



Extend Integration for Salesforce Commerce Cloud SiteGenesis

Welcome to Extend! We provide an easy way for any merchant to sell extended warranties - generating new revenue, increasing purchase conversion, and dramatically improving the customer experience.

This guide will walk you through the process of getting live with Extend Protection Plans, so you can start selling and earning revenue as quickly and easily as possible. The process is pretty simple - and even more simple with a connected SFCC SiteGenesis v19.10.0 store. Read on to learn more!

Official Extend Cartridge Link: <https://github.com/helloextend/salesforce-commerce-cloud>



Table of Contents

Extend Integration for Salesforce Commerce Cloud SiteGenesis	1
Table of Contents	2
Component Overview	3
Functional Overview	3
Use Cases	3
Compatibility	4
Privacy Policies on Payments	4
Implementation Guide	5
Setup of Business Manager	5
Upload cartridge & include in the cartridge path	5
Import metadata	5
Import custom jobs	5
Import service	5
Import Extend catalog	6
Configuration	6
Catalog setup	6
Custom preferences	6
Jobs	8
Extend Products Export	8
Extend Orders Creation	8
Extend Contracts Creation	9
Send Historical Orders	9
Create Refunds from SFCC	9
Custom Code	10
External Interfaces	23
Firewall Requirements	23
Orders API	24
Custom preferences	24
Extend SDK analytics	29
Custom Preferences	29
9. Locale restrictions.	33
10. Failover/Recovery process.	33





34

Testing

Enable Extend	34
PDP	34
Cart	36
Business Manager	38
ExtendOrderQueue custom object	38
ExtendContractsQueue custom object	41

Questions? Feedback? 43



3



Component Overview

1. Functional Overview

This integration cartridge enables the following features:

- Automated product catalog export to Extend
- Present offers via the Product Display Page (PDP), an up-sell modal during the add-to-cart action, and within the Cart page for eligible products
- Purchase Extend Protection Plans for eligible products
- Contract creation and cancellation with Extend via the API

2. Use Cases

For a merchant, the primary use cases include:

- Maintaining core Extend functionalities directly within Business Manager
- Display Extend Protection Plan products in PDP & Cart
- Display an Extend Protection Plan up-sell modal in PDP & Cart
- Capture customer behavior with Extend's analytics module
- Generate contracts for newly purchased protection plans with Extend
- Cancel contracts for returned orders with Extend

For a customer, the primary use cases include:

- Attach an Extend Protection Plan to an eligible catalog product during their purchase journey
- View details, including coverage and terms & conditions about the Extend Protection Plans
- Remove Extend Protection Plans from Cart (either by removing the main product or just the warranty product) during their purchase journey

3. Compatibility

This cartridge has been developed and tested against Commerce Cloud Digital 19.10 and integrated on top of SiteGenesis 104.1.5.

4. Privacy Policies on Payments

This cartridge implements a custom object for temporary storage of order data needed for Extend Protection Plan contract generation. This custom object (ExtendContractsQueue) was implemented based on a queue pattern that is consumed



by a job configured to execute every 5 minutes. This data is not exposed publicly and used only in a server-to-server communication setup.



Extend does not handle payments from the end-user. The merchant is responsible for charging the end-user for the Extend Protection Plans purchased, using the existing tenders that are currently supported on the specific environment. The merchant is responsible for generating Extend Protection Plan contracts only after charging the end-user for the warranty plans purchased. Please visit <https://extend.com/privacy> for more information.





Implementation Guide

The SiteGenesis version of this integration follows the best practices for this storefront version. In order to achieve the features described in this document, please use the sample code snippets below as reference for this integration.

1. Setup of Business Manager

1.1. Upload cartridge & include in the cartridge path

- i. Upload the int_extend cartridge to the sandbox.
- ii. Go to Business Manager → Administration → Site → Manage Sites. Select the desired site and go to Settings. At the beginning of the Cartridge Path add “int_extend:” in order to include the cartridge in the current site’s configuration.

1.2. Import metadata

- i. Go to Business Manager → Administration → Site Development → Import & Export
- ii. Upload the following files from ./metadata/meta/ folder
 - system-objecttype-extensions.xml
 - custom-objecttype-definitions.xml
- iii. Go to Business Manager → Administration → Site Development → Import & Export → Meta Data → Import. Both previous files should be available for import. Proceed to importing both.

