



Accelerators

Generated on: 2020-12-08 14:41:21 GMT+0000

SAP Commerce | 1905

PUBLIC

Original content: <https://help.sap.com/viewer/4c33bf189ab9409e84e589295c36d96e/1905/en-US>

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/viewer/disclaimer>.

Accelerators

SAP Commerce Accelerator is a ready-to-use web implementation template that enables you to jumpstart your implementation and easily build and maintain a feature-rich and flexible commerce solution.

Accelerator Essentials



Discover the core Accelerators and tools.

[Core Accelerator Module](#)

[B2C Accelerator Module](#)

[B2B Accelerator Module](#)

[Accelerator for China](#)

Industry Accelerators



Explore the Accelerators for specific industries that are available with this release.

[Industry Accelerator Compatibility Matrix](#)

[Financial Services Accelerator](#)

[Telco & Utilities Accelerator](#)

[Travel Accelerator](#)

Resources



Find further information to help you with your Accelerator implementation.

[Technical Design Guide](#)

[Wireframes](#)

B2C Accelerator Module

The B2C Accelerator module lets you set up fully-functional apparel and electronics storefronts.

Features



[Multi-Dimensional Products](#)

Architecture



[apparelstore Extension](#)
[electronicsstore Extension](#)

Implementation



[Removing Apparel and the Apparel Stores](#)

B2C Accelerator Features

The B2C Accelerator module provides the multi-dimensional products feature, which allows you to configure products to include multiple attributes.

[Multi-Dimensional Products](#)

This page describes how to build and and expand multi-dimensional products.

Multi-Dimensional Products

This page describes how to build and and expand multi-dimensional products.

Overview

Multi-dimensional products have multiple attributes. Multi-dimensional products are described using multiple Categorys (dimensions) and multiple VariantValueCategorys (attributes).

To understand the difference between multi-dimensional products and variants, consider the following:

- Products in Powertools are multi-dimensional because they include a group products that can be indirectly related. For example, you can list safety footwear a being related to powertools.
- Products in Electronics are multi-dimensional because they include a group of products that match that cateory. For example, electronics can include digital cameras, music players, GPS devices, etc.
- Products in the Apparel store contain variants, which are the same basic model of a product, but that differ in aspects (color, size, pattern, etc.).

i Note

For more information, see the following topics:

- [B2B Multi-Dimensional Products](#)

Building Product Variants by Adding Dimensions

A product variant is a product that varies from the base product in some way that creates uniqueness but not independence. The VariantCategory and VariantValueCategory Object Diagram below shows how the VariantCategory dimensions color, size, and fit are used to describe the different aspects of a product. A product can have as many as three dimensions in the out-of-the-box version of SAP Commerce Accelerator. However, they do not have to be color, size, and fit. As you define a product, you can define your attributes as weight, length, or width for example or any other three dimensions. You define the dimensions in the Backoffice Administration Cockpit or through ImpEx when you define the product; see [Creating Multi-Dimensional Products](#). If you need more than three attributes, see [Creating Additional Dimensions for Multi-Dimensional Products](#) for a description of how to create and add them.

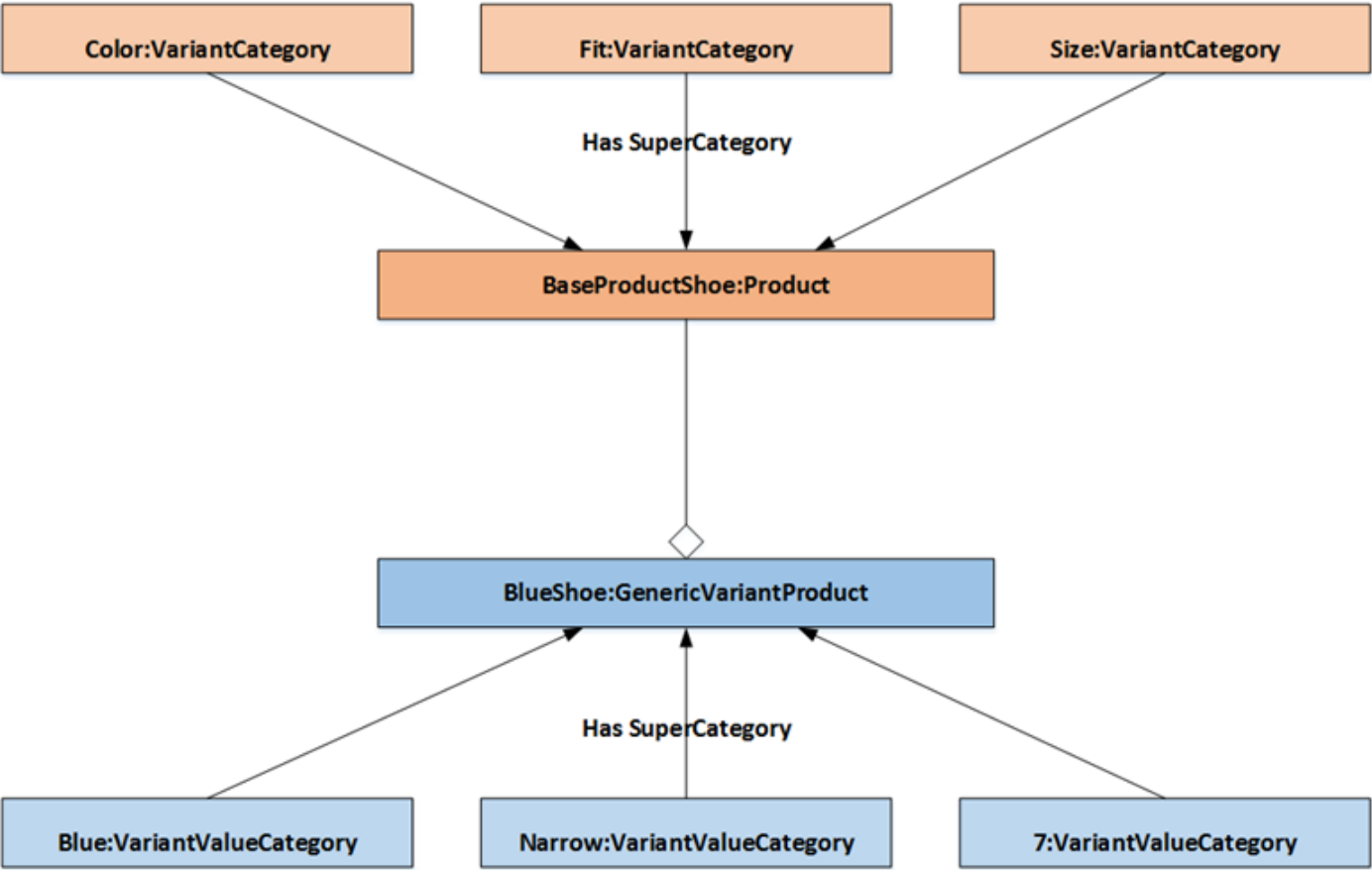


Figure: VariantCategory and VariantValueCategory Object Diagram

The following table describes the elements within the diagram.

Element	Description
---------	-------------

Element	Description
Product	This is the base product, such as a shoe. Base products begin with no <code>VariantValueCategory</code> s (attributes).
<code>GenericVariantProduct</code>	A <code>GenericVariantProduct</code> describes a product with at least one <code>VariantValueCategory</code> (attribute), such as red.
<code>SuperCategory</code>	A <code>SuperCategory</code> is a root category that serves as a container for other subordinate categories. The subordinate categories all share something in common with the <code>SuperCategory</code> .
<code>VariantCategory</code>	A <code>VariantCategory</code> is equivalent to one of the dimensions, such as size, color, or fit. <code>VariantCategory</code> s are containers for <code>VariantValueCategory</code> s.
<code>VariantValueCategory</code>	A <code>VariantValueCategory</code> is a product attribute that describes something about the product, such as red, green, large, small, narrow, or wide.

Example

You can use ImpEx or the Backoffice to create a multi-dimensional product. Use the following steps:

1. Create your `VariantCategory`s (dimensions). For a shoe, this might be color, fit, and size. For a drill, this might be power source and chuck size. For a contact lens, this might be power, base curve, diameter, cylinder, axis, add power, color, and brand. Link these to their `SuperCategory`.

i Note

This contact lens example requires extending the base implementation of Accelerator to support additional dimensions. See [Creating Additional Dimensions for Multi-Dimensional Products](#).

2. Create your `VariantValueCategory`s. `VariantValueCategory`s (attributes) are the values of the dimensions. For a shoe, they might be color - brown, fit - narrow, and size - 7, or they might be color - black, fit - medium, and size - 7.

i Note

Even though both examples are size 7, the shoes have other unique characteristics, which define them as separate products.

3. Create your product and add its `SuperCategory`s.

For additional information, see also: [B2B Multi-Dimensional Products](#).

Loading a Multi-Dimensional Product Using ImpEx

You can use ImpEx to load `GenericVariantProducts`, `VariantCategory`s, and `VariantValueCategory`s. For examples, see [Loading a Multi-Dimensional Product Using ImpEx](#).

Indexing Solr for Multi-Dimensional Products

Product search services are provided by the Apache Solr search engine. All the product information must be indexed for Apache Solr to provide search results for multi-dimensional products.

All the parameters used for the indexing process are defined in the `powertoolsstore` extension located in the `/resources/powertoolsstore/import/coredata/store/powertools/solr.impex` file.

Indexing all of the dataset's products is a two step process. First, you must index the base products, which are products with no dimensions, and then you index all of the multi-dimensional products.

1. The following query is run against the platform database and selects only base products by filtering out all the products whose type is not `GenericVariantProduct`. Only base products fit this description.

```
SELECT {PK} FROM {Product} WHERE {*code*} NOT IN(  {{ SELECT {*code*} FROM {GenericVariantProduct} }})
```

2. You now need to capture all the multi-dimensional products and index them. To do this, `de.hybris.platform.commerceservices.search.solrfacetsearch.provider.impl.VariantCategorySource` was modified to collect `VariantValueCategory`s and uses the `de.hybris.platform.commerceservices.search.solrfacetsearch.provider.impl.VariantProductSource`,

which was modified to collect `VariantCategory`s, to retrieve all the multi-dimensional products and make them available for indexing.

Validation Rules for the Multi-Dimensional Structure

Validation rules are used by the Backoffice, any cockpit, and properly configured `ImpEx` to ensure the integrity of the multi-dimensional data structure. The following rules are enforced by the validators within the `Service Layer`:

- A `GenericVariantProduct` can only have supercategories of type `VariantValueCategory`.
- A `GenericVariantProduct` must have at least one `VariantValueCategory`.
- A `VariantCategory` is composed of `VariantValueCategory`s. While a `VariantCategory` represents a single product attribute, many kinds of this attribute can exist. For example, color is a single product attribute and is represented by a `VariantCategory`, but the product might come in a wide variety of colors. Each of these colors is represented by a single `VariantValueCategory`.
- The attributes defined by a `VariantCategory` have an ordering. This ordering is defined by the order the `VariantCategory` objects are chained. Each `VariantCategory`, if it is not the first in the ordering has another `VariantCategory` in its supercategories. For reordering of the attributes you need to redefine the supercategories of the `VariantCategory` objects.
- When a `VariantCategory` has `VariantValueCategory`s, the `VariantValueCategory`s appear in a specific order. This order is defined by a `VariantValueCategory Sequence` attribute. If your `VariantValueCategory`s contain five colors, for example red, blue, purple, black, and green, then these colors always appear in that Sequence, and the Sequence cannot be changed when you define the product.
- If a `GenericVariantProduct` refers to several `VariantValueCategory` entries, it is validated against its base product and the `VariantCategory` entries of the base product. The verification prevents you from assigning a `GenericVariantProduct` entry into a wrong set of `VariantValueCategory` entries.
- The ordering of each `VariantCategory` entry is validated as chained. This is true for all `VariantCategory`s of the `BaseProduct` and exists within one setup of a `GenericVariantProduct` entry.

Validators in the Service Layer



The validation rules are only used when using the Backoffice or any Cockpit to save the models through the Service Layer. If you are importing data through `ImpEx`, it is strongly recommended that the `ImpEx` validation be enabled as well. To enable `ImpEx` validation, you must deactivate the legacy `ImpEx` mode by setting the following property to false:

```
impex.legacy.mode=false
```

With this deactivated, `ImpEx` imports using the SAP Service Layer, which triggers the multi-dimensional product validators.

Product Detail Order Grid

The Product Detail Order Grid is used to display a multi-dimensional product. This grid is used within the Product Detail Page for a multi-dimensional product and makes it easy to place an order for more than one SKU at a time. As shown below, you simply enter the quantity for each product into the grid and then click [Add to Cart](#).





Welcome Anthony | [My Account](#) | [Sign Out](#) | [Find a Store](#)
Call us: +1 302 295 5067 | [English](#)

I'm looking for

POWER DRILLS | ANGLE GRINDERS | SCREWDRIVERS | SANDERS | MEASURING & LAYOUT TOOLS | HAND TOOLS | SAFETY

Home > Open Catalogue > Tools > Safety > Footwear > Women's




TITAN WOMEN'S WP
\$85.00 - \$85.00

Order Form Total

(0 items) 0.00

In Stock ■ Availability ■ Out of Stock ■

UPDATE FUTURE


Brown

YOUR PRICE

M

(AVAILABILITY)

SIZE	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10	11
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00
M	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	351	332	434	103	302	5	1	363	358	435	146

YOUR PRICE

W

(AVAILABILITY)

SIZE	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10	11
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00
W	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>		<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	145	71	404	447	60	119	403	147		28	312

Subtotal:

 0.00

Average Price / Unit:

 0.00

Quantity:

 0

i Note

To view the order form, the user must be logged in to the site.

