

|  |
| --- |
| SAP Hybris - Developer Toolset  Documentation |
|  |

Contents

[1 IMport hybris extensions 3](#_Toc470876774)

[1.1 Import hybris Extensions wizard 3](#_Toc470876775)

[1.1.1 Import based on “localextensions.xml analyze” method 3](#_Toc470876776)

[1.1.2 Import based on scanning method 4](#_Toc470876777)

[1.2 Refresh required extensions 4](#_Toc470876778)

[2 impex 4](#_Toc470876779)

[2.1 Content Assist 5](#_Toc470876780)

[2.1.1 Highlighting 6](#_Toc470876781)

[2.1.2 Formatting 7](#_Toc470876782)

[2.1.3 Outline view integration 7](#_Toc470876783)

[2.1.4 Folding 7](#_Toc470876784)

[2.1.5 Hyperlinks 7](#_Toc470876785)

[2.2 ImpEx Validation/Import 8](#_Toc470876786)

[2.2.1 Configure remote server 9](#_Toc470876787)

[2.2.2 Run/Validate impex 9](#_Toc470876788)

[3 Flexible search 10](#_Toc470876789)

[3.3 Flexible search builder 10](#_Toc470876790)

[3.4 Flexible search Query editor 12](#_Toc470876791)

[3.4.1 Suggestions/Auto-completions 12](#_Toc470876792)

[3.4.2 Highlighting 13](#_Toc470876793)

[3.4.3 Query Executing 13](#_Toc470876794)

[4 Extensions 14](#_Toc470876795)

[4.1 extension generation 14](#_Toc470876796)

[4.2 module generation 15](#_Toc470876797)

[5 Hybris class creation 15](#_Toc470876798)

[5.1 Populator generation 16](#_Toc470876799)

[5.2 INTERCEPTORS generation 17](#_Toc470876800)

# IMport hybris extensions

SAP Hybris - Developer Toolset allows you to easily import SAP Hybris projects with different extension sets and keep them up-to-date in your project workspace.

## Import hybris Extensions wizard

Import functionality can work based on the "*config/localextensions.xml*" file or via importing from directory methods.

To start importing, go to *File->Import->Hybris->Import Hybris Extensions wizard*.

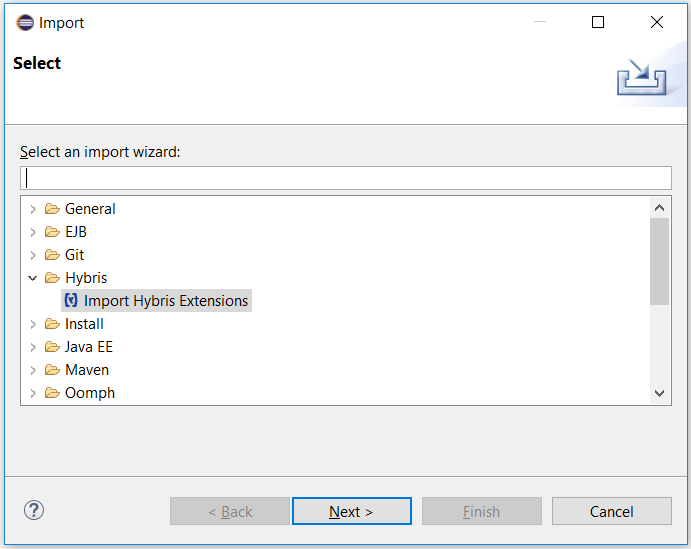


Figure 1: Choosing Import Hybris Extension wizard

### Import based on “localextensions.xml analyze” method

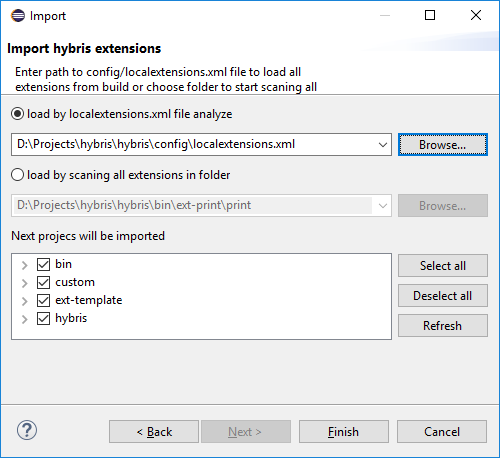
This import method is based on the analysis of the “***config/localextensions.xml”*** file and extensions dependencies in the “***extensioninfo.xml*”** file.

Figure : Import Hybris Extension wizard

To start importing, choose "*load by localextensions.xml file analyze*" option and press "*Browse*" button to find your file.

System "Open file dialog" will be opened. Choose your “***localextensions.xml***” file.

Once loaded, the plugin will start to analyze extensions found in this file and their dependencies.

This process ensures that all extensions and their dependencies will be automatically loaded. It guarantees that your workspace will have that set of extensions loaded by the SAP Hybris build framework.

When all extensions are loaded, you can manually select/deselect all extensions you you need/don't need for your workspace.

Notice that the selection of any extension will also automatically select all of its dependencies.

When you’re done, press "*Finish”.* All selected extensions will be imported to your workspace.

### Import based on scanning method

To start importing by this method, choose "load by canning all extensions in folder" option and press the "Browse" button. System "Open folder dialog" will be opened.

When you select the folder you wish to scan, the wizard will find all extensions in that folder and let you manually select, which specifies all extensions you wish to load.

Notice that the selection of any extension will also automatically select all of its dependencies.

When you’re done, press “*Finish*”. All selected extensions will be imported to your workspace.

## Refresh required extensions

If your “*localextension.xml*” file is changed, an extension has a new dependency, and/or you feel that your workspace is missing an extension, you can use the “*Refresh required extensions*” command. This command will find the “*localextension.xml*” file in your workspace and start analyzing and importing missing extension processes.

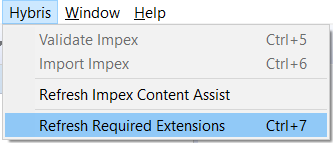


Figure 3: Refresh Required Extensions command

**Warning**! Be aware that during import of your Hybris project, there is the possibility of rewriting classpaths based on required extension tag in extensioninfo.xml file. You can disable/enable this by going to *Windows->Preferences->Hybris* and checking/unchecking “*Enable rewriting classpath*”.