1.3. Import custom jobs

- i. Go to Business Manager → Administration → Operations → Import & Export
- ii. Upload the following file: ./metadata/jobs.xml
- iii. Go to Business Manager → Administration → Operations → Import & Export → Jobs → Import. The previous file should be available for import. Proceed to importing it.

1.4. Import service

- i. Go to Business Manager → Administration → Operations → Import & Export
- ii. Upload the following file: ./metadata/services.xml
- iii. Go to Business Manager → Administration → Operations → Import & Export → Services → Import. The previous file should be available for import. Proceed to importing it.





- iv. Note: by default, the service URL is configured to point to the demo instance of the Extend API. Prior to going live with the integration, this will need to be changed and tested with the live production API.

1.5. Import Extend catalog

- i. Select the site integrating with Extend
- ii. Go to Business Manager → Merchant Tools → Products and Catalogs → Import & Export
- iii. Upload the following file: ./metadata/catalog.xml. This will import 3 dummy option products, which are used in order to create dynamic product line items in the cart. We recommend keeping this as a separate master catalog, for visibility purposes.

2. Configuration

2.1. Catalog setup

- i. The 3 Extend products that require configuration are:
 - EXTEND-12
 - EXTEND-24
 - EXTEND-36
- ii. To make the Extend products merchandisable, perform the following:
 - Assign to the storefront catalog
 - Allocate an inventory record in the inventory file assigned to the current site
 - Configure this entry as perpetual, as these are **intangible goods and need to always be in-stock**
- iii. These Extend products do not require any description, pricing, or any other regular product attributes to be configured

2.2. Custom preferences

- i. Go to Business Manager → Merchant Tools → Site Preferences → Custom Preferences → **Extend: Integration Preferences**
 - **Enable Extend:** Global switch for all things Extend related within the environment.
 - **Extend Store ID:** Merchant specific Store ID available from their Extend account. The Store ID is environment specific (Extend production or Extend sandbox), and is required to make the API connections with Extend.





- **Extend Environment in Production:** Defines if Extend APIs call the Extend Production environment (Yes = <https://api.helloextend.com/>) or Extend Demo environment (No/None = <https://api-demo.helloextend.com/>). Access Token must be updated to correspond to the appropriate environment. Note: this is available from the Extend Merchant Portal and is environment specific.
- **Extend Access Token:** Merchant specific API key available from their Extend account. This key is environment specific (Extend production or Extend sandbox), and is required to make the API connections with Extend.
- **Extend Store Name:** This is the name of the user's store. Used for calls to Extend API. Note: Extend Store Name is by default configured to "Online Store".
- **Extend Integration Method:** Determines to which Extend API order data is sent.
Orders API on Order Create triggers the integration with Extend near real-time upon the order creation in SFCC Business Manager.
Orders API on Schedule creates a custom queue record (ExtendOrderQueue) upon order creation in the SFCC Business Manager, but will not integrate with Extend until the Job is run manually or on a schedule. The Job will consume the custom queue.
Contracts API on Schedule is a legacy feature whereby only Contract creation occurs (requires Extend API Version 2021-04-01 or earlier). Integration with Contracts API uses a Job that consumes a custom queue (ExtendContractQueue) when run manually or on a schedule. Please confirm with the Extend delivery team before selecting this option.
- **Product Image Size:** Determines the size of the product image stored on the Extend catalog. Accepted values are "small", "medium", "large", and "hi-res". This is the image that Extend uses when sending warranty contract emails to customers. Image Size is by default configured to large.
- **Extend Refund Key:** Unique key defined by the merchant to be used as part of Extend's contract cancellation process in SFCC if applicable.

- ii. Go to Business Manager → Merchant Tools → Site Preferences → Custom Preferences → **Extend: Storefront Preferences**





- **Extend SDK URL:** Denotes the SDK version used on the Storefront specifically for the Extend PDP offer, Interstitial Modal offer, and Cart offer (individual settings below).
- **Enable in PDP Offers:** Allows merchants to enable or disable the Extend Product Detail Page (PDP) offer.
- **Enable Modal Offers:** Allows merchants to enable or disable the Extend Interstitial Modal offer.
- **Enable Cart Offers:** Allows merchants to enable or disable the Extend Cart offer.
- **Enable Analytics:** Invokes Extend's Analytics methods within the Extend SDK.