The `ProductPageController` has been modified to render the configured product into an Product Detail Order Grid based on its Backoffice definition.

Support for Any Product

The [Add to Cart](#) button was changed to support the selection of single or multiple Stock Keeping Units (SKUs). The required changes were made to the `CartFacade` and the `AddToCartController`. The modifications were made because the addition of Product Detail Order Grid required an Add to Cart functionality that could handle multiple products with one [Add to Cart](#) event.

See also:

- [Add to Cart in the Product Detail Order Grid](#)
- [commercefacades Extension](#)

Support for more than Three Dimensions

While Accelerator supports three dimensions out-of-the-box, it can support addition dimensions. Using `ImpEx` or the Backoffice, you can extend the model to support any number of dimensions. This also requires refactoring of the Product detail page, the Order Form page, and the Shopping Cart.

For more information about adding dimensions to your model, see [Creating Additional Dimensions for Multi-Dimensional Products](#).

Related Information

[Modeling Product Variants](#)

[Styles and Size Variants in the SAP Commerce Product Cockpit](#)

[Catalog Guide](#)

Loading a Multi-Dimensional Product Using ImpEx

This document provides three sample scripts using the sample data from SAP Commerce Accelerator that can be used to load `VariantValueCategories` or `GenericVariantProducts` using ImpEx.

Creating a VariantCategory Using ImpEx

The following code snippet demonstrates how to load a `VariantCategory` using ImpEx.

```
# Macros

$productCatalog=powertoolsProductCatalog
$catalogVersion=catalogversion(catalog(id[default=$productCatalog]),version[default=$catalogVersion])
$supercategories=supercategories(code, $catalogVersion)
$baseProduct=baseProduct(code,$catalogVersion)
$approved=approvalstatus(code)[default='approved']
$priority=variantCategoryPriority(code)

# Language
$lang=en

# Insert Variant Category
INSERT_UPDATE VariantCategory;code[unique=true];name;$supercategories;hasImage;$baseProduct;B2B_Color;Color;;true
;B2B_Fit;Fit;B2B_Color;false
;B2B_Size;Size;B2B_Fit;false
```

The `<hasImage>` property: Include `hasImage` if there is a category image to display.

i Note

The ordering of the `VariantCategory` objects is defined by chaining them. In the sample above the order is Color, Fit, and Size. The `VariantCategory` of `B2B_Fit` has `B2B_Color` as a supercategory and `B2B_Size` has `B2B_Fit` as supercategory. This interconnection of the `VariantCategory` objects defines the order for display of the selectors on the front-end and also the priority of the dimensions for the grid view.

Creating VariantValueCategories Using ImpEx

The following code snippet demonstrates how to load a `VariantValueCategory` using ImpEx.

```
# Macros

$productCatalog=powertoolsProductCatalog
$catalogVersion=catalogversion(catalog(id[default=$productCatalog]),version[default=$catalogVersion])
$supercategories=supercategories(code, $catalogVersion)
$baseProduct=baseProduct(code,$catalogVersion)
$approved=approvalstatus(code)[default='approved']
$priority=variantCategoryPriority(code)

# Language
$lang=en

# Insert variant value category
INSERT_UPDATE VariantValueCategory;code[unique=true];name;$supercategories;sequence;B2B_3_5;3.5;B2B_Size;1
;B2B_4;4;B2B_Size;2
;B2B_4_5;4.5;B2B_Size;3
;B2B_M;M;B2B_Fit;1
;B2B_W;W;B2B_Fit;2
;B2B_Black;Black;B2B_Color;1
;B2B_Brown;Brown;B2B_Color;2
```

The `<sequence>` property: This is the sequence or order of appearance of the `VariantValueCategories`.

Creating GenericVariantProducts Using ImpEx

The following code snippet demonstrates how to load a `GenericVariantProduct` using ImpEx.

```
# Macros

$productCatalog=powertoolsProductCatalog
$catalogVersion=catalogversion(catalog(id[default=$productCatalog]),version[default=$catalogVersion])
$supercategories=supercategories(code, $catalogVersion)
$baseProduct=baseProduct(code,$catalogVersion)
$approved=approvalstatus(code)[default='approved']
$priority=variantCategoryPriority(code)

# Language
$lang=en

# Insert Products
INSERT_UPDATE Product;code[unique=true];$supercategories;manufacturerName;variantCode;24097000;1805,B2B_Color,B2B_Fit,B2B_Size;TITAN 6" SOFT TOE;GenericVariantProduct

# Update product name and description
UPDATE Product;code[unique=true];name[$lang=$lang];description[$lang=$lang];$approved;24097000;TITAN 6" SOFT TOE;TITAN 6" SOFT TOE;approved

# Update generic variant product information
INSERT_UPDATE GenericVariantProduct;code[unique=true];$baseProduct;name;summary;code;24097000_1;24097000;TITAN 6" SOFT TOE BROWN 7 M;TITAN 6" SOFT TOE BROWN 7 M
```

The `<variantType>`: The `variantType` needs to be of type `GenericVariantProduct`.

Related Information

[ImpEx](#)

Add to Cart in the Product Detail Order Grid

This document describes the implementation of the [Add to Cart](#) button in the Product Detail Order Grid, its behavior, and the main components that interact to fulfill its functionality.

Overview and Features

The Add to Cart feature was redesigned because the Product Detail Order Grid can push multiple product order requests into a single Add to Cart event.

The following example is from the B2B Accelerator implementation of the order grid.



Direct Attach Waterproof Insulated 8" Steel Toe

Direct Attach Waterproof Insulated 8" Steel Toe Yellow 7

\$85.00

ORDER FORM TOTAL

(0 items) \$0.00

View Details

ADD TO CART


In Stock

Availability

Out Of Stock

Expand

UPDATE FUTURE

 Yellow

Subtotal: \$0.00

Average Price / Unit: \$0.00

Quantity: 0

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
QUANTITY	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	402	510	150	75	402	510	150	75	402	510	150	75	402	510

Multi-dimensional products are grouped into one line in the mini-cart and the cart page. For example, four size 5 shoes and three size 6 shoes, all variants of the same base product, would appear as seven shoes in one line.

Standard Behavior

In the standard process, a user adds one item to the cart at a time. However, multiple SKUs can be added to the cart on the Product Detail Order Grid page, and this is done the same way single SKUs are added in the Order Form page.

Facade Layer

The implementation `DefaultCartFacade` of the interface `CartFacade` defined in the [commercefacades](#) is used for both multi-dimensional and non-dimensional items. The Controller takes care of the action `Add to Cart` in the Product Details page and the Order Form page.

This default behavior can be overridden by creating a new implementation of the `CartFacade` and assigning the alias `cartFacade` in your extension `spring.xml` definition, as follows:

```
<alias name="myCartFacade" alias="cartFacade"/>
<bean id="myCartFacade" class="de.hybris.platform.commercefacades.order.impl.MyC
```

MVC Layer

Controllers

Changes were made to the controller, `AddToCartController`, to allow querying a non-dimensional, single SKU or multi-dimensional, multiple SKUs when called from the web page. This controller reuses the logic for a single SKU and applies it to multiple SKUs.

Related Information

- [commercefacades Extension](#)
- [Using Order Forms](#)

Adding Multi-Dimensional Products to a Cart Using an Order Form

This document describes how to add multi-dimensional products to a cart using an order form.


When adding a multi-dimensional product to the cart, you can create orders that specify different quantities of each product variant. For example, you can display an order form for a particular boot and then order different quantities of size 7, 7.5, 8, and so on. The order form can be accessed from a product page or category listing.


Accessing the Order Form feature


i Note

To view the order form, the user must be logged in to the site.

The following example is of a product page for a multi-dimensional product. The [Order Form](#) button is visible next to the [Add to Cart](#) button.








Direct Attach Waterproof Insulated 8" Steel Toe Yellow 7

\$85.00

Write a Review

Direct Attach Waterproof Insulated 8" Steel Toe Yellow 7

Color



Size

7

Quantity

1

402 in Stock

Future Availability


Order Form

ADD TO CART

Share

NEW PSR 14.4 LI-2

Lightweight and powerful for all screwdrivin work »




PRODUCT DETAILS

REVIEWS

DELIVERY

The following is an example of search results containing a multi-dimensional product. The [Order Form](#) button is only visible for the last product because it is the only multi-dimensional product in the list.




Professional Network Installer Tool Kit

\$117.00

Professional Network Installer Tool Kit - Installation Kit

ADD TO CART




High-speed rotational tool 1415

\$49.00

Energy management
Power supply type: AC

ADD TO CART



Pit Boss 6" Steel Toe

\$85.00 - \$97.00

Pit Boss 6" Steel Toe


Order Form

View Details

12/8/2020

Adding Variants of a Product using the Order Form

Clicking [Order Form](#) displays the Order Form page for that product, which is used to specify quantities for all the product's variants.



Direct Attach Waterproof Insulated 8" Steel Toe

Direct Attach Waterproof Insulated 8" Steel Toe Yellow 7

\$85.00

ORDER FORM TOTAL

(0 items) \$0.00


View Details

ADD TO CART

In Stock Availability Out Of Stock

Expand

UPDATE FUTURE



Yellow

Subtotal: \$0.00

Average Price / Unit: \$0.00


Quantity: 0

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
QUANTITY	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	402	510	150	75	402	510	150	75	402	510	150	75	402	510

After specifying quantities for each variant, the customer clicks [Add to Cart](#) to add the items to the cart. The total quantity of all variants for the product is displayed in a popup window at the cart area at top-right of the page.

In the following example, 18 variants of the steel toe boot were added to the cart. The price range for all the boots is also displayed.

ADDED TO YOUR SHOPPING CART



Direct Attach Waterproof Insulated 6" Steel Toe

Quantity Added 18

\$85.00 - \$97.00

CHECKOUT


Note
Variants of a multi-dimensional product appear as a single group in the mini-cart and in the cart.

The following is an example of the cart with a multi-dimensional product.

RECEIVED PROMOTIONS	ORDER TOTALS
You saved \$52.38 for spending over \$500.00	Subtotal: \$1,746.00
	Savings: \$52.38
	Total: \$1,693.62
	*No taxes are included in the total

To change the quantity of a multi-dimensional product in the cart, the user clicks **Edit Quantities**, which displays the order form grid. The order form displays all variants, even if the variants did not have any quantities ordered, in case the user would like to add others.


UPDATE FUTURE



Black

Size	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$97.00	\$97.00	\$97.00	\$97.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00
M	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	402	510	150	75	402	510	150	75	402	510	150	75	402	510

Subtotal: \$1,746.00
 Average Price / Unit: \$97.00
 Quantity: 18



Brown

Size	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$97.00	\$97.00	\$97.00	\$97.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00
W	<input type="text" value="6"/>	<input type="text" value="6"/>	<input type="text" value="6"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
(AVAILABILITY)	150	75	402	510	150	75	402	510	150	75	402	510	150	75

Subtotal: \$0.00
 Average Price / Unit: 0
 Quantity: 0

Displaying Future Stock Availability

Clicking **Update Future** from the Order Form page fetches the quantities that will be available by a certain date. Pointing at a date displays the future stock for that date.

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$97.00	\$97.00	\$97.00	\$97.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00
W	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>
(AVAILABILITY)	150 7/22/47	75 7/22/47	402 7/22/47	510 7/22/47	Delivery 7/22/47	QTY 1	402 7/22/47	510 7/22/47	150 7/22/47	75 7/22/47	402 7/22/47	510 7/22/47	150 7/22/47	75 7/22/47

The color of the **Availability** row changes depending on availability:

- Green: In stock
- Yellow: Future stock only (no current stock)
- Red: Out of stock

Stock levels will only appear in yellow if current stock is red, future stock exists, and the user clicks **Update Future**.

Creating Additional Dimensions for Multi-Dimensional Products

Learn how to create additional dimensions for multi-dimensional products.

By default, SAP Commerce Accelerator supports up to three dimensions, but you can add any number of dimensions. The structure can be extended to support four, five, six or however many dimensions needed to describe the product. The steps to add more dimensions to the product definition are shown below.

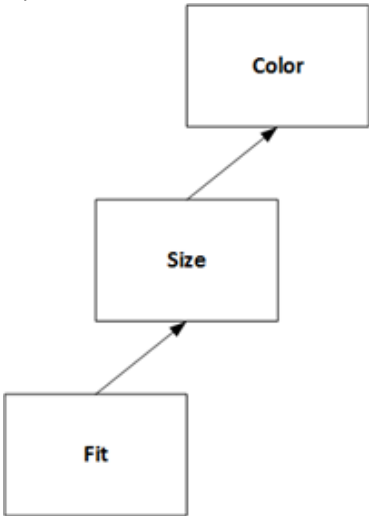
Technical Overview

Multi-dimensional products consist of one, two, three, or more **VariantCategories**. Each category (dimension) is a characteristic of the product such as color, size, fit, weight, texture, or material. Each category is also a container for product attributes (**VariantValueCategory**), such as red, green, medium, large, coarse, fine, leather, or suede, which define the details of the category. Multi-dimensional products can be fully defined through dimensions and attributes.

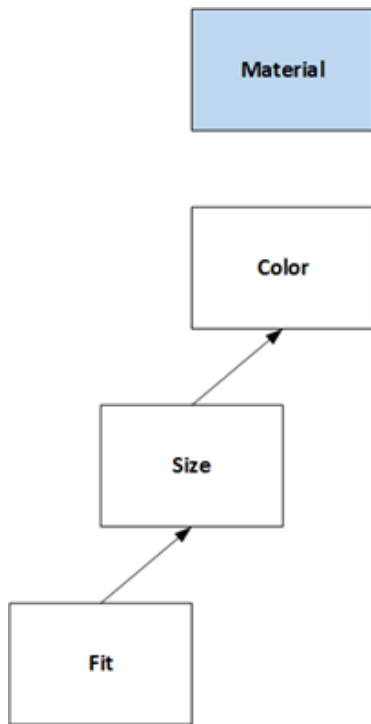
Adding Another Dimension

Commerce Accelerator supports three different dimensions (categories) out-of-the-box. These dimensions are chained together with each one serving as a supercategory for the next lower category in the chain. A **SuperCategory** is a root category for subordinate categories. By adding an additional category, the number of product dimensions can be extended. The example below begins with a product that has three dimensions, and then an additional dimension is added using either **ImpEx** or the **Backoffice**.