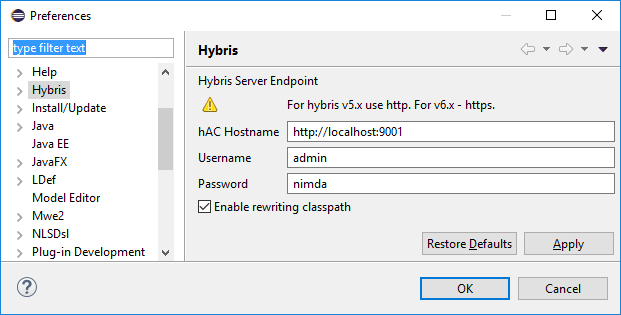


Figure 4: Hybris preferences page

# impex

All files matching the “*\*. impex”* pattern will now be automatically opened in a new Impex Editor.

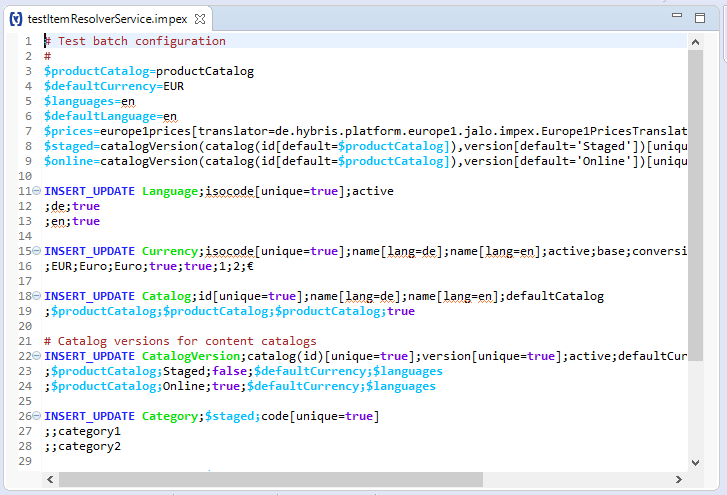


Figure 5: Impex Editor

## Content Assist

The Impex Editor featured smart content assist (Ctrl + space), which shows a list of proposed words in the Impex Editor based on a system of common workspace project types. Proposals listed in the window include:

* **Header commands** displays all the supported commands for Impex, e.g. *INSERT\_UPDATE*, *INSERT*, *UPDATE*, *REMOVE*;

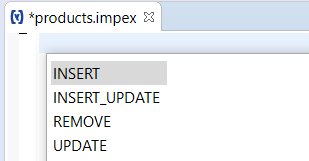


Figure 6: Header commands

* **Item type (model)** displays all the matched item types for the impex. If you have some characters typed in before pressing Ctrl + Space, it will show only the item types containing these characters in the name:

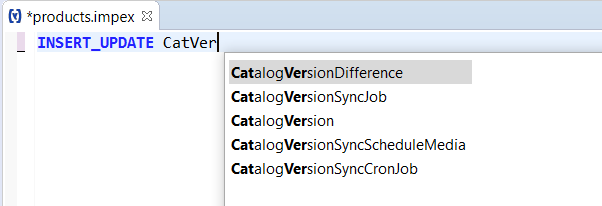


Figure 7: Item types

* **Type attributes** displays all defined attributes and relations for a specific item type:

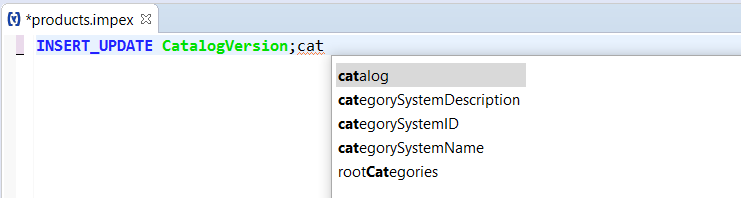


Figure 8: Type Attributes

* **Modifiers** displays all supported modifiers for impex (available after '[' character), e.g. *unique*, *batch*, etc.:

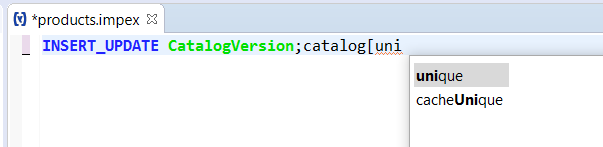


Figure 9: Modifiers

* **Complex type attributes** displays search attributes for inserting a complex type attribute, i.e. when you insert *"catalogVersion(*” and press Ctrl + Space, you will be able to choose one of the *catalogVersion'*s attributes;

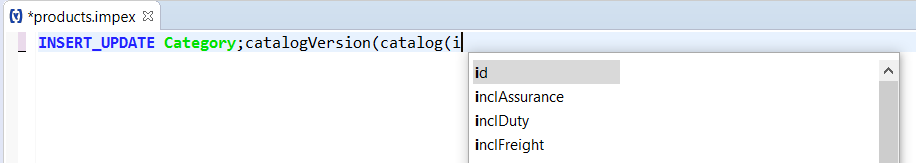


Figure 10: Complex type attributes

You can also disable/enable the content assist via *Preferences*->*Impex*->*Advance*. By default, all content assist functionality will be enabled. If an option in the preference is deselected, content related to that option will not be shown when using Ctrl + Space.

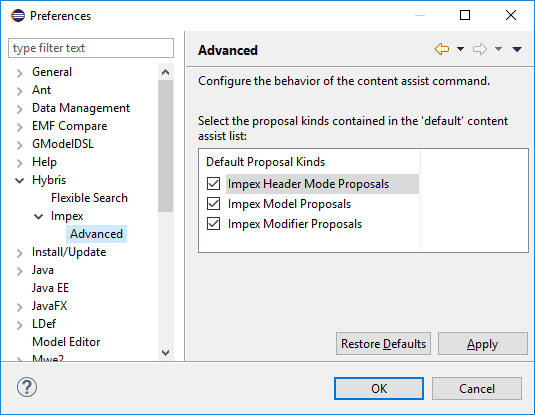


Figure : Impex assist preferences

### Highlighting

When using the Impex Editor, the header lines will be highlighted with appropriate colors (shown in the above pictures). You can also change the colors via *Preferences*->*Hybris*->*Impex*.