3. Jobs

3.1. Extend Products Export

This job uses Extend's API to traverse the storefront catalog and export each Simple and Variant Product that is currently merchandisable (online and in stock).

- i. Go to Business Manager → Operations → Jobs. Select the Extend Products Export job.
- ii. Go to Job Steps and select Scope. Configure this to run in the scope of the site integrating with Extend.
- iii. Schedule this to run at the preferred cadence (Extend recommends once per day), at a time that's most suitable for a recurring job setup, ideally after the catalog is being updated.
- iv. Ensure this job is triggered at least once without errors and notify Extend after the successful run. Extend will confirm if the product sync was successful.

3.2. Extend Orders Creation

This job uses Extend's Orders API to consume the orders queue, stored in the [ExtendOrdersQueue](#) custom object. During SFCC order creation (after checkout), an instance of this object is created for each SFCC order. When this job is triggered, it sends an Order generation request to Extend and Extend will generate any necessary service contracts or lead tokens (for Post Purchase flow) and write them back onto the corresponding order line items. Upon successful order creation, the job removes the completed instances from the queue. This job is only required if the **Extend Integration Method** custom preference is set to **Orders API on Schedule**.





3.3. Extend Contracts Creation

This job uses Extend's Contracts API to consume the contracts queue, stored in the **ExtendContractsQueue** custom object. During SFCC order creation (after checkout), an instance of this object is created for each Extend Protection Plan on the SFCC order. When this job is triggered, it sends a contract generation request to Extend for each queued element. Upon successful contract creation, the job removes the completed instances from the queue. This job is only required if the **Extend Integration Method** custom preference is set to **Contracts API on Schedule**.

3.4. Send Historical Orders

This job uses Extend's Orders API to send historical orders from the past 2 years in SFCC to Extend during extension/cartridge installation for analytics and optimization reasons.

Note: For new merchants only who install the SFCC extension for the first time.

3.5. Create Refunds from SFCC

This job uses Extend's API to cancel an active contract on a canceled SFCC order (Order Status = "Canceled"). Please note that Extend will never refund the customer directly, as Extend does not capture the customer's payment information at any point. Customer refunds should be processed based on the merchant's regular refund process. This job will simply terminate active contract(s) after the customer's refund request is accepted by the merchant. All canceled Extend contracts are represented as credit lines on Extend's monthly invoice. For additional information on Extend's refund policy, please reach out to the Extend project team.

Note: This job is not required if the merchant's return and refund process occurs outside of SFCC as the contract cancellation requests to Extend will likely occur in a downstream system. Confirm with the Extend project team prior to scheduling this job.





4. Custom Code

4.1. Include PDP functionality

- In: */app_storefront_core/cartridge/js/pages/product/variant.js
- Where: function updateContent(), at the end of the \$.ajax() callback function

Code:

```
if (window.EXT_GLOBAL_SWITCH) {  
    window.extendPDP();  
}
```

```
1  'use strict';  
2  
3  var ajax = require(' ../../ajax'),  
4      image = require('./image'),  
5      progress = require(' ../../progress'),  
6      productStoreInventory = require(' ../../storeinventory/product'),  
7      tooltip = require(' ../../tooltip'),  
8      util = require(' ../../util');  
9  
10 //**  
11 * @description update product content with new variant from href, load new content to #product-content panel  
12 * @param {String} href - url of the new product variant  
13 *  
14 **/  
15 var updateContent = function (href) {  
16     var $pdpForm = $('.pdpForm');  
17     var qty = $pdpForm.find('input[name="Quantity"]').first().val();  
18     var params = {  
19         Quantity: isNaN(qty) ? '1' : qty,  
20         format: 'ajax',  
21         productlistid: $pdpForm.find('input[name="productlistid"]').first().val()  
22     };  
23  
24     progress.show($('#pdpMain'));  
25  
26     ajax.load({  
27         url: util.appendParamsToUrl(href, params),  
28         target: $('#product-content'),  
29         callback: function () {  
30             if (SitePreferences.STORE_PICKUP) {  
31                 productStoreInventory.init();  
32             }  
33             image.replaceImages();  
34             tooltip.init();  
35             if (window.EXT_GLOBAL_SWITCH) {  
36                 window.extendPDP();  
37             }  
38         }  
39     });  
40 };
```