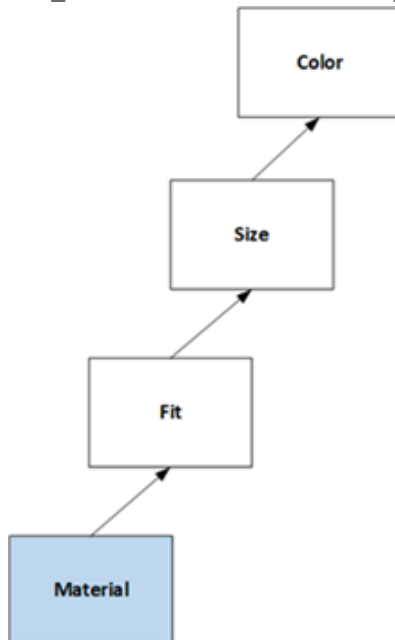
1. A product has three dimensions: **B2B_Color** < **B2B_Size** < **B2B_Fit**.



2. A new dimension, **B2B_Material**, is created.



3. B2B_Material is then linked to B2B_Fit and B2B_Fit is set as B2B_Material's SuperCategory.



4. Through the Backoffice or ImpEx, create the required VariantValueCategories. You would create as many of these as is needed for the product descriptors, such as red, brown, gray, and tan.
5. The Accelerator storefront supports up to three attributes by default, so the Product Detail page, the Order Form page, and the Shopping Cart have to be refactored to support more attributes. Specific files to be edited are:
 - /yacceleratorstorefront/web/webroot/WEB-INF/tags/desktop/product/productVariantSelector.tag
 - /yacceleratorstorefront/web/webroot/WEB-INF/tags/desktop/grid folder, which contains these files:
 - coreTable.tag
 - coreTableHeader.tag
 - grid1dimension.tag
 - grid2dimension.tag
 - grid3dimension.tag
 - Add a grid4dimensions.tag file when adding your fourth dimension.

Related Information

Multi-Dimensional Product Grouping


This document provides a technical overview of how multi-dimensional products are grouped in the cart and mini-cart.

The Multi-Dimensional Product feature of SAP Commerce Accelerator allows buyers to order variants of the same base product using an order form. For example, a buyer can order sizes 7, 8, and 9 of a particular boot at the same time. To keep the cart and mini-cart organized, variants of a multi-dimensional product are grouped under the base product in one line. Users can edit the quantities by displaying the order form from the cart.

Feature Overview

Cart

Multi-dimensional products are added to the cart using the order form available in Commerce B2B Accelerator, as shown in the screenshot below.



Pit Boss 6" Steel Toe

Pit Boss 6" Steel Toe Wheat 7 M

\$97.00

-

\$85.00

ORDER FORM TOTAL

(30 items)

\$2,790.00

View Details

ADD TO CART


In Stock

Availability

Out Of Stock

Expand

UPDATE FUTURE



Black

Subtotal:

\$2,790.00

Average Price / Unit:

\$93.00


Quantity:

30

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
M	10	0	0	0	0	0	0	0	0	0	0	0	0	20
(AVAILABILITY)	402	510	150	75	402	510	150	75	402	510	150	75	402	510


SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(AVAILABILITY)	150	75	402	510	150	75	402	510	150	75	402	510	150	75

All variants are grouped into one line in the cart. The unit price shows the price range if the variants in the cart have different prices.

The following example shows the cart; users can modify the quantities of each variant by clicking .

YOUR CART

CART ID: 00002006

ITEM	ITEM PRICE	QUANTITY	TOTAL
<div><div></div><div>Pit Boss 6" Steel Toe</div></div> <div>\$85.00</div> <div>30</div> <div>\$2,790.00</div> <div><div>+</div><div>Remove</div></div>			

Received Promotions

You saved \$83.70 for spending over \$500.00

ORDER TOTALS

Subtotal: \$2,790.00

Savings: \$83.70

Total: \$2,706.30

*No taxes are included in the total

Continue Shopping

CHECKOUT

Select an alternative checkout flow


Clicking

+


 displays the variant grid below the item, as shown in the following example.

YOUR CART

CART ID: 00002006

ITEM	ITEM PRICE	QUANTITY	TOTAL
<div><div></div><div>Pit Boss 6" Steel Toe</div></div> <div>\$85.00</div> <div>30</div> <div>\$2,790.00</div> <div><div>+</div><div>Remove</div></div>			

UPDATE FUTURE



Black

Subtotal: \$2,790.00

Average Price / Unit: \$93.00

Quantity: 30

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
M	<div>10</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>20</div>
(AVAILABILITY)	402	510	150	75	402	510	150	75	402	510	150	75	402	510

SIZE	7	7.5	8	8.5	9	9.5	10	10.5	11	11.5	12	13	14	15
YOUR PRICE	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$85.00	\$97.00	\$97.00	\$97.00	\$97.00	\$97.00
W	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>	<div>0</div>
(AVAILABILITY)	150	75	402	510	150	75	402	510	150	75	402	510	150	75

Modifying the Grid Quantities

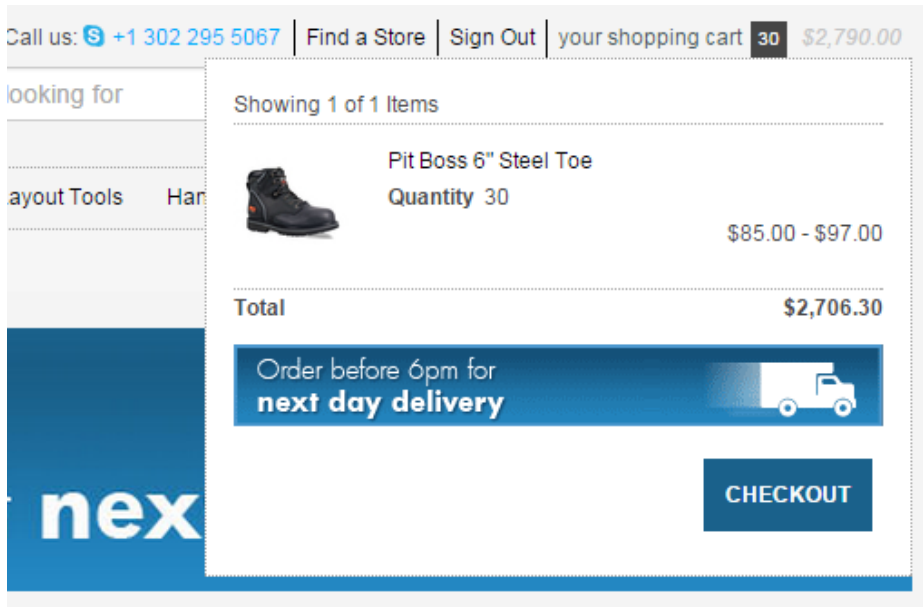
The grid displays the quantities ordered for each variant. Users can update the quantities of each variant in the grid, and all cart data is dynamically updated and refreshed when the quantity in the input field is changed. For example, if the quantity of the size 7 boot is changed from 10 to 0, the quantity, subtotal, average price per unit, and order totals are all automatically updated in the cart.

Checkout Page

The Checkout page is similar to the Cart page, except that the grid appears in a pop-up window and the quantities cannot be edited.

Mini-Cart

Similarly, variants are grouped in the mini cart.



Technical Overview

By default, the `convert` function in the `ycommercewebservices` extension converts `cartModel` to `cartData` as follows:

```
public CartData getSessionCart()
{
    final CartData cartData;
    final CartModel cart = getCartService().getSessionCart();
    cartData = getCartConverter().convert(cart);
    return cartData;
}
```

A list that is used to store the sub-entries of each entry is added to `OrderEntryData` in the `ycommercewebservices-beans.xml` file.

```
<bean class="de.hybris.platform.ycommercewebservices.order.data.OrderEntryDataList">
    <property name="orderEntries" type="java.util.List<de.hybris.platform.commerci
</bean>
```

Variants are put into the sub-entries when they are grouped into one item.

Grouping multi-dimensional products occurs when `cartModel` is converted to `cartData`. The default converters (`cartConverter` and `orderConverter`) are replaced by `groupedCartConverter` and `groupedOrderConverter`, and a `groupOrderEntryPopulator` is added to their populator list. The bean definition to achieve product grouping is located in the `commercefacades-spring.xml` file:

```
...

<bean id="groupOrderEntryPopulator"
class="de.hybris.platform.b2bacceleratorfacades.order.converters.populator.Group
<property name="priceDataFactory" ref="priceDataFactory"/>
<property name="productService" ref="productService" />
</bean>

<alias name="groupedOrderConverter" alias="orderConverter"/>
<bean id="groupedOrderConverter" parent="defaultOrderConverter">
<property name="populators">
<list merge="true">
<ref bean="orderPopulator"/>
<ref bean="groupOrderEntryPopulator"/>
<ref bean="orderConsignmentPopulator"/>
</list>
</property>
</bean>

<alias name="groupedCartConverter" alias="cartConverter"/>
<bean id="groupedCartConverter" parent="defaultCartConverter">
<property name="populators">
<list merge="true">
```

```
<ref bean="cartPopulator"/>
<ref bean="groupOrderEntryPopulator"/>
</list>
</property>
</bean>

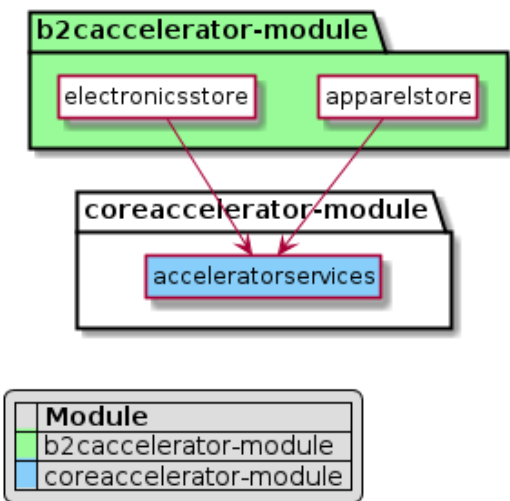
...

```

B2C Accelerator Architecture

The B2C Accelerator module is a set of extensions providing all the sample data necessary to set up fully functioning storefronts. The included data sets are for an apparel storefront and an electronics storefront.

Dependencies



Dependencies Diagram

Recipes

The following recipes contain the B2C Accelerator module:

- b2c_acc_cis
- b2c_acc_plus
- b2c_acc_ymkt
- b2c_b2b_acc_cpq
- b2c_b2b_acc_dp
- b2c_b2b_acc_oms
- b2c_c4c
- b2c_china
- sap_aom_som_b2b_b2c
- sap_oms_aom_b2b_b2c

For further information, see [Installer Recipes](#).

Extensions

The B2C Accelerator module consists of the following extensions:

[apparelstore Extension](#)

The Apparel sites demonstrate the use of Variants and how to set up multiple sites that support different currencies and languages on a single storefront. They offer shipping to multiple countries and have separate tax rules set at the store point of sale country.

[electronicsstore Extension](#)

The electronics storefront offers shipping to multiple countries, it has a tax rule set at the store point of sale country.

apparelstore Extension

The Apparel sites demonstrate the use of Variants and how to set up multiple sites that support different currencies and languages on a single storefront. They offer shipping to multiple countries and have separate tax rules set at the store point of sale country.

The `apparelstore` extension adds the necessary dataset for the Accelerator reference sites Apparel DE and Apparel UK. This dataset was previously present in the `acceleratorsampledata` extension. This extension was deprecated in favor of smaller extensions representing only one store.

→ **Tip**

This Extension contains sample data for the B2C Apparel storefronts. You can find the description of:

- B2C Electronics sample data in the [electronicsstore Extension](#) document
- B2B sample data in the [powertoolsstore Extension](#) document

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.

Extension Definition

Name	apparelstore
Description	It loads sample and core data for reference apparel storefronts.
Requires	The <code>apparelstore</code> Extension depends on the acceleratorservices Extension .
Author	SAP

Supported Markets and Channels

The **apparelstore** Extension is specifically designed for B2C market. It can be used for both, desktop and mobile, channels.