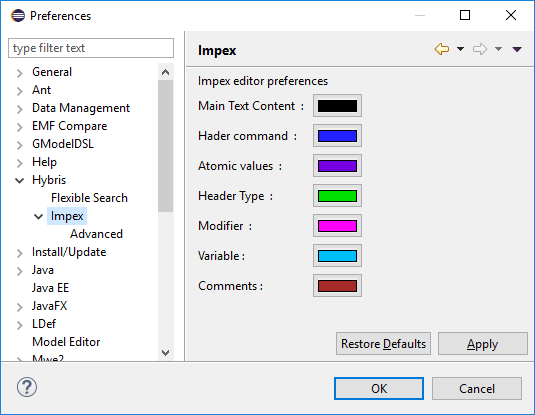


Figure 12: Impex highlighting preferences

### Formatting

To format impex text, press Ctrl + Shift + F.

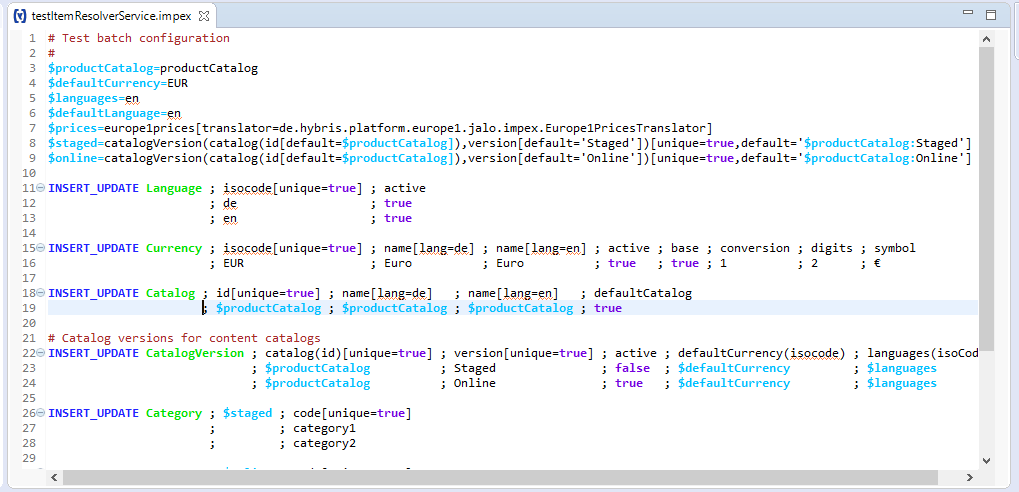


Figure 13: Impex after formatting

### Outline view integration

Use the Outline view for easy navigation and search within an impex file:

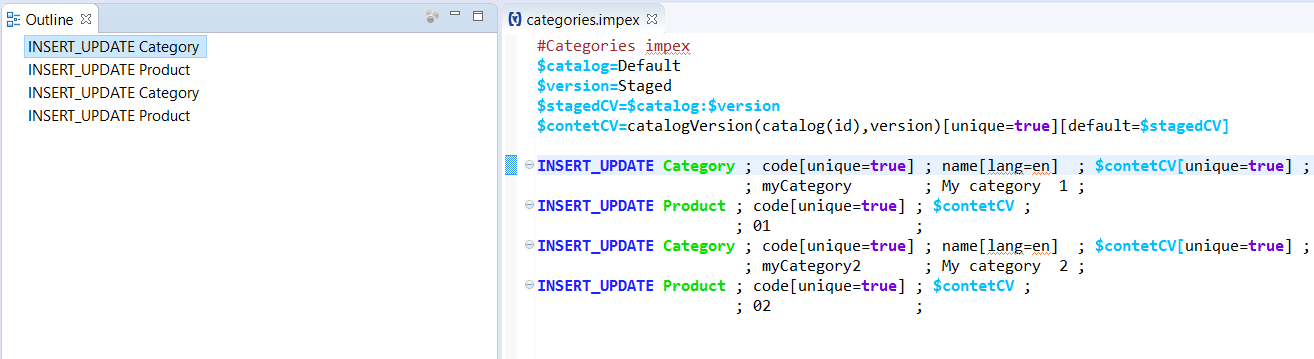


Figure 14: Outline view integration

### Folding

Folding functionality enables easy impex editing:

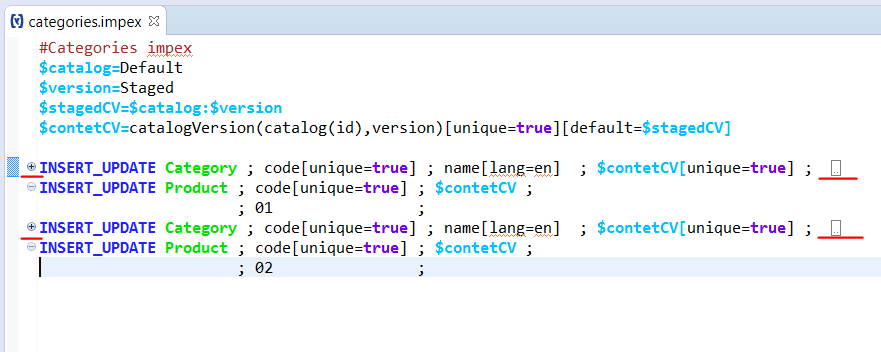


Figure 15: Folding

### Hyperlinks

While working on the impex file, it is possible to navigate from the impex file to Type/Attribute. Press Ctrl and browse with your mouse onto the impex file to make links appear:

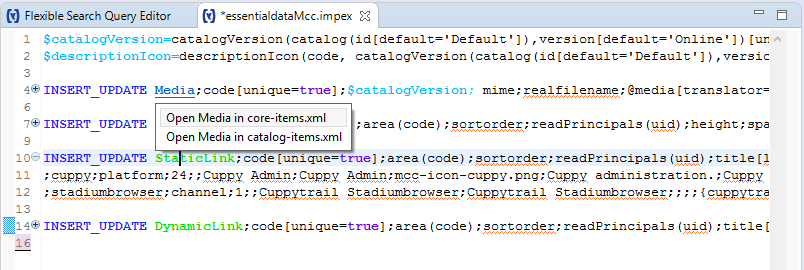


Figure 16: Hyperlinks

If you click on Model type, you will be able to choose the file you want to navigate from a list where this type is declared.

## ImpEx Validation/Import

EPAM’s SAP Hybris Plugin for Eclipse allows you to run or validate your impex on remote servers. This functionality is similar to the impex import console in hAC. In addition, the plugin also supports running impex executions for multiple files at the time, which is not supported in hAC.