4.2. Include Upsell / cross-sell modal functionality

- In: */app_storefront_core/cartridge/js/pages/product/addToCart.js
- Where: function addToCart(), after the \$form variable declaration and before addItemToCart()

Code:

```
// Extend Integration
if ($('#extend-offer').length) {
    extendAddToCart($form, page, minicart, dialog, addCartItem);
    return;
}
```

```
35  /**
36   * @description Handler to handle the add to cart event
37   */
38 var addToCart = function (e) {
39     e.preventDefault();
40     var $form = $(this).closest('form');
41
42     // Extend Integration
43     if ($('#extend-offer').length) {
44         extendAddToCart($form, page, minicart, dialog, addCartItem);
45         return;
46     }
47
48     addCartItem($form).then(function (response) {
49         var $uuid = $form.find('input[name="uuid"]');
50         if ($uuid.length > 0 && $uuid.val().length > 0) {
51             page.refresh();
52         } else {
53             // do not close quickview if adding individual item that is part of product set
54             // @TODO should notify the user some other way that the add action has completed successfully
55             if (!$(this).hasClass('sub-product-item')) {
56                 dialog.close();
57             }
58             minicart.show(response);
59         }
60     }).bind(this));
61 };
62 
```





4.3. Import extendHelpers.js in CartModel.js

- In: */app_storefront_controllers/cartridge/scripts/models/CartModel.js
- Where: include this script at the API Includes section

Code:

```
// Import Extend Script Helpers
var extendHelpers = require('*/*cartridge/scripts/extend');
```

```
1  'use strict';
2  /**
3   * Model for cart functionality. Creates a CartModel class with payment, shipping, and product
4   * helper methods.
5   * @module models/CartModel
6   */
7  var Transaction = ...require('dw/system/Transaction');
8
9  /* API Includes */
10 var AbstractModel = require('./AbstractModel');
11 var ArrayList = require('dw/util/ArrayList');
12 var BasketMgr = require('dw/order/BasketMgr');
13 var Money = require('dw/value/Money');
14 var MultiShippingLogger = dw.system.Logger.getLogger('multishipping');
15 var OrderMgr = require('dw/order/OrderMgr');
16 var PaymentInstrument = require('dw/order/PaymentInstrument');
17 var Product = require('~cartridge/scripts/models/ProductModel');
18 var ProductInventoryMgr = require('dw/catalog/ProductInventoryMgr');
19 var ProductListMgr = require('dw/customer/ProductListMgr');
20 var QuantityLineItem = require('~cartridge/scripts/models/QuantityLineItemModel');
21 var Resource = require('dw/web/Resource');
22 var ShippingMgr = require('dw/order/ShippingMgr');
23 var StoreMgr = require('dw/catalog/StoreMgr');
24 var TransientAddress = require('~cartridge/scripts/models/TransientAddressModel');
25 var UUIDUtils = require('dw/util/UUIDUtils');
26
27 var lineItem;
28 var app = require('~cartridge/scripts/app');
29 var ProductList = app.getModel('ProductList');
30
31 // Import Extend Helpers
32 var extendHelpers = require('*/*cartridge/scripts/extendHelpers');
```





4.4. Include Extend line item creation in CartModel.js

- In: */app_storefront_controllers/cartridge/scripts/models/CartModel.js
- Where: function addProductToCart(), before the return statement