Supported	B2C Commerce	B2B Commerce	Telco Commerce
Market	✔	✖	✖
Channel	Desktop: ✔ Mobile: ✔	Desktop: ✖ Mobile: ✖	Desktop: ✖ Mobile: ✖

Hooks for System Initialization

The **apparelstore** extension implements the **AbstractSystemSetup** class in the **ApparelStoreSystemSetup**. When initialization is triggered, the **createProjectData** method will be called. This method will then call the **CoreDataImportService** and **SampleDataImportService** (in the **yacceleratorinitialdata** extension) to trigger the import of the different ImpEx files.

```
/**
 * This method will be called during the system initialization.
 *
 * @param context the context provides the selected parameters and values
 */
@SystemSetup(type = SystemSetup.Type.PROJECT, process = SystemSetup.Process.ALL)
public void createProjectData(final SystemSetupContext context)
{
    // ...
}
```

12/8/2020

```
final ImportData apparelImportData = new ImportData();
apparelImportData.setProductCatalogName(APPAREL);
apparelImportData.setContentCatalogNames(Arrays.asList(APPAREL_UK, APPAREL_DE));
apparelImportData.setStoreNames(Arrays.asList(APPAREL_UK, APPAREL_DE));

getCoreDataImportService().importData(context, apparelImportData);
getEventService().publishEvent(new CoreDataImportedEvent(context, Arrays.asList(apparelImportData)));

getSampleDataImportService().importData(context, apparelImportData);
getEventService().publishEvent(new SampleDataImportedEvent(context, Arrays.asList(apparelImportData)));
}
```

UK Apparel Store

The following is a list of data loaded by the **apparelstore** extension for the sample UK Apparel store:

- Core data:
 - Essential data like languages, currencies, titles
 - Empty catalogs
 - CMS components
 - Email templates
 - Tax rows
 - Cart removal jobs
 - Store
 - Delivery costs
- Sample data:
 - Shared product catalog (with DE Apparel) with content in two languages:
 - Snowboard and surf
 - Product data model:
 - Variants
 - Exclusive content catalog with content in one language
 - Languages:
 - English GB
 - Currencies:
 - GBP
 - Gross prices (using decoupled price rows, as described in [Decoupling PDTRows from Product](#))
 - Shipping:
 - UK territory only
 - Site theme:
 - Black
 - Enhanced SOLR configuration (shared with DE Apparel)
 - Some promotions

DE Apparel Store








The following data is loaded by the **apparelstore** extension for the sample DE Apparel Store store:








- Core data:
 - Essential data like languages, currencies, titles
 - Empty catalogs
 - CMS components
 - Email templates



- Tax rows
 - Cart removal jobs
 - Store
 - Delivery costs
- Sample data:
 - Shared product catalog (with UK Apparel) with content in two languages:
 - Snowboard and Surf
 - Product data model:
 - Variants
 - Exclusive content catalog with content in one language
 - Language:
 - German (default)
 - Currencies:
 - EUR
 - Gross prices (using decoupled price rows, as described in [Decoupling PDTRows from Product](#))
 - Shipping:
 - Continental Europe
 - Site theme:
 - Apparel
 - Enhanced Solr configuration (shared with UK Apparel)
 - Some promotions

Modifications Checklist

The modifications that this Extension makes to the Accelerator are listed below:

Impex Configuration Scripts	
Core Data Listeners	
Model Layer	
Model Interceptors	
Cockpit Configuration	
Cockpit Beans	
Validation Rules	

Service Layer	
Facade DTO	
Facade Layer	
CMS Components	
Page Templates	
JavaScript	
CSS	

Page Controllers	
Tags	

TLD	-
Filters	-
MVC Interceptors	-
Spring Security	-
Message Resources	-

Responsive Sample Data Loading Mechanism

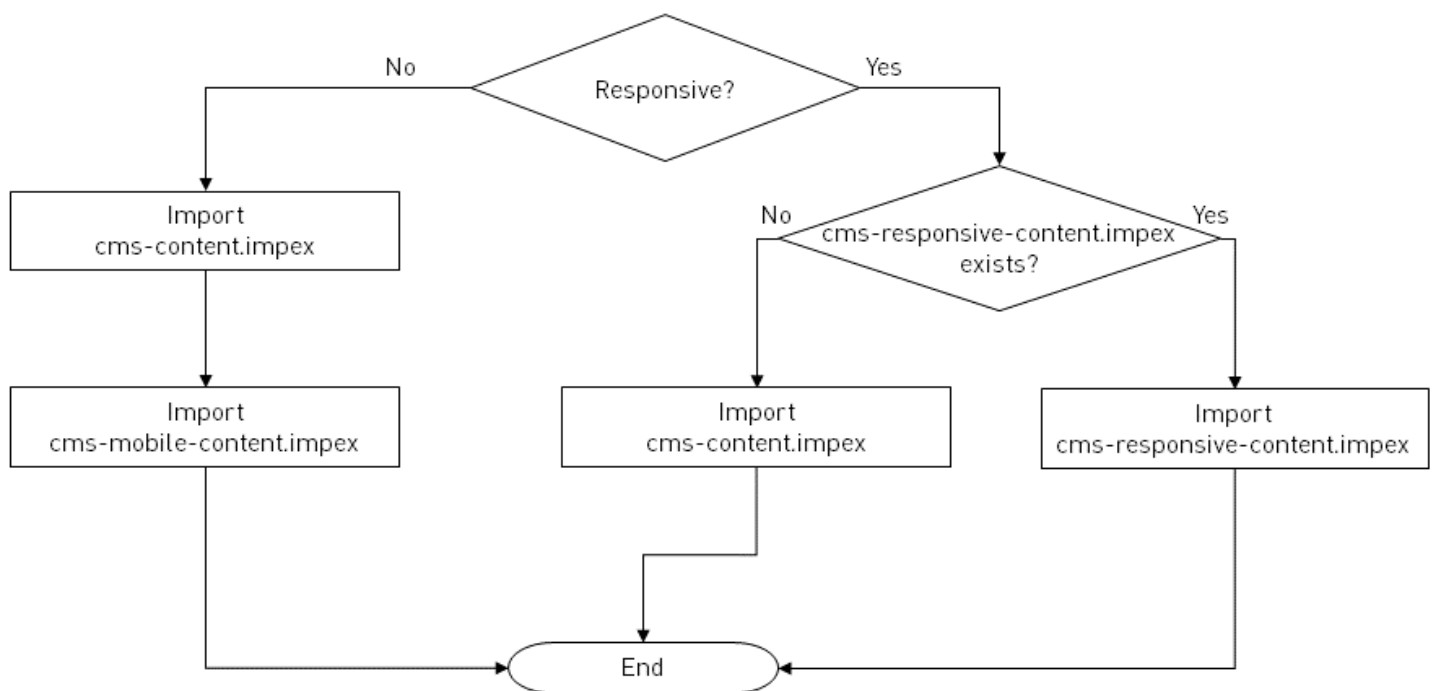
The apparel responsive sample data is self-contained resulting in faster initialization and cleaner sample data. This approach forms the basis for future improvements in features and tools such as SAP Commerce SmartEdit.

i Note

The changes described in this topic apply to the apparel store sample data only (both UK and German). The electronics storefront uses the same loading mechanism as before.

To streamline loading and initiation, the responsive sample data is self-contained and the import functions of the sample store's `contentCatalog` are modified. Desktop, responsive, and mobile all use the same `productCatalog` data, common data, store data, and Solr index data.

The loading function works as follows:



Localization

All the required `cms-responsive-content.vt`, `cms-responsive-content_en.properties` (for the apparel-uk storefront), and `cms-responsive-content_de.properties` (for the apparel-de storefront) are added in `resource-lang`, so all the localized impex files are auto-generated.

electronicsstore Extension

The electronics storefront offers shipping to multiple countries, it has a tax rule set at the store point of sale country.

The `electronicsstore` extension adds the necessary dataset for the Accelerator reference Japanese Electronics site. This dataset was previously present in the `acceleratorSampleData` extension. This extension was deprecated in favor of smaller extensions representing only one store.

→ Tip

This Extension contains sample data for the B2C Japanese Electronic storefront. You can find the description of:

- B2C Apparel sample data in the [apparelstore Extension](#) document
- B2B sample data in the [powertoolsstore Extension](#) document

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.

Extension Definition

Name	electronicsstore
Description	It loads sample and core data for reference electronic storefront.
Requires	The electronicsstore Extension depends on the accelerator services Extension .
Author	SAP

Supported Markets and Channels

The **electronicsstore** Extension is specifically designed for B2C market. It can be used for both, desktop and mobile, channels.

Supported	B2C Commerce	B2B Commerce	Telco Commerce
Market	✓	✗	✗
Channel	Desktop: ✓	Desktop: ✗	Desktop: ✗
	Mobile: ✓	Mobile: ✗	Mobile: ✗

Hooks for System Initialization

The **electronicsstore** extension implements the **AbstractSystemSetup** class in the **ElectronicsStoreSystemSetup**. When initialization is triggered **createProjectData** method will be called. This method will then call the **CoreDataImportService** and **SampleDataImportService** (in the **yacceleratorinitialdata** extension) to trigger the import of the ImpEx files.

```
/**
 * This method will be called during the system initialization.
 *
 * @param context the context provides the selected parameters and values
 */
@SystemSetup(type = SystemSetup.Type.PROJECT, process = SystemSetup.Process.ALL)
public void createProjectData(final SystemSetupContext context)
{
    final ImportData electronicsImportData = new ImportData();
    electronicsImportData.setProductCatalogName(ELECTRONICS);
    electronicsImportData.setContentCatalogNames(Arrays.asList(ELECTRONICS));
    electronicsImportData.setStoreNames(Arrays.asList(ELECTRONICS));

    getCoreDataImportService().importData(context, electronicsImportData);
    getEventService().publishEvent(new CoreDataImportedEvent(context, Arrays.asList(electronicsImportData)));

    getSampleDataImportService().importData(context, electronicsImportData);
    getEventService().publishEvent(new SampleDataImportedEvent(context, Arrays.asList(electronicsImportData)));
}
```








Data for Japanese Electronics Store








The following is a list of data loaded by the **electronicsstore** extension for the sample Japanese Electronics store:








- Core data:
 - Essential data like languages, currencies, titles
 - Empty catalogs
 - CMS components
 - Email templates
 - Tax rows
 - Cart removal jobs
 - Store
 - Delivery costs
- Sample data:
 - Exclusive Product Catalog with content in three languages:
 - Electronics
 - Product Data Model :
 - Vanilla with classification system
 - Exclusive WCMS content catalog with content in three languages
 - Languages:
 - Japanese
 - English (default)
 - German
 - Currencies:
 - JPY
 - USD (default)
 - Gross prices (using decoupled price rows, as described in [Decoupling PDTRows from Product](#))
 - Shipping:
 - Japan
 - USA
 - Europe
 - Enhanced Solr configuration
 - Advanced Personalization samples
 - Promotions
 - Customer review samples
 - Cross-sells

Modifications Checklist

The modifications that this Extension makes to the Accelerator are listed below:

Impex Configuration Scripts	
Core Data Listeners	
Model Layer	
Model Interceptors	
Cockpit Configuration	
Cockpit Beans	
Validation Rules	

Service Layer	
Facade DTO	
Facade Layer	
CMS Components	
Page Templates	
JavaScript	
CSS	

Page Controllers	
Tags	
TLD	
Filters	
MVC Interceptors	
Spring Security	
Message Resources	

B2C Accelerator Implementation

The B2C Accelerator module allows you to remove apparel store functionality from your installation of SAP Commerce Accelerator.

[B2C Store Extensions Related Data](#)

The other extensions providing data for the project are the `apparelstore` and `electronicsstore` AddOn extensions and `yacceleratorinitialdata` extension. They come with a set of only Project data.

[Removing Apparel and the Apparel Stores](#)

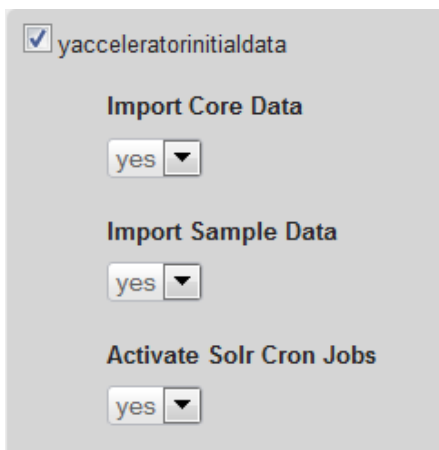
You can remove apparel support from your SAP Commerce Accelerator.

B2C Store Extensions Related Data

The other extensions providing data for the project are the `apparelstore` and `electronicsstore` AddOn extensions and `yacceleratorinitialdata` extension. They come with a set of only Project data.

B2C Store Extensions Project Data

Project data setup for the `yacceleratorinitialdata` extension provides the following options:



☒ **yacceleratorinitialdata**

Import Core Data
yes ▼

Import Sample Data
yes ▼

Activate Solr Cron Jobs
yes ▼

Import Core Data loads:

12/8/2020

- Essential data: warehouses, delivery modes, and currencies
- CMS templates
- Empty catalogs
- Email templates

Import Sample Data loads:

- Products
- Cockpits users
- CMS content
- Promotions
- Points of service

Activate Solr Cron Jobs runs the Solr cron jobs.

Project data setup for the sample data extensions contain:

- Site and store configuration: catalogs, classifications, countries, languages, currencies, delivery options, themes, media formats, and tax groups.
- Base CMS content: page templates, page types, pages, components, and cockpit configurations.
- Solr search configuration: index configurations, indexed types and properties, ranges, queries, and sorts.
- Products
- Product categories
- Promotions
- Product and category media
- Product stock levels
- CMS content, including sample components, pages, images, and paragraphs.
- Email content
- Points of Service (POS): stores and associated media.
- Cockpit users
- BTG rules
- Sample product reviews

Resource Folder Structure

The resource folders of the `apparelstore` and `electronicsstore` AddOn extensions are divided into two parts :

1. Core data: create empty sites, empty catalogs, and essential data
2. Sample data: populate catalogs with products, import users, and user rights

Products

The sample data provided with the SAP Commerce Accelerator demonstrates two approaches to defining a product catalog in the system:

- Using [Classifications](#) in **electronics** shop
- Using [Variants](#) in **apparel** shop

CMS Content

The sample CMS content for both sites is almost identical but using different media and themes in the two sites makes them look totally different.

Points of Sale

Points of Sale are loaded into the Electronics shop, Apparel UK and Apparel DE separately, so searching in one Apparel site, for example, does not return results from the other.

Promotions

Sample promotions are included that apply for specific products, specific categories and across entire orders.

Related Information

[yacceleratorcore Extension](#)

[acceleratorservices Extension](#)

[apparelstore Extension](#)

[electronicsstore Extension](#)

Removing Apparel and the Apparel Stores

You can remove apparel support from your SAP Commerce Accelerator.