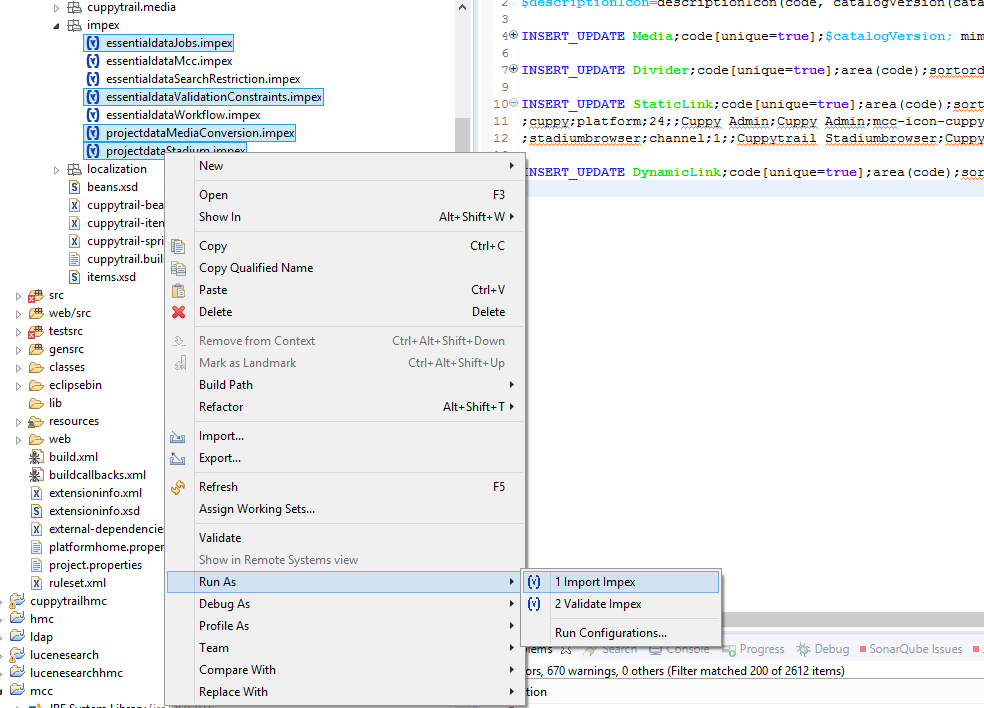


Figure 17: Importing multiple files at once

### Configure Remote Server

To configure paths to your running Hybris instance for the impex executions, go to *Windows->Preferences->Hybris*. Type the URL to hAC of your running Hybris instance on which you want to run/validate the impex.

By default, it will be a local instance: http://localhost:9001. Make sure your Hybris instance is running when you want to execute the impex commands.

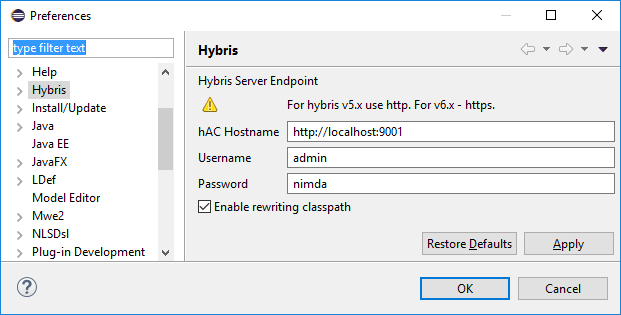


Figure 18: Hybris preference page

### Run/Validate impex

These executions can be applied to a single impex file or a list of impex files. File names must be suffixed by “.impex”. Executions will not be disabled when non-impex files are selected (i.e. not suffixed “.impex”).

#### Trigger impex executions

There are two ways to trigger the impex Validation/Import executions:

1. Go to the top menu *Hybris*->*Impex Import* (or *Validate Impex*).
2. *Right click the impex file*->*Run As*->*Impex Import/Validate Impex*.

#### Execution Results

* If the process is successful, it will show the filenames in the popup window that are successfully imported/validated;
* If the process fails, it will show the filenames in the popup window that have failed to import/validate; detailed error messages will be shown in the Eclipse Console (shown as below, e.g. when validating two impex files);

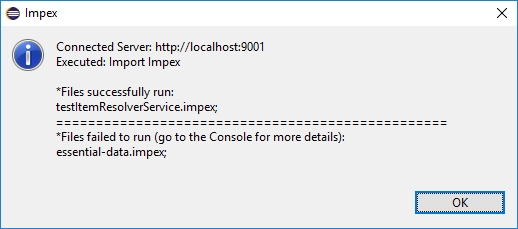


Figure 19: Impex run results

# 3 Flexible search

EPAM’s SAP Hybris Plugin for Eclipse has functionality which allows the simplification of work with Hybris Flexible Search. To correctly use this feature, you should enable the Hybris server and input the proper hAC URL and credentials to your running instance in the Hybris preference page (*Window->Preferences->Hybris*).

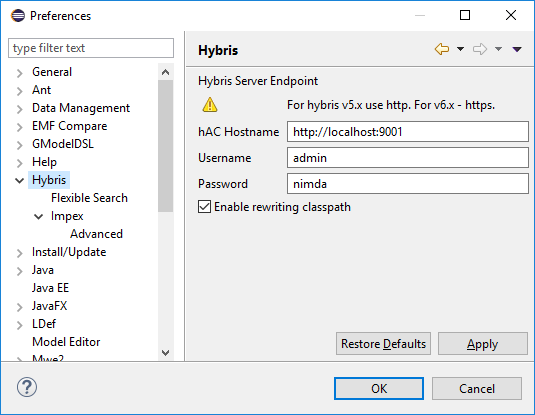


Figure 20: Hybris preference page

## Flexible search builder

You can quickly create and run simple queries without subqueries or join statements within Flexible Search Builder View. This view can be found via *Window->Show View->Others->Hybris->Flexible Search Builder*.