Code:

```
// Extend Create Warranty LineItem
extendHelpers.createExtendLineItem(cart, params, Product, Transaction);
```

```
106      // Adds a product.
107  } else {
108      var previousBonusDiscountLineItems = cart.getBonusDiscountLineItems();
109      productToAdd = Product.get(params.pid.stringValue);
110
111      if (productToAdd.object.isProductSet()) {
112          var childPids = params.childPids.stringValue.split(',');
113          var childQtys = params.childQtys.stringValue.split(',');
114          var counter = 0;
115
116          for (var i = 0; i < childPids.length; i++) {
117              var childProduct = Product.get(childPids[i]);
118
119              if (childProduct.object && !childProduct.isProductSet()) {
120                  var childProductOptionModel = childProduct.updateOptionSelection(params);
121                  cart.addProductItem(childProduct.object, parseInt(childQtys[counter]), childProductOptionModel);
122              }
123              counter++;
124          }
125      } else {
126          productOptionModel = productToAdd.updateOptionSelection(params);
127          cart.addProductItem(productToAdd.object, params.Quantity.doubleValue, productOptionModel);
128      }
129
130      // When adding a new product to the cart, check to see if it has triggered a new bonus discount line item.
131      newBonusDiscountLineItem = cart.getNewBonusDiscountLineItem(previousBonusDiscountLineItems);
132  }
133
134  // Extend Create Warranty LineItem
135  extendHelpers.createOrUpdateExtendLineItem(cart, params, Product);
136
137  return {
138      format: format,
139      template: template,
140      BonusDiscountLineItem: newBonusDiscountLineItem
141  };
142},
143
```





4.5. Import extendHelpers.js in Cart

- In: */app_storefront_controllers/cartridge/controllers/Cart.js
- Where: include this script at the API Includes section

Code:

```
var extendHelpers = require('*cartridge/scripts/extendHelpers');
```

```
1  'use strict';
2
3  /**
4   * Controller that adds and removes products and coupons in the cart.
5   * Also provides functions for the continue shopping button and minicart.
6   *
7   * @module controllers/Cart
8   */
9
10 /* API Includes */
11 var ArrayList = require('dw/util/ArrayList');
12 var ISML = require('dw/template/ISML');
13 var Resource = require('dw/web/Resource');
14 var Transaction = require('dw/system/Transaction');
15 var URLUtils = require('dw/web/URLUtils');
16
17 /* Script Modules */
18 var app = require('~/cartridge/scripts/app');
19 var guard = require('~/cartridge/scripts/guard');
20
21 /* EXTEND */
22 var extendHelpers = require('*cartridge/scripts/extendHelpers');
```





4.6. Add functionality to remove Extend warranty line items in Cart.js

- In: */app_storefront_controllers/cartridge/controllers/Cart.js
- Where: ‘deleteProduct’ action handler; replace this entire function with the following code

Code:

```
'deleteProduct': function (formgroup) {
    // EXTEND - Check if there is a warranty associated with the product
    // that is being deleted
    var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart,
    formgroup.getTriggeredAction().object);
    Transaction.wrap(function () {
        cart.removeProductLineItem(formgroup.getTriggeredAction().object);

        // EXTEND - if there is a warranty associated with the product
        remove the corresponding warranty
        if (warrantyLineItem) {
            cart.removeProductLineItem(warrantyLineItem);
        }
    });
    return {
        cart: cart
    };
}
```

```
168  'deleteGiftCertificate': function (formgroup) {
169      Transaction.wrap(function () {
170          cart.removeGiftCertificateLineItem(formgroup.getTriggeredAction().object);
171      });
172
173      return {
174          cart: cart
175      };
176  },
177  'deleteProduct': function (formgroup) {
178
179      // EXTEND - Check if there is a warranty associated with the product that is being deleted
180      var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart, formgroup.getTriggeredAction().object);
181
182      Transaction.wrap(function () {
183          cart.removeProductLineItem(formgroup.getTriggeredAction().object);

184          // EXTEND - if there is a warranty associated with the product remove the corresponding warranty
185          if (warrantyLineItem) {
186              cart.removeProductLineItem(warrantyLineItem);
187          }
188      });
189
190      return {
191          cart: cart
192      };
193  },
194  'editLineItem': function (formgroup) {
```





4.7. Add contract generation snippet to COPlaceOrder.js

- In: */app_storefront_controllers/cartridge/controllers/COPlaceOrder.js
- Where: function submit(), in the if (!orderPlacementStatus.error) statement