The SAP Commerce Accelerator offers a reference storefront with an Apparel functionality that supports product variants out of the box. If your project does not require apparel support, you can remove this functionality. This topic explains how to remove Apparel from the Commerce Accelerator source code, which includes the Apparel data model, any Apparel services and facades, as well as the Apparel sample storefronts configuration.

Before starting the workflow described here, see [Customizing the Accelerator with extgen and modulegen](#) to get an overview of working with template extensions.

The procedure for removing the Apparel features depends on the extensions they are related to. The following sections provide the procedure for each extension.

yacceleratorcore Extension

Removing Apparel features related to the `yacceleratorcore` extension is executed in several steps and in a specific order. Remove Apparel features in the following order:

1. Remove the data model.
2. Hot folder batch import
3. Remove beans related to Solr Search

Remove the Data Model

To remove the data model, you start by removing the data model changes and any associated configurations for the cockpits as follows.

1. In the `yacceleratorcore/resources/yacceleratorcore-items.xml` file, remove everything inside the **Apparel** typegroup :

```
<typegroup name="Apparel">
```

```
<itemtype code="ApparelProduct" extends="Product"
autocreate="true" generate="true"
jaloclass="de.hybris.platform.yacceleratorcore.jalo.Apparel
<description>Base apparel product extension that contains a
<attributes>
<attribute qualifier="genders" type="GenderList">
<description>List of genders that the ApparelProduct is des
<modifiers />
<persistence type="property" />
</attribute>
</attributes>
</itemtype>
```

```
<itemtype code="ApparelStyleVariantProduct" extends="Variar
autocreate="true" generate="true"
jaloclass="de.hybris.platform.yacceleratorcore.jalo.Apparel
<description>Apparel style variant type that contains addit
<attributes>
```

```

<attribute qualifier="style" type="localized:java.lang.Stri
metatype="VariantAttributeDescriptor">
<description>Colour/Pattern of the product.</description>
<modifiers />
<persistence type="property" />
</attribute>

<attribute qualifier="swatchColour" type="java.lang.String"
<description>A normalised colour mapping to a standardised
<modifiers />
<persistence type="property" />
</attribute>
</attributes>

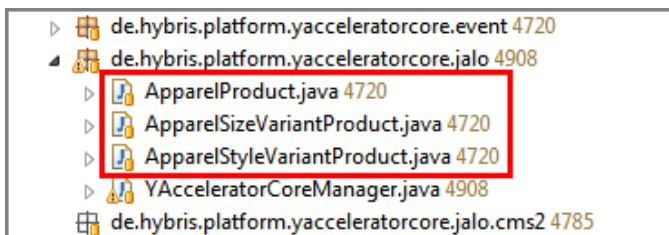
</itemtype>

<itemtype code="ApparelSizeVariantProduct" extends="Apparel
autocreate="true" generate="true"
jaloclass="de.hybris.platform.yacceleratorcore.jalo.Apparel
<description>Apparel size variant type that contains additi
<attributes>
<attribute qualifier="size" type="localized:java.lang.Strir
metatype="VariantAttributeDescriptor">
<description>Size of the product.</description>
<modifiers />
<persistence type="property" />
</attribute>
</attributes>

</itemtype>
</typegroup>

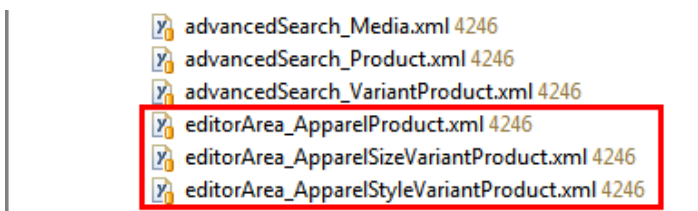
```

2. Remove the **ApparelProduct**, **ApparelSizeVariantProduct**, and **ApparelStyleVariantProduct** classes from `yacceleratorcore/src/de/hybris/platform/yacceleratorcore/jalo`.



3. Remove the following Apparel-specific Product Cockpit editor configuration XML files from the `yacceleratorcockpits/resources/yacceleratorcockpits-config/cockpitgroup` folder:

- `editorArea_ApparelProduct.xml`
- `editorArea_ApparelSizeVariantProduct.xml`
- `editorArea_ApparelStyleVariantProduct.xml`



4. Remove the **BatchIntegrationTest.testVariant** test method from the `yacceleratorcore` extension.
5. Execute the `ant clean all` command in the `yacceleratorcore` extension to ensure the extension still compiles.

Hot Folder Batch Import

You must remove the back import instance for the Apparel store. To do so, remove the `yacceleratorcore/resources/yacceleratorcore/integration/hot-folder-store-apparel-spring.xml` file.

Solr Search

When you delete the store content, the scripts that create the Solr index are removed. You must also remove specific value-provider Spring beans from the `yacceleratorcore` extension application context.

1. Remove the **CategorySource** beans from the `yacceleratorcore/resources/yacceleratorcore-spring.xml` file:

```
<bean id="apparelCategorySource" parent="abstractCategorySource">
    <property name="rootCategory" value="categories"/> <!-- 'categories' -->
</bean>
<bean id="apparelBrandCategorySource" parent="abstractCategorySource">
    <property name="rootCategory" value="brands"/> <!-- 'brands' -->
</bean>
```

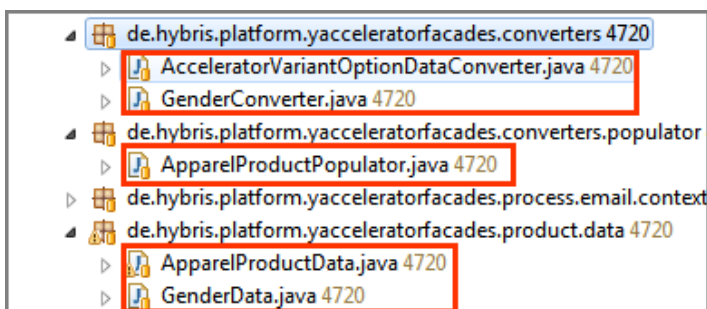
2. Remove the **ValueProvider** beans from the `yacceleratorcore/resources/yacceleratorcore-spring.xml`:

```
<bean id="apparelCategoryCodeValueProvider" parent="abstractCategoryCodeValueProvider">
    <property name="categorySource" ref="apparelCategorySource"/>
</bean>
<bean id="apparelBrandCategoryCodeValueProvider" parent="abstractCategoryCodeValueProvider">
    <property name="categorySource" ref="apparelBrandCategorySource"/>
</bean>
<bean id="apparelCategoryNameValueProvider" parent="abstractCategoryNameValueProvider">
    <property name="categorySource" ref="apparelCategorySource"/>
</bean>
<bean id="apparelBrandCategoryNameValueProvider" parent="abstractCategoryNameValueProvider">
    <property name="categorySource" ref="apparelBrandCategorySource"/>
</bean>
```

yacceleratorfacades Extension

Most of the code in the `yacceleratorfacades` extension template is provided to support Apparel functionality. This procedure assumes that you have not adapted any of the code you plan to delete for purposes other than Apparel.

1. Remove the **AcceleratorVariantOptionDataConverter** and **GenderConverter** classes from the `yacceleratorfacades/src/de/hybris/platform/yacceleratorfacades/converters` folder of the `yacceleratorfacades` extension.
2. Remove the **ApparelProductPopulator** class from the `yacceleratorfacades/src/de/hybris/platform/yacceleratorfacades/converters/populator` folder of the `yacceleratorfacades` extension.
3. Remove the **ApparelProductData** and **GenderData** classes from the `yacceleratorfacades/src/de/hybris/platform/yacceleratorfacades/product/data` folder of the `yacceleratorfacades` extension:



4. Remove the Spring bean configuration for all of these classes in the `yacceleratorfacades/resources/yacceleratorfacades-spring.xml` file.

```
<alias name="acceleratorVariantOptionDataConverter" alias="variantOptionDataConverter"/>
<bean id="acceleratorVariantOptionDataConverter" class="de.hybris.platform.yacceleratorfacades.converters.AcceleratorVariantOptionDataConverter">
    <property name="mediaService" ref="mediaService"/>
    <property name="mediaContainerService" ref="mediaContainerService"/>
    <property name="typeService" ref="typeService"/>
    <property name="imageFormatMapping" ref="imageFormatMapping"/>
    <property name="variantAttributeMapping">
        <map>
            <entry key="ApparelStyleVariantProduct.style" value="styles"/>
        </map>
    </property>
</bean>
```

```

<alias name="acceleratorGenderConverter" alias="genderConve
<bean id="acceleratorGenderConverter" class="de.hybris.plat
<property name="typeService" ref="typeService"/>
</bean>

<bean id="apparelProductPopulator" class="de.hybris.platfor
<property name="genderConverter" ref="genderConverter"/>
</bean>

<alias name="acceleratorProductConverter" alias="productCor
<bean id="acceleratorProductConverter" class="de.hybris.pla
<property name="apparelProductPopulator" ref="apparelProduc
</bean>

```

5. Execute the command `ant clean all` in the `yacceleratorfacades` extension to ensure the extension still compiles.

yacceleratorstorefront Extension

The storefront controllers are agnostic of apparel, therefore few changes are required here.

Spring Application Context

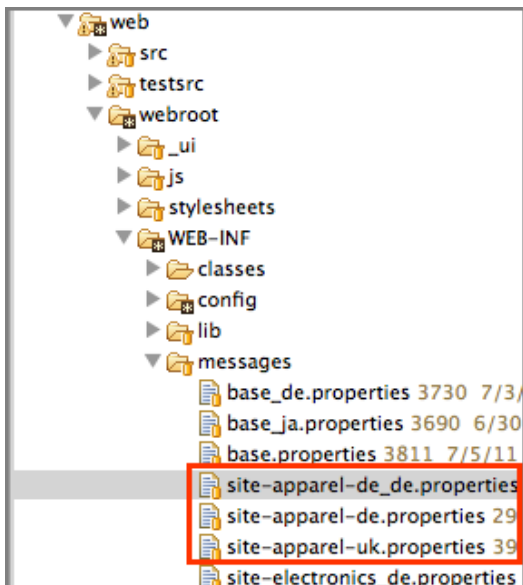
In the `yacceleratorstorefront/web/webroot/WEB-INF/config/spring-mvc-config.xml` file of the `yacceleratorstorefront` extension, remove the following line:

```
<entry key="size" value-ref="sizeAttributeComparator"/>
```

Site Resource Files

Remove the following apparel site resource files, which are located in the `yacceleratorfacades/web/webroot/WEB-INF/messages` folder:

- `site-apparel-de_de.properties`
- `site-apparel-de_en.properties`
- `site-apparel-uk_en.properties`



apparelstore Extension

Remove the `apparelstore` extension from the `localextensions.xml` file.

Recompiling

Execute the command `ant clean all` in the `apparelstore` extension to ensure the extension still compiles.

Reinitializing

Reinitialize the SAP Commerce Platform database to remove the apparel data model from the type system and any orphaned data no longer added by the scripts. For more information on initialization, see the **Initialize System** section of the [Initialization and Update of the SAP Commerce](#) document.

Related Information

[yacceleratorcore Extension](#)

[acceleratorfacades Extension](#)




[yacceleratorstorefront Extension](#)

[apparelstore Extension](#)

[electronicsstore Extension](#)

B2C Accelerator AddOns Module

The B2C Accelerator AddOns module is a set of special extensions, called AddOns, that allow you to extend the functionality of your storefront without having to edit the core code base.

Features	Architecture	Implementation
		
Business Events in the Commerce Accelerator Google Analytics	captchaaddon AddOn hybrisAnalytics AddOn wishlist Extension	Updating captchaaddon to Support reCAPTCHA v2.0 Using Wishlist2Service API

B2C Accelerator AddOns Features

The B2C Accelerator AddOns module provides a number of AddOn extensions that allow you to add functionality to your storefront, such as business event tracking, monitoring web traffic with Google Analytics, and optimizing your storefront for mobile devices.

[Business Events in the Commerce Accelerator](#)

The Commerce Accelerator sends business events for a variety of user actions in the storefront. The `hybrisAnalytics` AddOn, which contains the Piwik scripts that are used to track user events, must be installed in the storefront in order to track user actions and send business events to the event system that collects the analytics data.

[Google Analytics](#)

This document discusses what is added to the SAP Commerce Accelerator code to enable the monitoring. Using this information, you can also remove Google Analytics from your system, if necessary.

Business Events in the Commerce Accelerator

The Commerce Accelerator sends business events for a variety of user actions in the storefront. The `hybrisAnalytics` AddOn, which contains the Piwik scripts that are used to track user events, must be installed in the storefront in order to track user actions and send business events to the event system that collects the analytics data.

Prerequisites

In order to capture and analyze business events in the Commerce Accelerator, the following requirements must be met:

- Accelerator must be installed and running a storefront.
- the `hybrisAnalytics` AddOn must be installed with the Accelerator storefront. For more information, see [hybrisAnalytics AddOn](#).
- the `eventtrackingws` AddOn must be installed with the Accelerator storefront. For more information, see [eventtrackingws AddOn](#).
- the Business Event extensions must be installed. For more information, see [hybris Business Event Extensions - Technical Guide](#).

View Page Events

There are several View Page events captured by the Piwik script when the `hybrisAnalytics` AddOn is installed in the Accelerator storefront. View Page events are generally sent to the event system whenever a user views any page. Each View Page event includes different attributes, depending on the type of page that is viewed by the user. For example, when a user views a Product Page, there are Product Page-specific events that are captured so that you have access to this information in the analytics system.

Commerce Accelerator generally sets the `pageType` model attribute in its controllers so that we can check the page type and send the appropriate View Page event.