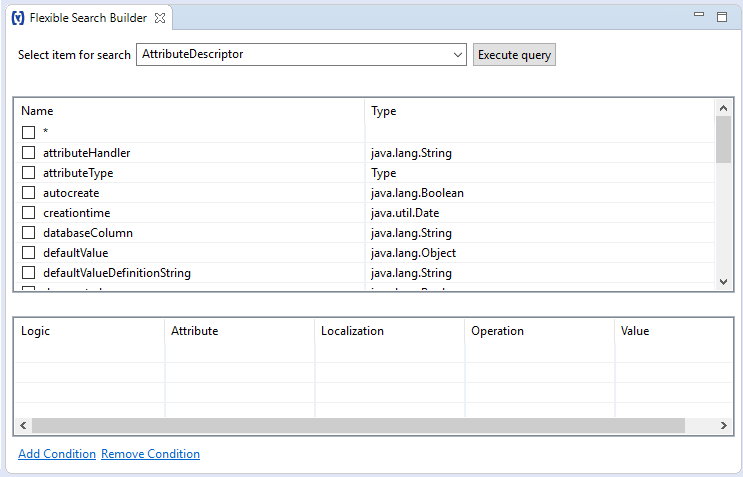


Figure 21: Common view of Flexible Search Builder

Flexible Search Builder consists of three main parts:

1. ItemType section: This allows you to select one of the types declared in workspace projects.

Figure : ItemType section

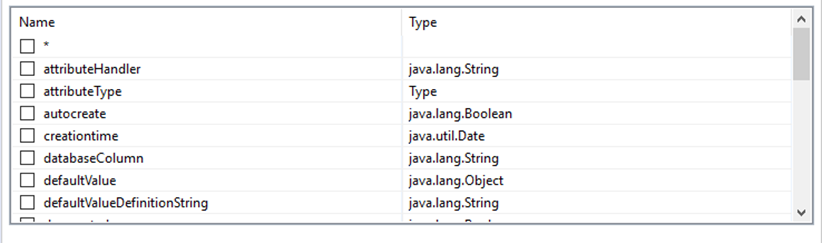
1. Attributes section: This allows you to choose which attributes must be included in the Flexible Search results table. Unless you include specific attributes, asterisk (all attributes) is chosen by default. Chosen attributes will be ignored if you choose the asterisk.

Figure : Attributes section

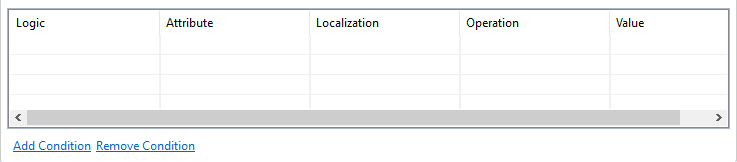
1. Conditions section: This allows you to create or remove conditions for filtering query.

Figure 24: Flexible Search Builder Conditions section

Every condition has five columns:

* 1. **Logic** – Used to filter records based on some logic with more than one condition. It has two operators: AND and OR
  2. **Attribute** – Used to select attribute for filtering.
  3. **Localization** – Used only for localized attributes to define locale.
  4. **Operation** – Operation for condition, which will be applied for the selected attribute. It includes a series of standard operations, e.g. =, !=, <=, LIKE, IS NULL, etc.
  5. **Value** – Used to specify value.

To execute query, press the ‘Execute Query’ button.

The results of the query will appear in Flexible Search Results View:

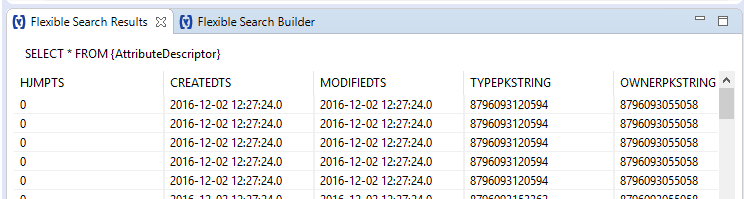


Figure 25: Flexible Search Results View

## Flexible search Query editor

For creating more complex queries, you can use the Flexible Search Query Editor View via *Window->Show View->Others->Hybris->Flexible Search Query Editor*.

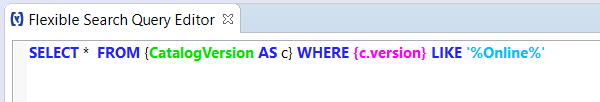


Figure 26: Flexible Search Query Editor

### Suggestions/Auto-completions

Flexible Search Query Editor has smart content assist *(*Ctrl + space*),* which shows the list of proposals based on the system type of workspace projects.

Content assist will automatically add brackets, aliases, and/or sub-queries constructions if they are necessary.

Content assist will analyze context of assistance and will display different proposals.

Types of proposals include:

* **Keywords** displays all the supported keywords for Flexible Search, e.g. *SELECT*, *FROM*, *IN*, *NULL*, etc. All types of operations for different attributes are also supported:

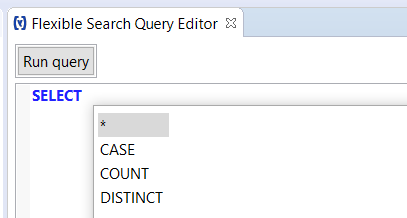


Figure 27: Keywords

* **Item type (model)** displays all the matched item types and relation tables. If you have typed in some characters before pressing Ctrl + Space, it will show only the types that start with the characters:

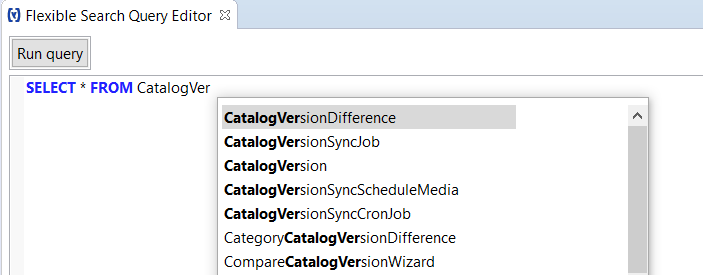


Figure 28: Item types

* **Item attributes** displays attributes of the previously chosen type:

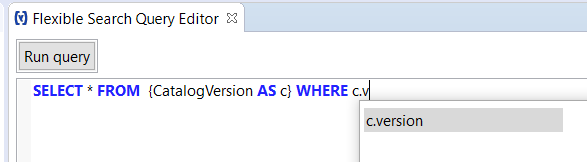


Figure 29: Attributes

### Highlighting

When using the Flexible Search Editor, the keywords, types, and attributes will be highlighted with appropriate colors (as shown in the above pictures). You can also change the colors via *Preferences* -> *Hybris* -> *Flexible* *Search*.

