching for certain items. If you have a synonym, let's say A to B, you will most likely get different results when you search for A from when you search for B. This is due to the fact that synonyms cannot be used for some types of queries (for example wildcard and fuzzy queries).

Solr provides you with a default list of synonyms, and you can also define your own synonyms in the Backoffice Administration Cockpit.

ℹ Note

Synonyms are by default also used in the autocomplete and spell checking functionality.

Procedure

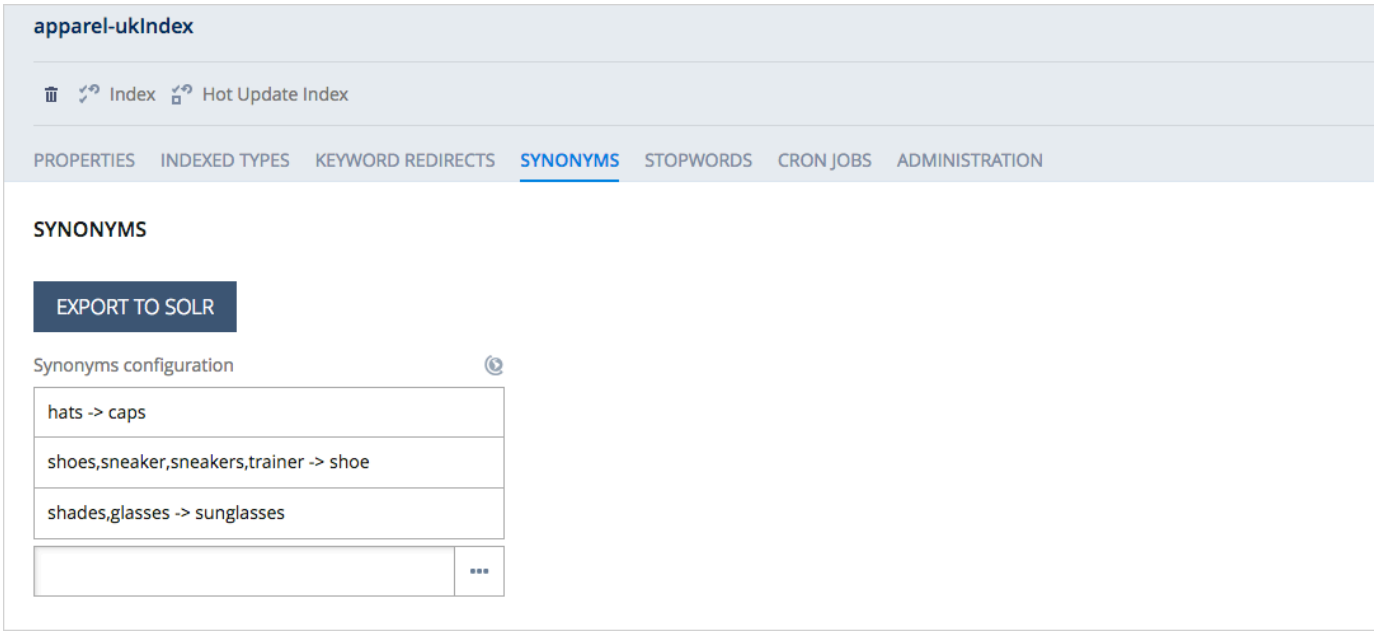
1. Navigate to System > Search and Navigation > Solr Facet Search Configuration > Facet Search Configurations in the Explorer Tree.

   A list of available configurations (if any) appears on the right.

2. Select a configuration and click it.

   An editor appears.

3. Select the Synonyms tab.



4. At this point you can select the already created synonyms or add new ones.
5. In order to select an already existing synonym, open the Reference Search window. You can also start typing the name of a synonym to find and add it.
6. Select the synonyms you want to add to your configuration from the list and click Select.

   Selected synonyms appear on the list.

7. To add a new synonym click the Create new Synonyms configuration button.