Generic Page View Events

If any of the View Page events do not fall under a specific category, or do not require additional attributes to be captured, then we use the `trackPageView` method from the Piwik JS client.

Product Page View Events

Product Page View events are sent whenever a Product Page is viewed. Piwik provides the `setEcommerceView` method to set the product-related details. This method is used to capture all the product-related attributes from the model attributes set in the `ProductPageController` Java class. Once `setEcommerceView` is pushed, we call `trackPageView` with the appropriate `ViewPage` name (`ViewProduct`), according to the Piwik recommendation.

Category Page View Event

The Category Page View event is sent when a user navigates to a product category from the navigation bar or through the breadcrumb. In the Accelerator, the Category Page is treated like a Search Result page. For this reason, the Category Page View event is considered a Search event, because we get the product results for the category from the SOLR/indexing engine. In this case, we call the `trackSiteSearch` method that has the required parameters. Some custom data is also added using the `setCustomData` method. In this custom data we send the `categoryName`.

Search Page Events

Search Page events track the usage of the Search Box component in the Accelerator. The Search Box component is a WCMS component that allows users to input a search term, and also provides auto-search results when users enter a search term. The Search event is sent when a user enters a search term, clicks the search button, and gets back a search result that is non-empty. In this case, the information that is sent are the search term and the number of results. Here, we also call the `trackSiteSearch` method to send search-related information.

No Search Result Event

The No Search Result event is similar to the Search Page event, except that it is sent when there are no search results.

Banner Click Event

A Banner Click event is sent when any of the banners in the Accelerator site are clicked by a user. Banners in the Accelerator are WCMS components that have their own view (JSP). To capture the banner click, we need to make sure that the `simple-banner` class element is present in the component view, because the jquery selection is based on this class being present.

In this case, we use the `trackLink` method provided by Piwik. Also, the `setCustomVariable` method is called to pass the banner ID, which is the URL the banner is pointing to.

ProceedToCheckout Event

12/8/2020

The `ProceedToCheckout` event is sent when a user decides to check out the item in their cart page. In this case, the `Checkout` button in the cart page is attached to a click event in the `hybrisAnalytics` AddOn. Piwik provides the `trackEvent` method to track certain events. In terms of parameters, the endpoint expects `checkout` as a category and `proceed` as an action when calling the `trackEvent` method. The endpoint also expects `cartCode` or `cartId` as one of the custom variables included with this event.

Successful CheckoutEvent

The Successful Checkout event is sent when a user has placed an order and then views the Order Confirmation page.

The order confirmation sends the following requests : `pageview`, `trackevent`, and `ecommerce`.

Cart Modification Events

The Accelerator already provides some events that you can subscribe to regarding cart modifications. The `hybrisAnalytics` AddOn subscribes to these events and then sends the proper information through Piwik when any of these cart modifications is triggered.

AddToCart Event

The `AddtoCart` event is sent when a user clicks the `Add to Cart` button to add an item to their shopping cart. The Accelerator has `Add To Cart` buttons on various pages, including the Cart Page, the Product Details Page, and the Category Page, as well as in the Suggestions component. Whenever a user clicks the `Add to Cart` button, JS captures the button click and a back-end call is made to the server. In return, we receive the `CartModificationData` response to display the updated information. This data is then used to populate the Piwik event, and the `trackEcommerceCartUpdate` method is used.

UpdateCart Event

The `UpdateCart` event in the Accelerator is sent when a user updates the quantity of a particular order entry on the Cart Page. This event is identical to the `AddtoCart` event. The `UpdateCart` event is captured from JavaScript where click events are bonded to a specific button.

RemoveCart Event

The `RemoveCart` event in the Accelerator is sent when a user removes a cart item on the Cart Page. This event is identical to the `Add toCart` and `UpdateCart` events except that it sends a quantity of 0.

Google Analytics

This document discusses what is added to the SAP Commerce Accelerator code to enable the monitoring. Using this information, you can also remove Google Analytics from your system, if necessary.

You can monitor detailed statistics about the visitors to your website using the Google Analytics service.

i Note

Monitoring of statistics requires signing in to Google services, and some of the features that are offered must be purchased before they can be used.

Google Analytics Configuration

Google Analytics-specific properties are added to the `yacceleratorstorefront/project.properties` and `yb2bacceleratorstorefront/project.properties` files. In the `yacceleratorstorefront`, these properties are loaded into the page model in the `AnalyticsPropertiesBeforeViewHandler`. In the `yb2bacceleratorstorefront`, these properties are loaded into the page model in the `AbstractPageController`.

It is necessary to add definitions of the storefronts, which are monitored, to the relevant `project.properties` files of your storefronts. The reference version of the SAP Commerce Accelerator provides several storefronts, which are defined in the `yacceleratorstorefront` and `yb2bacceleratorstorefront` extensions. The following are examples:

The following example is from the `yacceleratorstorefront/project.properties` file:

https://help.sap.com/http.svc/dynamicpdfcontentpreview?deliverable_id=21802332&topics=8adca7a186691014bd31f1d2d9... 33/48

```
google.analytics.tracking.id
google.analytics.tracking.id.electronics.local
google.analytics.tracking.id.apparel-uk.local
google.analytics.tracking.id.apparel-de.local
```

The following example is from the `yb2bacceleratorstorefront/project.properties` file:

```
google.analytics.tracking.id.powertools.local
```

In order to have all the Google Analytics-specific JavaScript in one tag file, the controllers pass a **PageType** to views where specific tracking is required. In order to track certain calls to Ajax, calls to JavaScript functions defined in the `web/webroot/WEB-INF/tags/analytics/googleAnalytics.tag/googleAnalytics.tag` file have been added in the `addToCart.tag` and `cartItems.tag` files. These functions are only called if they have been defined, so removing all the Google Analytics JavaScript does not cause an error.

Creating a Google Analytics Account

For information on setting up a Google Analytics account, see [Google Analytics - Installation Guide](#).

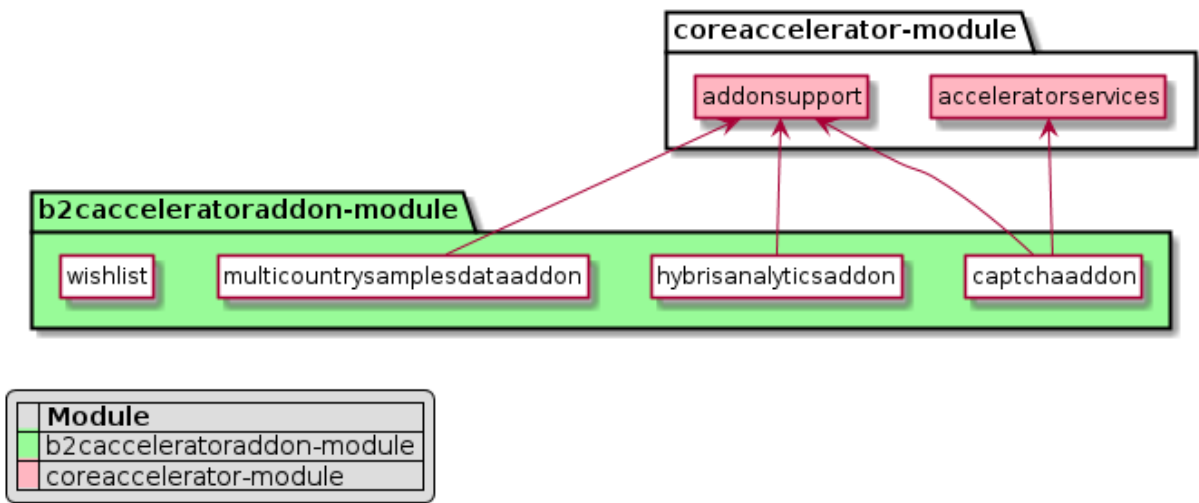
Related Information

[Google Analytics - Installation Guide](#)

B2C Accelerator AddOns Architecture

The B2C Accelerator AddOns module is a set of extensions that allow you to extend the functionality of B2C Accelerator without having to edit the core code base. For example, you can add a reCAPTCHA widget to your storefront with the `captchaaddon` AddOn, or integrate business event tracking functionality with `hybrisanalyticsaddon`.

Dependencies



Dependencies Diagram

Recipes

The following recipes contain the B2C Accelerator AddOns module:

- `b2c_acc_plus`
- `b2c_acc_ymkt`

For further information, see [Installer Recipes](#).

Extensions

The B2C Accelerator AddOns module consists of the following extensions:

- [captchaaddon AddOn](#)
- [hybrisAnalytics AddOn](#)
- multicountrysampledadataaddon AddOn
- [wishlist Extension](#)

captchaaddon AddOn

The captchaaddon AddOn introduces a reCAPTCHA widget to the **New Customer Registration** form of the Accelerator storefronts. The technology is used to block spammers and bots that try to automatically harvest e-mail addresses, or that try to automatically sign up for (or make use of) websites, blogs, or forums.

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.

AddOn Definition

Name	captchaaddon
Description	CAPTCHA is a program that protects websites against bots by generating and grading tests that human users can pass but current computer programs cannot. The captchaaddon AddOn adds a reCAPTCHA widget to the registration form in the Accelerator storefronts using a free reCAPTCHA service from Google.
Requires	The captchaaddon AddOn depends on: <ul style="list-style-type: none">• addonsupport Extension
Author	SAP Commerce

Spring and JavaScript Configuration

The captchaaddon extension uses Spring AOP in order to intercept the controller methods used for registration-form rendering and submission. It also checks the challenge answer from the reCAPTCHA widget, which is rendered using JavaScript.

The aspect is configured using Spring in the `/hybris/bin/modules/b2c-accelerator-addons/captchaaddon/resources/captchaaddon/web/spring/captchaaddon-web-spring.xml` file.

JavaScript is used to inject the reCAPTCHA widget into the registration form in the storefront. The `captchaaddon.js` file that contains the code `/hybris/bin/modules/b2c-accelerator-addons/captchaaddon/acceleratoraddon/web/webroot/_ui/responsive/common/js/captchaaddon.js`.

All CSS properties used by the captchaaddon AddOn are located in `/hybris/bin/modules/b2c-accelerator-addons/captchaaddon/acceleratoraddon/web/webroot/WEB-INF/_ui-src/responsive/less/captchaaddon.less`.

Enabling or Disabling CAPTCHA

You can enable or disable the CAPTCHA feature in the Backoffice Administration Cockpit.

1. Log into the Backoffice and navigate to **Base Commerce Base Store**.
2. Click the **Search** button and select a storefront.
3. In the **Properties** tab, select the **True** or **False** radio buttons under **Captcha Widget Enabled** to enable or disable the CAPTCHA widget.

Installing the captchaaddon AddOn

i Note

The captchaaddon AddOn is disabled by default. You must enable it after you install it.

1. Add the captchaaddon AddOn to your localextensions.xml file, ensuring the listed required extensions are also included.

```
<extension dir="${HYBRIS_BIN_DIR}/ext-addon/captchaaddon"/>
<extension dir="${HYBRIS_BIN_DIR}/ext-addon/addonsupport"/>
```

2. Call the addoninstall ant command.

```
ant addoninstall -Daddonnames="captchaaddon" -DaddonStorefront.yacceleratorstorefront="yacceleratorstorefront"
```

This generates the correct properties in the project.properties file of the captchaaddon AddOn and adds a dependency from your storefront to the captchaaddon AddOn.

3. Register your domain, which will allow you to obtain a reCAPTCHA key. You can do this at <https://www.google.com/recaptcha/intro/index.html> . When you receive your public and private keys, add them to the local.properties file.

```
recaptcha.publickey=myGeneratedSiteKey
recaptcha.privatekey=myGeneratedSecretKey
```

If you wish to have store-specific keys, you can add your store name to the properties. For the following example, change electronics to the name of your storefront.

```
recaptcha.publickey.electronics=myGeneratedSiteKey
recaptcha.privatekey.electronics=myGeneratedSecretKey
```

4. Add the Spring configuration property of the captchaaddon AddOn to the project.properties file.

```
yacceleratorstorefront.additionalWebSpringConfigs.captchaaddon=classpath:/captchaaddon/web/spring/captchaaddon
```

i Note

If you initialized your system before installing the captchaaddon AddOn, you have to run ant clean all and update the system.

Modifications Checklist

The modifications that this AddOn makes to Accelerator are listed below:

Impex Configuration Scripts	[-]
Core Data Listeners	[-]
Model Layer	[-]
Model Interceptors	[-]
Cockpit Configuration	[-]
Cockpit Beans	[-]
Validation Rules	[-]

Service Layer	[-]
Facade DTO	[-]
Facade Layer	[-]
CMS Components	[-]
Page Templates	[-]
JavaScript	[+]
CSS	[+]

Page Controllers	✓
Tags	✓
TLD	✓
Filters	✗
MVC Interceptors	✗
Spring Security	✗
Message Resources	✓

Related Information

[yacceleratorstorefront Extension](#)

<http://www.google.com/recaptcha>

<https://www.google.com/recaptcha/admin/create>

hybrisAnalytics AddOn

The purpose of the `hybrisanalyticsaddon` AddOn is to provide an integration point between the SAP Commerce Accelerator and the `eventtrackingws` AddOn.

The `hybrisanalyticsaddon` AddOn does this by sending events through JavaScript to the `eventtrackingws` AddOn. The events consist of user actions in the storefront, such as viewing a page, or adding an item to the cart. The `hybrisanalyticsaddon` AddOn provides the tag files and necessary JavaScripts that can be installed in the storefront.

For more information, see [eventtrackingws AddOn](#), [Business Events in the Commerce Accelerator](#), and [Event Tracking Module](#).

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.