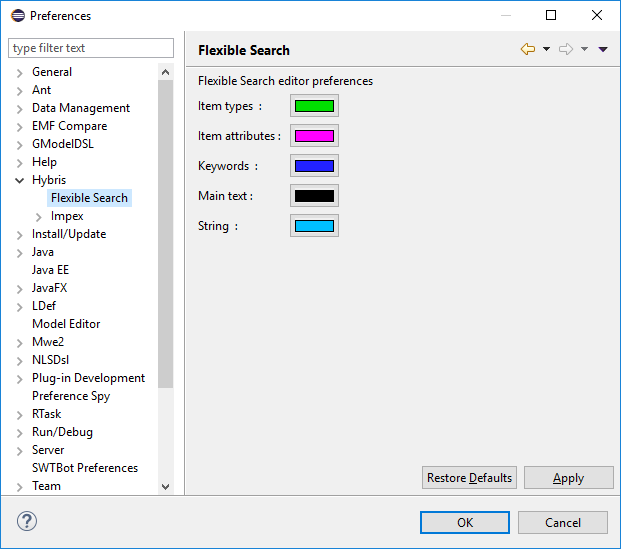


Figure 30: Flexible Search preferences

### Query Executing

Flexible Search Query Editor allows you to run your query on remote servers. Press the “*Execute query*” button to execute it on your remote Hybris instance.

To configure paths to your running Hybris instance for query executions, go to *Window-*>*Preferences* -> *Hybris* and input the proper URL to your hAC corresponding to the running instance.

Make sure your Hybris instance is running when you are executing the query.

Query Results will be opened in Flexible Search Results view:

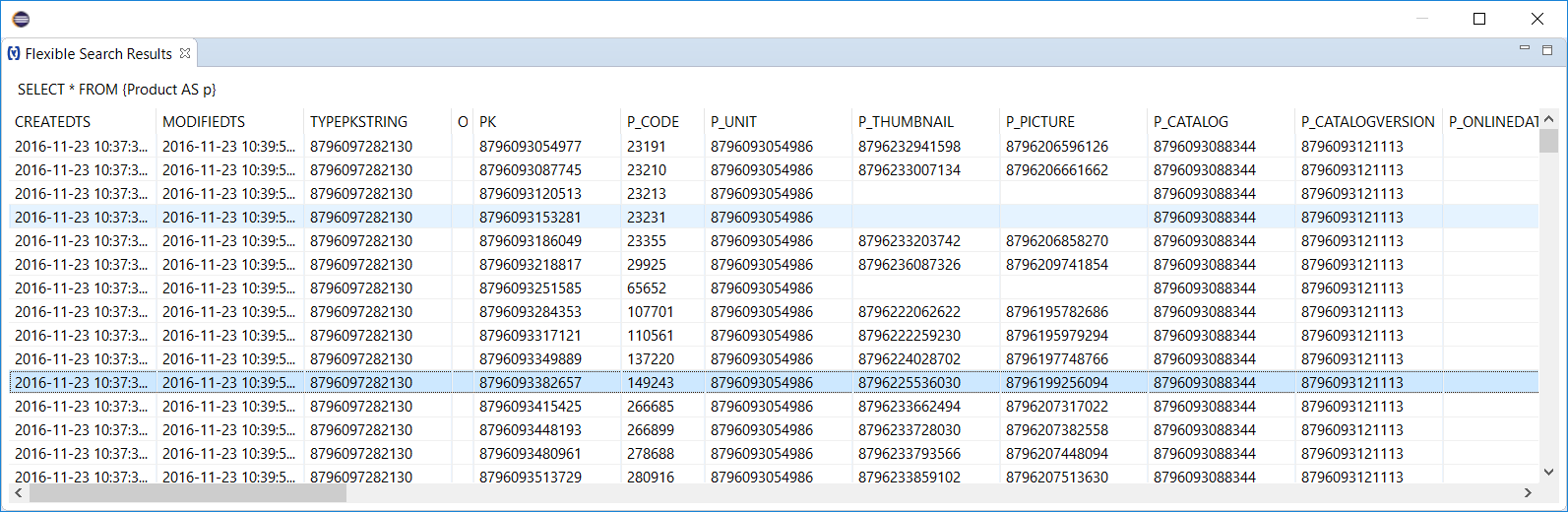


Figure 31: Results view

If any errors appear during execution, they will be displayed in a dialog box:

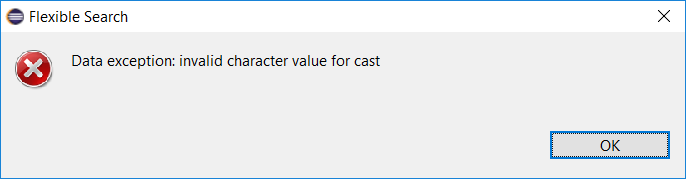


Figure 32: Exception display box

# Extensions

EPAM’s SAP Hybris Plugin for Eclipse is able to generate extensions and modules.

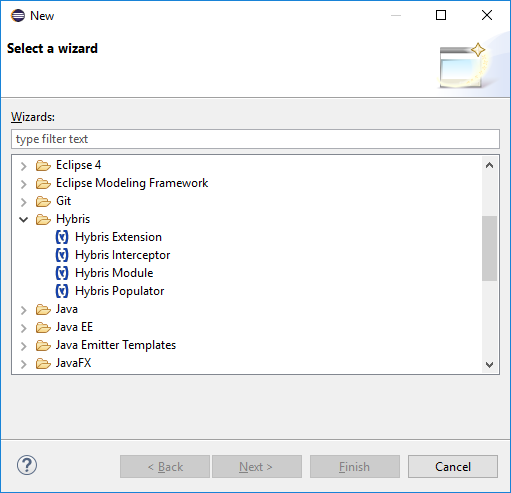


Figure 33: New wizard

## extension generation

In the Hybris new extension wizard, you are able to create a new extension, as well as define the name, package, template, required extensions, and location where the new extension will be located.

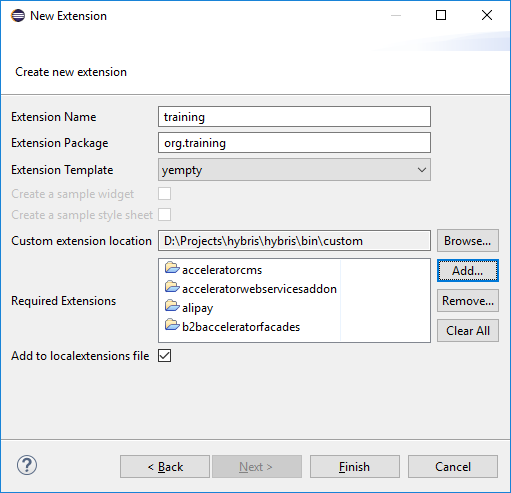


Figure 34: New extension wizard

## module generation

In the Hybris new module wizard, you are able to create a new Hybris module, which is a set of Hybris extensions, as well as define the name, package, template, and location where the new module will be located.