Code:

```
// Add Extend products to Contracts queue
var extendHelpers = require('*cartridge/scripts/extend');
addContractToQueue(order.object);
```

```
164     } else if (handlePaymentsResult.missingPaymentInfo) {
165         return Transaction.wrap(function () {
166             OrderMgr.failOrder(order);
167             return {
168                 error: true,
169                 PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical')
170             };
171         });
172     }
173
174     var orderPlacementStatus = Order.submit(order);
175     if (!orderPlacementStatus.error) {
176         clearForms();
177
178         // Add Extend products to Contracts queue
179         var extendHelpers = require('*cartridge/scripts/extendHelpers');
180         extendHelpers.addContractToQueue(order);
181     }
182     return orderPlacementStatus;
183 }
```

4.8. CartView.js

- In:
/cartridges/app_storefront_controllers/cartridge/scripts/views/CartView.js
- Where: line 38
- Move the following code snippet from line 38 to line 33 to ensure the warranty price is recalculated before displaying in the cart.

Code:

```
// Refreshes the cart calculation
Transaction.wrap( function () {
    Cart.get(cart).calculate();
})
```

See screenshots below:





Code location **before**:

```
var CartView = View.extend({  
  
    /**  
     * Updates shipments, coupons, and cart calculation for the view and validates  
     * for checkout.  
     */  
    prepareView: function () {  
  
        var cart = this.Basket;  
        if (cart) {  
  
            // Refreshes shipments.  
            session.forms.cart.shipments.copyFrom(cart.shipments);  
            // Refreshes coupons.  
            session.forms.cart.coupons.copyFrom(cart.couponLineItems);  
  
            // Refreshes the cart calculation.  
            Transaction.wrap(function () {  
                Cart.get(cart).calculate();  
            });  
  
            var validationResult = Cart.get(cart).validateForCheckout();  
            this.EnableCheckout = validationResult.EnableCheckout;  
            this.BasketStatus = validationResult.BasketStatus;  
            this.WishList = customer.authenticated ? require('~/cartridge/scripts/m...  
  
            // Extend Analytics.  
            extendAnalyticsHelpers.setUpdateCartPayload(cart);  
            this.analyticsPayload = extendAnalyticsHelpers.getAnalyticsPayload();  
        }  
  
        return;  
    },  
},
```





Code location **after**:

```
1  var CartView = View.extend([
2
3
4    /**
5     * Updates shipments, coupons, and cart calculation for the view and validates the
6     * for checkout.
7     */
8    prepareView: function () {
9
10      var cart = this.Basket;
11      if (cart) {
12
13        // Refreshes the cart calculation.
14        Transaction.wrap(function () {
15          |   Cart.get(cart).calculate();
16        });
17
18        // Refreshes shipments.
19        session.forms.cart.shipments.copyFrom(cart.shipments);
20        // Refreshes coupons.
21        session.forms.cart.coupons.copyFrom(cart.couponLineItems);
22
23        var validationResult = Cart.get(cart).validateForCheckout();
24        this.EnableCheckout = validationResult.EnableCheckout;
25        this.BasketStatus = validationResult.BasketStatus;
26        this.WishList = customer.authenticated ? require('~/cartridge/scripts/model'
27
28        // Extend Analytics.
29        extendAnalyticsHelpers.setUpdateCartPayload(cart);
30        this.analyticsPayload = extendAnalyticsHelpers.getAnalyticsPayload();
31      }
32
33      return;
34    },
35  ],
```





4.9. Include ExtendAnalyticsHelpers in CartView.js

- In:
/cartridges/app_storefront_controllers/cartridge/scripts/views/CartView.js
- Where: top of the file

Code:

```
var extendAnalyticsHelpers =
require('*cartridge/scripts/extendAnalyticsHelpers')
```

```
/**
 * View used to render the cart. This view makes sure the coupons, shipments, and basket
 * calculation are up to date before rendering the cart.
 * @module views/CartView
 */
var View = require('./View');

var Cart = require('~/cartridge/scripts/models/CartModel');
var Transaction = require('dw/system/Transaction');
var extendAnalyticsHelpers = require('*cartridge/scripts/extendAnalyticsHelpers')
```

- In:
/cartridges/app_storefront_controllers/cartridge/scripts/views/CartView.js
- Where: “prepareView” function → line 48