AddOn Definition

Name	hybrisanalyticsaddon
Description	The purpose of the <code>hybrisanalyticsaddon</code> AddOn is to provide an integration point between the Accelerator and the <code>eventtrackingws</code> AddOn by sending events through JavaScript to this endpoint.
Requires	Business Events extensions
Author	SAP Commerce

Supported Markets and Channels

The `hybrisanalyticsaddon` AddOn supports the B2C and B2B markets. It can only be used for the desktop channel.

Supported	B2C Commerce	B2B Commerce	Telco Commerce
Market	✓	✓	✗
Channel	Desktop: ✓	Desktop: ✓	Desktop: ✗
	Mobile: ✗	Mobile: ✗	Mobile: ✗

JS and Tag Components

12/8/2020

The `hybrisanalyticsaddon` AddOn uses a Piwik library to send asynchronous JavaScript events to the `eventtrackingws` AddOn. The `piwikAnalytics.tag` file contains the JavaScript code that calls the Piwik JavaScript, which then sends the events to the `eventtrackingws` AddOn.

The `base.js.properties` file can be used to install key-value properties as JavaScript variables in the storefronts. The `base.js.properties` file is located in `/hybrisSrc/acceleratoraddons/hybrisanalyticsaddon/acceleratoraddon/web/webroot/WEB-INF/messages/`.

For more information, see [Business Events in the Commerce Accelerator](#).

The HybrisAnalyticsBeforeViewHandler

The `HybrisAnalyticsBeforeViewHandler` gets executed every time before the view is rendered. The `HybrisAnalyticsBeforeViewHandler` is responsible for reading the `base.js.properties` file in the AddOn, and sets these key-values as model attributes.

The `HybrisAnalyticsBeforeViewHandler` also calls the `SitesConfigService` to set the attributes in the model object, such as the `piwik site id` and other attributes.

Installation

The following procedure describes how to install the `hybrisanalyticsaddon` AddOn.

1. Add the `hybrisanalyticsaddon` AddOn to your `localextensions.xml` file, along with any required extensions, as follows:

```
<extension dir="${HYBRIS_BIN_DIR}/ext-addon/hybrisanalyticsaddon"/>
<extension dir="${HYBRIS_BIN_DIR}/ext-addon/addonsupport"/>
```

2. Install the AddOn by executing the `addoninstall` ant command.

```
ant addoninstall -Daddonnames="hybrisanalyticsaddon" -DaddonStorefront.yacceleratorstorefront="yaccelerator
```

This generates the correct properties in the `project.properties` file of the `hybrisanalyticsaddon` AddOn, and adds a dependency from your storefront to the `hybrisanalyticsaddon` AddOn.

The `yacceleratorstorefront`, or the storefront generated from `yacceleratorstorefront`, has CSRF filters that expect a CSRF token during the POST request. To ensure that the Ajax POST request from the Piwik JavaScript doesn't get filtered by the CSRF filter, add the events servlet path in the list of URLs where the CSRF is allowed to ignore POST requests.

The following is an example from the `project.properties` of your storefront:

```
csrf.allowed.url.patterns=/[/^/]+(/[^?]*)+(sop/response)$, /[/^/]+(/[^?]*)+(merchant_callback)$, /[/^/]+(/[^?]*)+(hop,
```

In this example, `/(events)$` is added to this property to ensure that the CSRF filter allows POST requests without the CSRF tokens.

Configuring the Site IDs

Piwik expects site IDs to be included with each event. These site IDs must be integers and they're configurable. You can have a site ID for your entire SAP Commerce setup, or you can configure them per site. This is done through the `local.properties` file.

The following is the global site ID:

```
piwik.tracker.siteid=<your-site-id>
```

The following is the site ID per site :

```
piwik.tracker.siteid.<storeuid>=<your-site-id>
```

For example:

```
piwik.tracker.siteid.electronics=123456789
```

Configuring the Tracking URLs

You can define the URLs to send the events to. You can define two different URLs that depend on which channel you're sending the events on (for example, HTTP and HTTPS). By default, the AddOn sends the events to the localhost.

```
piwik.tracker.url=http://electronics.local:9002/yacceleratorstorefront/events
piwik.tracker.https.url=https://electronics.local:9002/yacceleratorstorefront/events
```

Integration with the Endpoint

For information on installing the event tracking endpoint, see [eventtrackingws AddOn](#).

Verifying the Integration

The following procedure describes how to verify the integration of the `hybrisanalyticsaddon` AddOn.

1. Start the SAP Commerce server.
2. Access the Accelerator storefront home page.

The following link is an example: `http://electronics.local:9001/yacceleratorstorefront`
3. Browse to a product page to trigger event tracking.
4. Using the Chrome browser, right-click on a page and select `Inspect Element`.
5. In the window that appears, select the `Network` header tab and find the `POST` method with the following URL:
`/yacceleratorstorefront/events`.

By adding two properties, you can enable the logging of the events, as follows:

```
log4j.logger.de.hybris.eventtracking.publisher=DEBUG
spring.profiles.active=eventtrackingpublisher_develop
```

Note

These properties aren't specific to the `hybrisanalyticsaddon` AddOn but to the `eventtrackingws` AddOn.






6. The end-point URL is HTTPS, and for a development environment the browser may block the Ajax calls due to an invalid SSL certificate. If this is the case, open the browser, type in the endpoint URL `https://electronics.local:9002/events` and force the browser to allow this HTTPS URL even though the certificate may not be valid.








Modifications Checklist

The following table lists the modifications that the AddOn makes to the Accelerator:

Impex Configuration Scripts	[-]
Core Data Listeners	[-]
Model Layer	[-]
Model Interceptors	[-]
Cockpit Configuration	[-]
Cockpit Beans	[-]
Validation Rules	[-]

Service Layer	[-]
Facade DTO	[-]

Facade Layer	
CMS Components	
Page Templates	
JavaScript	
CSS	

Page Controllers	
Tags	
TLD	
Filters	
MVC Interceptors	
Spring Security	
Message Resources	

wishlist Extension

The `wishlist` extension provides functionality for listing and maintaining items a customer would like to have, for example products intended for buying or desired as gifts.

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.

Related Information

[Using Wishlist2Service API](#)

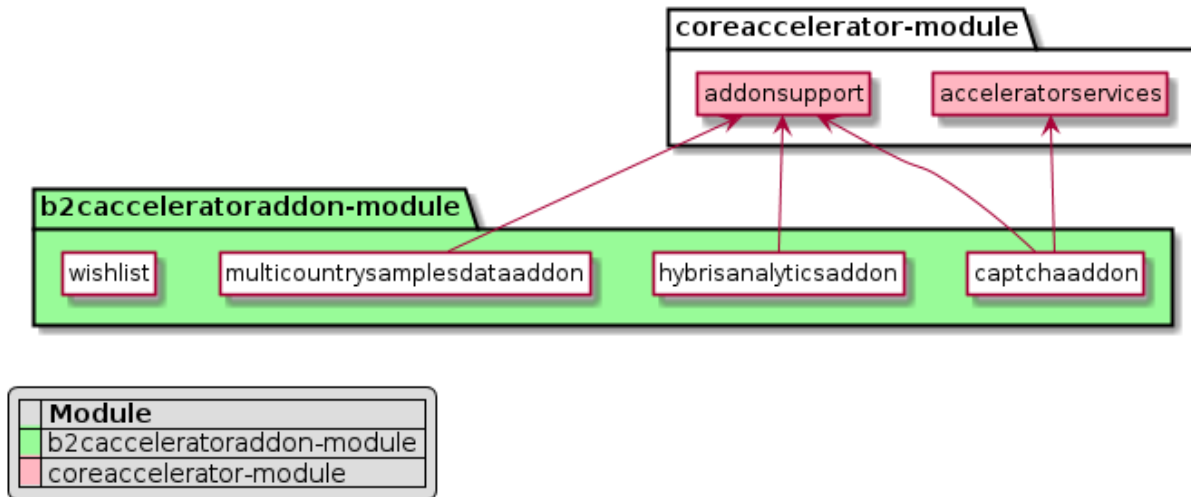
multicountrysampledadataaddon AddOn

The `multicountrysampledadataaddon` is the AddOn that is responsible for creating sample data for the electronics storefront to demonstrate the multi-country features in the Accelerator web application. The AddOn creates additional electronics sites to simulate parent-child catalog hierarchies.

About the Extension

Name	Directory	Related Module
multicountrysampledadataaddon	hybris/bin/modules/b2c-accerlator-addons	B2C Accelerator AddOns Architecture

Dependencies



Dependencies Diagram

B2C Accelerator AddOns Implementation

You can extend your B2C Accelerator storefront with wishlist functionality, add a reCAPTCHA widget, and optimize your storefront for mobile devices.

[Updating captchaaddon to Support reCAPTCHA v2.0](#)

Google no longer provides new keys for reCAPTCHA v1.0. Existing implementations of the captchaaddon AddOn will continue to work, but new implementations must use reCAPTCHA v2.0. The captchaaddon AddOn can be modified to support reCAPTCHA v2.0.

[Using Wishlist2Service API](#)

The wishlist extension is based on the ServiceLayer and is service-oriented. Use the Wishlist2Service API to manage wishlist.

Updating captchaaddon to Support reCAPTCHA v2.0

Google no longer provides new keys for reCAPTCHA v1.0. Existing implementations of the captchaaddon AddOn will continue to work, but new implementations must use reCAPTCHA v2.0. The captchaaddon AddOn can be modified to support reCAPTCHA v2.0.

Context

Google's reCAPTCHA v2.0 no longer supports public and private keys. Instead, reCAPTCHA v2.0 uses a site key instead of a public key, and a secret key instead of a private key. These keys are used to render the captcha widget and validate the response.

Procedure

1. Open the following URL: <https://www.google.com/recaptcha/intro/index.html>
2. Register your website and retrieve a new site key and secret key.
3. Update the `config/local.properties` file to include your two new keys, as follows:

```

...
recaptcha.publickey=YOUR SITE KEY
recaptcha.privatekey=YOUR SECRET KEY
...

```

4. Update `captchaaddon/acceleratoraddon/web/webroot/WEB-INF/views/responsive/pages/widget/recaptcha.jsp` to include an element with the appropriate reCAPTCHA class and site key, as follows:

```

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>
<%@ taglib prefix="theme" tagdir="/WEB-INF/tags/shared/theme" %>
<%@ taglib prefix="ycommerce" uri="http://hybris.com/tld/ycommercetags" %>

<c:if test="${requestScope.captchaEnabledForCurrentStore}">
  <div id="g-recaptcha_incorrect"><spring:theme code="recaptcha.challenge.field.invalid"/></div>

```

```
<div id="g-recaptcha_widget" class="g-recaptcha" data-sitekey="${requestScope.recaptchaPublicKey}"></div>
</c:if>
```

The `g-recaptcha` class indicates which `div` holds the reCAPTCHA widget. The `data-sitekey` is used for validating to the reCAPTCHA widget. In the preceding example, there is an additional `div` to show user validation in cases where a user tries to register without completing the captcha.

5. You may want to update the CSS. The following is an example from `captchaaddon/acceleratoraddon/web/webroot/WEB-INF/_ui-src/responsive/less/captchaaddon.less`.

```
#registerForm .form_field_error {
    width:auto;
    float: none;
    padding-right:0;
    clear:both;
}

.js-recaptcha-captchaaddon {
    margin: 20px 0;
}

#g-recaptcha_incorrect {
    color: red;
    display: none;
}
```