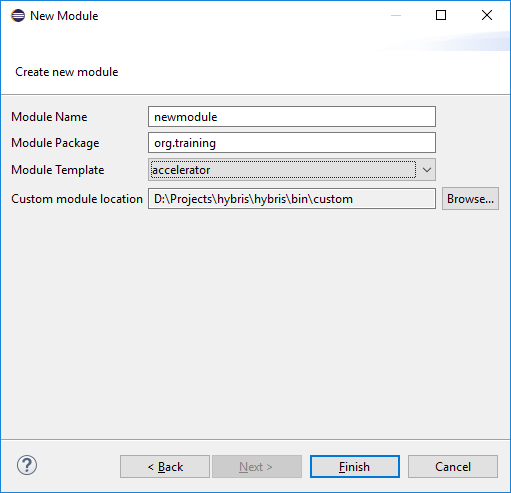


Figure 35: New hybris module wizard

# Hybris class creation

EPAM’S SAP Hybris Plugin for Eclipse is able to generate Hybris-specific classes.

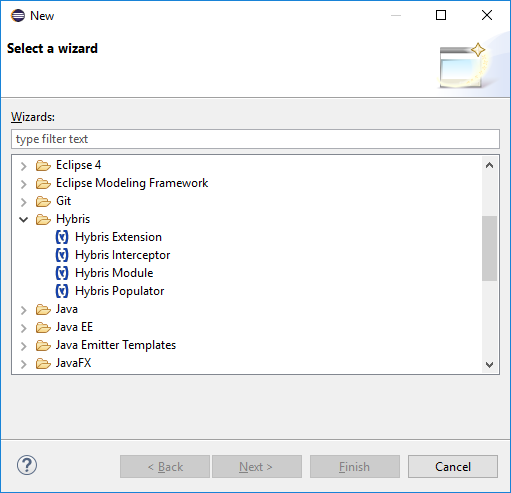


Figure 36: New wizard

## Populator generation

You can create a class, which will implement the Hybris Populator interface.

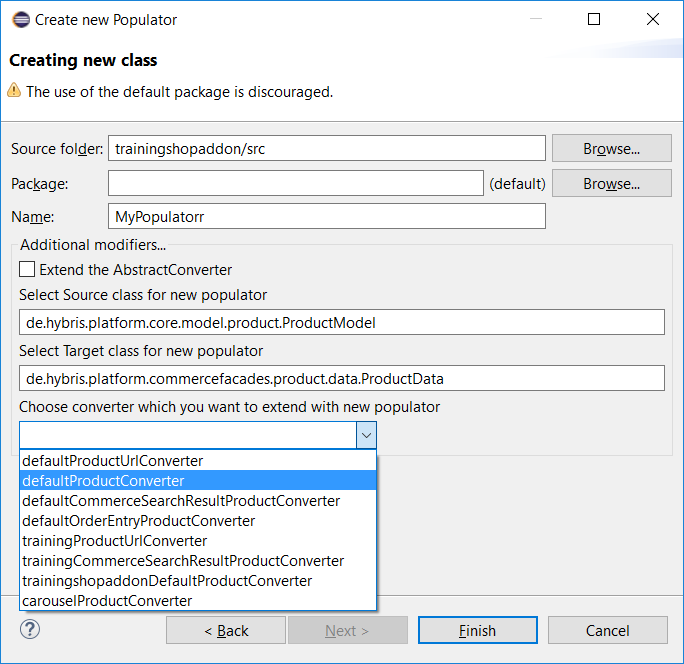
You can also choose the "*extend AbstractConverter*" check box. With this checkbox, your future class will extend the AbstractConverter class, rather than directly implement the Populator, but it will still be an instance of Populator.

Figure : New Populator wizard

In the new Populator wizard, you are able to define the source folder, package, and type name.

If you want to create a new class in existing Hybris projects, you will get an error message. You can only create new files in custom projects.

To select the source folder, just press the "*Browse...*" button. You will then see the file tree that contains all source folders from custom extensions in your workspace.

Additionally, you can choose **Source** and **Target** classes for your Populator.

You are able to select the converter which you want to extend from the drop-down. Wizard scan all *\*-spring.xml* files in the workspace and find all declared converters with selected **Target** class.

The generated file will have the following structure:

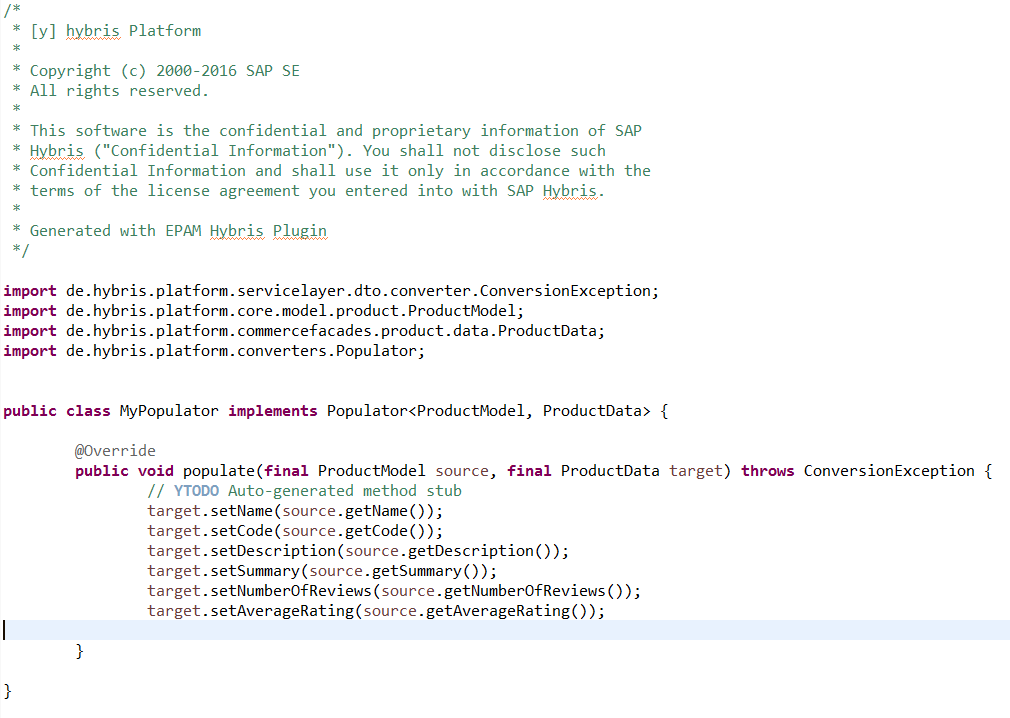


Figure 38: New Populator class

Additionally, the *\*-spring.xml* file from your selected project will be generated with a *Populator* declaration:

C:\Users\Dmytro_Pyvovarenko\Desktop\image2016-11-16 14-51-49.png

Figure 39: New Populator bean

If you selected a converter from the drop-down, a new *Converter* declaration will be generated:

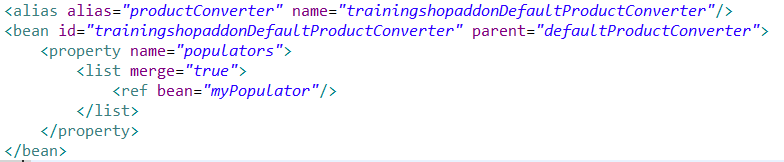


Figure 40: New Populator injection

If the selected converter is declared in the same file where the new *Populator* is, the converter will be extended by adding the Populator’s refs to the existing bean instead of creating a new converter bean.

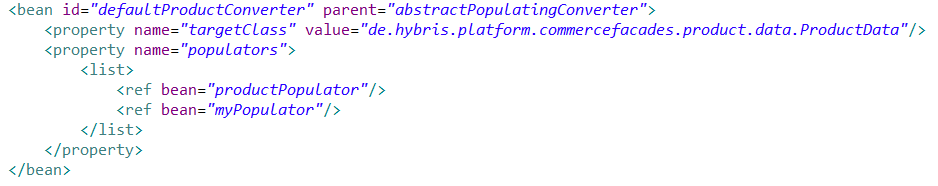


Figure 41: New Populator injection in same file

## INTERCEPTORS generation

In the new interceptor wizard, you are able to define the source folder, package, type name, and one of the five predefined Interceptor interfaces to implement, depending on your needs.

You will also be able to choose interceptor order and replace interceptor properties, if necessary.

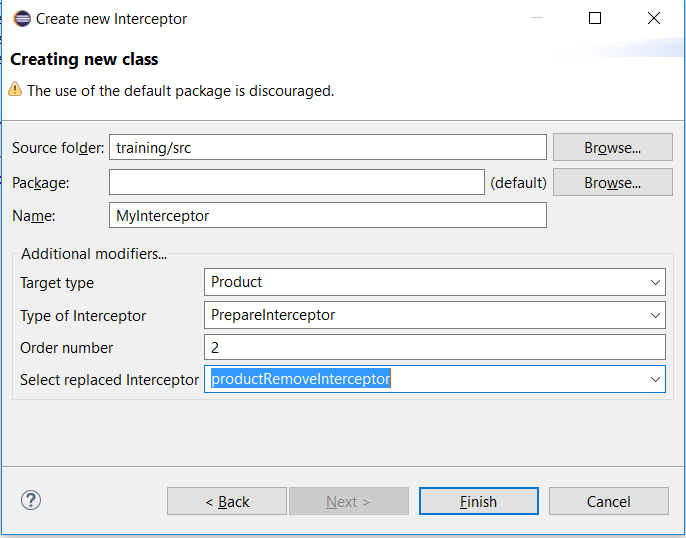


Figure 42: New interceptor wizard

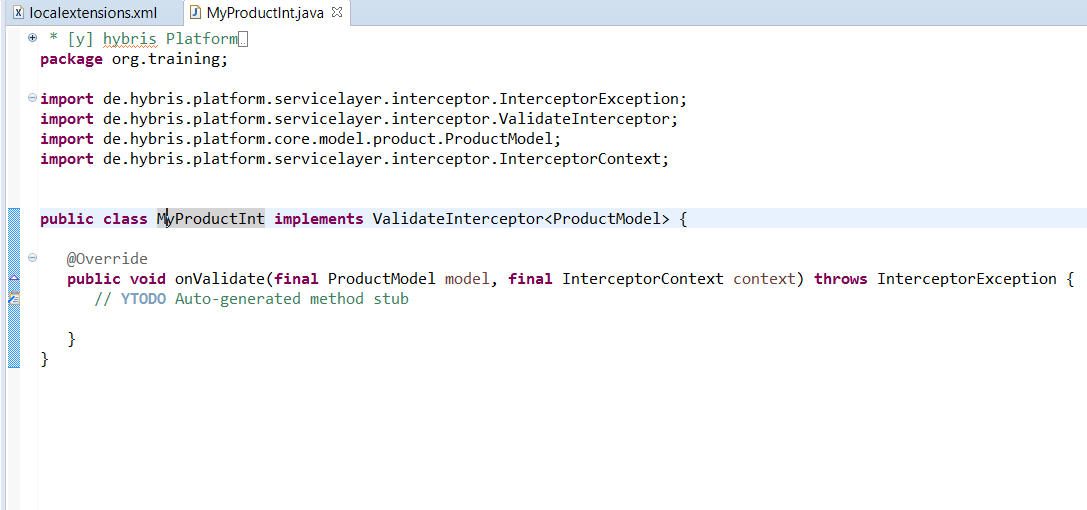


Figure 43: New Interceptor class

Mapping will also be generated in the extension’s spring file.

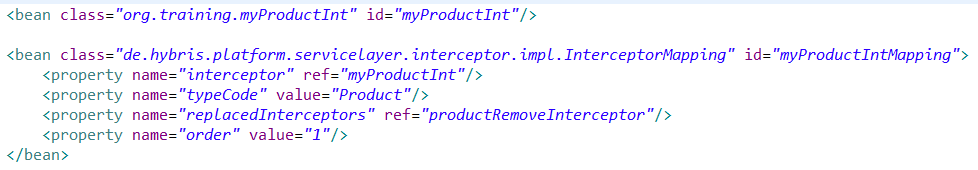


Figure 44: Interceptor mapping