6. Update `captchaaddon/acceleratoraddon/web/webroot/_ui/responsive/common/js/captchaaddon.js` as follows:

```
/*
 * [y] hybris Platform
 *
 * Copyright (c) 2000-2016 SAP SE or an SAP affiliate company.
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of SAP
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with SAP.
 */
ACC.captcha = {
    bindAll: function ()
    {
        this.renderWidget();
    },

    renderWidget: function ()
    {
        $.ajax({
            url: ACC.config.encodedContextPath + "/register/captcha/widget/recaptcha",
            type: 'GET',
            cache: false,
            success: function (html)
            {
                if ($(html) != [])
                {
                    $(html).appendTo('.js-recaptcha-captchaaddon');
                    $.getScript('https://www.google.com/recaptcha/api.js?hl=' + document.documentElement.lang, function () {
                        if ($('#recaptchaChallengeAnswered').val() == 'false')
                        {
                            $('#g-recaptcha_incorrect').show();
                        }
                    });
                }
            }
        });
    }
};

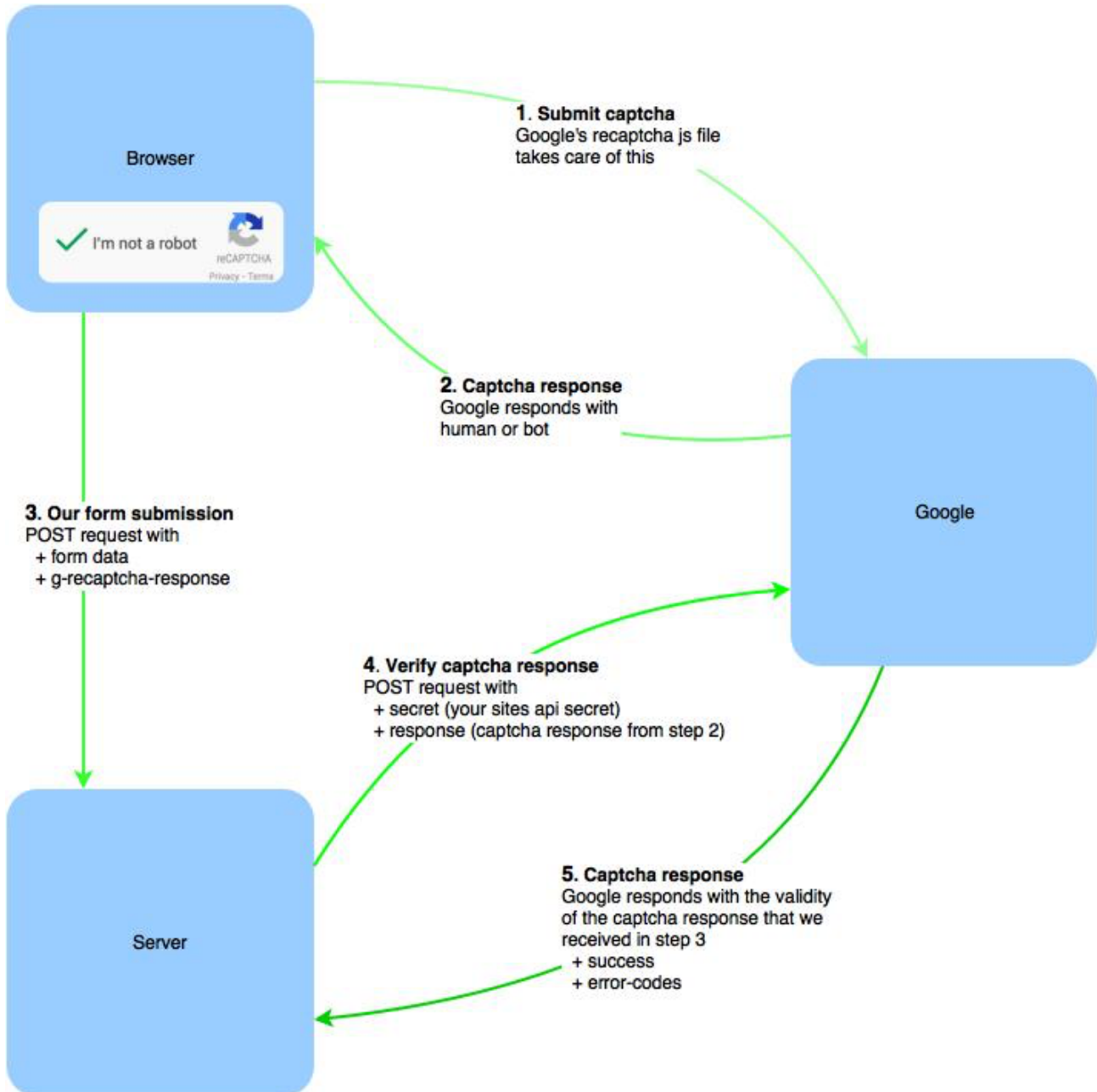
$(document).ready(function ()
{
    if ($('#registerForm').html() != null || $('#updateEmailForm').html() != null)
    {
        ACC.captcha.bindAll();
    }
});
```

The `captchaaddon.js` file renders the widget by loading Google's JavaScript API with the necessary parameters. For more information on displaying the widget, see the following Google documentation: <https://developers.google.com/recaptcha/docs/display>

7. Update the backend to validate the captcha response.

When the user completes the captcha and submits the form, a captcha response is returned. You can validate this response, using Google's API, by submitting a POST request to `https://www.google.com/recaptcha/api/siteverify` with the captcha response and secret key as parameters. This returns a JSON object with a boolean under **success** that indicates whether the captcha is valid or not.

The following diagram illustrates the process:



8. Update `captchaaddon/acceleratoraddon/web/src/de/hybris/platform/security/captcha/ReCaptchaAspect.java`. The following is an example:

```

/*
 * [y] hybris Platform
 *
 * Copyright (c) 2000-2016 SAP SE or an SAP affiliate company.
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of SAP
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with SAP.
 */

```

```

package de.hybris.platform.security.captcha;

import atg.taglib.json.util.JSONException;
import atg.taglib.json.util.JSONObject;
import de.hybris.platform.acceleratorservices.config.SiteConfigService;
import de.hybris.platform.store.BaseStoreModel;
import de.hybris.platform.store.services.BaseStoreService;

import java.io.IOException;
import java.util.Arrays;
import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.collections.PredicateUtils;
import org.apache.commons.httpclient.DefaultHttpClientRetryHandler;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.HttpStatus;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.aspectj.lang.ProceedingJoinPoint;
import org.springframework.beans.factory.annotation.Required;
import org.springframework.validation.BindingResult;
import org.springframework.web.context.request.RequestContextHolder;
import org.springframework.web.context.request.ServletRequestAttributes;

/**
 * An aspect which uses google ReCaptcha api to validate captcha answer on the storefront Registration form
 */

public class ReCaptchaAspect
{
    private static final Logger LOG = Logger.getLogger(ReCaptchaAspect.class);

    private static final String RECAPTCHA_SITE_KEY_PROPERTY = "recaptcha.publickey";
    private static final String RECAPTCHA_SECRET_KEY_PROPERTY = "recaptcha.privatekey";
    private static final String RECAPTCHA_RESPONSE_PARAM = "g-recaptcha-response";
    private static final String RECAPTCHA_VERIFY_URL = "https://www.google.com/recaptcha/api/siteverify";

    private SiteConfigService siteConfigService;
    private BaseStoreService baseStoreService;

    public Object prepare(final ProceedingJoinPoint joinPoint) throws Throwable
    {
        final List<Object> args = Arrays.asList(joinPoint.getArgs());
        final HttpServletRequest request = (HttpServletRequest) CollectionUtils.find(args,
            PredicateUtils.instanceOfPredicate(HttpServletRequest.class));

        if (request != null)
        {
            final boolean captchaEnabledForCurrentStore = isCaptchaEnabledForCurrentStore();
            request.setAttribute("captchaEnabledForCurrentStore", Boolean.valueOf(captchaEnabledForCurrentStore));
            if (captchaEnabledForCurrentStore)
            {
                request.setAttribute("recaptchaPublicKey", getSiteConfigService().getProperty(RECAPTCHA_SITE_KEY_PROPERTY));
            }
        }
        return joinPoint.proceed();
    }

    public Object advise(final ProceedingJoinPoint joinPoint) throws Throwable
    {
        final boolean captchaEnabledForCurrentStore = isCaptchaEnabledForCurrentStore();
        if (captchaEnabledForCurrentStore)
        {
            final List<Object> args = Arrays.asList(joinPoint.getArgs());
            HttpServletRequest request = (HttpServletRequest) CollectionUtils.find(args,
                PredicateUtils.instanceOfPredicate(HttpServletRequest.class));

            if (request == null && RequestContextHolder.getRequestAttributes() instanceof ServletRequestAttributes)
            {
                final ServletRequestAttributes requestAttributes = (ServletRequestAttributes) RequestContextHolder.getRequestAttributes();
                request = requestAttributes.getRequest();
            }

            if (request != null)

```

```

    {
        request.setAttribute("captchaEnabledForCurrentStore", Boolean.valueOf(captchaEnabledForCurrentStore));
        request.setAttribute("recaptchaPublicKey", getSiteConfigService().getProperty(RECAPTCHA_SITE_KEY));
        final String recaptchaResponse = request.getParameter(RECAPTCHA_RESPONSE_PARAM);
        if (StringUtils.isBlank(recaptchaResponse) || !checkAnswer(recaptchaResponse))
        {
            // if there is an error add a message to binding result.
            final BindingResult bindingResult = (BindingResult) CollectionUtils.find(args,
                PredicateUtils.instanceOfPredicate(BindingResult.class));
            if (bindingResult != null)
            {
                bindingResult.reject("recaptcha.challenge.field.invalid", "Challenge Answer is invalid.");
            }
            request.setAttribute("recaptchaChallengeAnswered", Boolean.FALSE);
        }
    }
}
return joinPoint.proceed();
}

protected boolean checkAnswer(final String recaptchaResponse)
{
    final HttpClient client = new HttpClient();
    final PostMethod method = new PostMethod(RECAPTCHA_VERIFY_URL);

    method.getParams().setParameter(HttpMethodParams.RETRY_HANDLER, new DefaultHttpClientRetryHandler(3));
    method.addParameter("secret", getSiteConfigService().getProperty(RECAPTCHA_SECRET_KEY_PROPERTY));
    method.addParameter("response", recaptchaResponse);

    try
    {
        final int statusCode = client.executeMethod(method);

        if (statusCode != HttpStatus.SC_OK)
        {
            return false;
        }

        final JSONObject response = new JSONObject(method.getResponseBodyAsString());
        return response.getBoolean("success");
    }
    catch (IOException | JSONException e)
    {
        LOG.error("Exception occurred while checking captcha answer", e);
        return false;
    }
    finally
    {
        method.releaseConnection();
    }
}

protected boolean isCaptchaEnabledForCurrentStore()
{
    final BaseStoreModel currentBaseStore = getBaseStoreService().getCurrentBaseStore();
    return currentBaseStore != null && Boolean.TRUE.equals(currentBaseStore.getCaptchaCheckEnabled());
}

protected SiteConfigService getSiteConfigService()
{
    return siteConfigService;
}

@Required
public void setSiteConfigService(final SiteConfigService siteConfigService)
{
    this.siteConfigService = siteConfigService;
}

protected BaseStoreService getBaseStoreService()
{
    return baseStoreService;
}

@Required
public void setBaseStoreService(final BaseStoreService baseStoreService)
{
    this.baseStoreService = baseStoreService;
}
}

```

The `json-taglib` has also been added to the `captchaaddon` AddOn. The `prepare()` method passes along the `sitekey` required to render the widget. The `advise()` method is called when the user submits the registration form and uses the `checkAnswer()` method to validate the user's response. The remaining methods are unchanged.

In the above example, a new library has also been added to parse the JSON object response.

9. Add the `json-taglib-0.4.1` library to the `captchaaddon` AddOn by adding the following lines to `captchaaddon/acceleratoraddon/web/webroot/WEB-INF/external-dependencies.xml`:

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    ...
    <dependencies>
        <dependency>
            <groupId>de.hybris.external</groupId>
            <artifactId>json-taglib</artifactId>
            <version>0.4.1</version>
        </dependency>
    </dependencies>
</project>
```

This adds `captchaaddon/acceleratoraddon/web/webroot/WEB-INF/lib/json-taglib-0.4.1.jar` when you build your project by running `ant clean all`. Add this jar as a dependency to the `captchaaddon` AddOn. You can do this in Eclipse by right-clicking on **Project Build Path Configure Build Path**, and under the **Libraries** tab, clicking **Add Jars** and selecting the jar.

Using Wishlist2Service API

The `wishlist` extension is based on the `ServiceLayer` and is service-oriented. Use the `Wishlist2Service` API to manage wishlist.

About Wishlists

The `wishlist` extension provides the possibility for the user to organize and keep track of the desired products. One user can have as many wishlists as needed, and the wishlist can contain as many entries as possible. However, in each wishlist the same product can only occur once. Duplicated products are not allowed in the same wishlist.



Using the Wishlist2Service API

The `de.hybris.platform.wishlist2.Wishlist2Service` provides the necessary methods to manage wishlists. You can refer to the service within your `spring.xml` file.

```
<bean id="yourService" class="yourServiceClass">
    <property name="wishlistService" ref="wishlistService"/>
</bean>
```

Alternatively, you can accomplish that by using Java code as in the code sample below.

```
Wishlist2Service wsService = (Wishlist2Service) Registry.getApplicationContext().getBean("wishlistService");
```

Creating a Wishlist

In order to create the default wishlist for a user, call the method as shown in the following code snippet.

```
public Wishlist2Model createDefaultWishlist(UserModel user, String name, String description);
```

To create a normal wishlist, call the method as shown in following code snippet.

```
public Wishlist2Model createWishlist(UserModel user, String name, String description);
```

Adding a Wishlist Entry to the Default Wishlist

A wishlist entry can be added to the default wishlist of the specific user.

```
public void addWishlistEntry(final UserModel user, final ProductModel product,
                             final Integer desired, final Wishlist2EntryPriority priority,
                             final String comment);
```

i Note

Avoiding Duplicated Wishlist Entries

The same product should not be added twice in the same wishlist. To prevent duplication problems, proceed as follows: Before the wishlist entry is created and added to the active wishlist, iterate the wishlist and check whether there is already a wishlist entry with the specified product:

- If true, then give the user a message and return.
- If false, create the wishlist entry and add it to the current wishlist.

Retrieving Wishlists of the User

If you want to retrieve all wishlists of the specific user, call this method:

```
public List<Wishlist2Model> getWishlists(final UserModel user);
```

Similarly, you can retrieve the default wishlist.

```
public Wishlist2Model getDefaultWishlist(final UserModel user);
```

Removing a Wishlist Entry from the Wishlist

If one product is no longer needed, call the method from the following code sample to remove the corresponding wishlist entry from the wishlist.

```
public void removeWishlistEntry(final Wishlist2Model wishlist, final Wishlist2EntryModel entry);
```

12/8/2020

The entry being removed by that method is an object of the public `Wishlist2EntryModel` class that extends the `ItemModel`. It has several attributes, for example:

- `received`: The number of products that the user has received. Initial value: 0
- `desired`: The number of products that the user wants. Initial value: 1

If the users buys a product, then the `received` property can be set as 1 or to the amount of products that users has bought. The value of the `desired` property depends on users, because only the users know how many products they still want to have.

Commerce B2B Accelerator

Commerce B2B Accelerator is a ready-to-use Web framework that enables you to jump-start your B2B implementation and easily build and maintain a feature-rich, omni-channel commerce solution.

[B2B Accelerator Module](#)

B2B Accelerator includes features that manage your inventory, security, AddOns, and user accounts.

[B2B Accelerator AddOns Module](#)

B2B Accelerator AddOns allow you to manage your accounts, order forms, login security, and punchout procurements.

[B2B Commerce Module](#)

B2B Commerce Module adds business-to-business functionality to SAP Commerce. It enables you to integrate multiple channels, business models, and markets on a single platform.