Code:

```
// Extend Analytics
extendAnalyticsHelpers.setUpdateCartPayload(cart)
this.analyticsPayload = extendAnalyticsHelpers.getAnalyticsPayload()
```





```
22 var CartView = View.extend([
23
24     /**
25      * Updates shipments, coupons, and cart calculation for the view and validates the
26      * for checkout.
27      */
28     prepareView: function () {
29
30         var cart = this.Basket;
31         if (cart) {
32
33             // Refreshes the cart calculation.
34             Transaction.wrap(function () {
35                 Cart.get(cart).calculate();
36             });
37
38             // Refreshes shipments.
39             session.forms.cart.shipments.copyFrom(cart.shipments);
40             // Refreshes coupons.
41             session.forms.cart.coupons.copyFrom(cart.couponLineItems);
42
43             var validationResult = Cart.get(cart).validateForCheckout();
44             this.EnableCheckout = validationResult.EnableCheckout;
45             this.BasketStatus = validationResult.BasketStatus;
46             this.WishList = customer.authenticated ? require('~/cartridge/scripts/model'
47
48             // Extend Analytics.
49             extendAnalyticsHelpers.setUpdateCartPayload(cart);
50             this.analyticsPayload = extendAnalyticsHelpers.getAnalyticsPayload();
51         }
52
53         return;
54     },
55 ])
```





4.10 Include ExtendAnalyticsHelpers.setDeleteProductPayload in Cart.js

- In:
/cartridges/app_storefront_controllers/cartridge/controllers/Cart.js
- Where: top of the file

Code:

```
var extendHelpers = require('*cartridge/scripts/extendHelpers');
var extendAnalyticsHelpers =
require('*cartridge/scripts/extendAnalyticsHelpers');
```

```
1  'use strict';
2
3 /**
4  * Controller that adds and removes products and coupons in the cart.
5  * Also provides functions for the continue shopping button and minicart.
6  *
7  * @module controllers/Cart
8 */
9
10 /* API Includes */
11 var ArrayList = require('dw/util/ArrayList');
12 var ISML = require('dw/template/ISML');
13 var Resource = require('dw/web/Resource');
14 var Transaction = require('dw/system/Transaction');
15 var URLUtils = require('dw/web/URLUtils');
16
17 /* Script Modules */
18 var app = require('~/cartridge/scripts/app');
19 var guard = require('~/cartridge/scripts/guard');
20
21 var extendHelpers = require('*cartridge/scripts/extendHelpers')
22 var extendAnalyticsHelpers = require('*cartridge/scripts/extendAnalyticsHelpers')
```

- In:
/cartridges/app_storefront_controllers/cartridge/controllers/Cart.js
- Where: “submitForm” function → “deleteProduct” function → line 182

Code:

```
var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart,
formgroup.getTriggeredAction().object);
extendAnalyticsHelpers.setDeleteProductPayload(cart,
formgroup.getTriggeredAction().object, warrantyLineItem);
```





```
177     };
178 },
179 'deleteProduct': function (formgroup) {
180     // EXTEND - Check if there is a warranty associated with the product that is being deleted
181     var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart, formgroup.getTriggeredAction().object);
182     extendAnalyticsHelpers.setDeleteProductPayload(cart, formgroup.getTriggeredAction().object, warrantyLineItem);
183     Transaction.wrap(function () {
184         cart.removeProductLineItem(formgroup.getTriggeredAction().object);
185
186         // EXTEND - if there is a warranty associated with the product remove the corresponding warranty
187         if (warrantyLineItem) {
188             cart.removeProductLineItem(warrantyLineItem);
189         }
190     });
191
192     return [
193         cart
194     ];
195 },
```

4.11 Include “Extend Integration” and “Extend Analytics Integration” in app.js

- In:
/cartridges/app_storefront_core/cartridge/static/default/js/app.js
- Where:
“addToCart” function. Line 2537 - 2547

Code:

```
// Extend Integration
if ($('#extend-offer').length || (window.EXT_GLOBAL_SWITCH &&
window.EXT_PDP_UPSELL_SWITCH)) {
    extendAddToCart($form, page, minicart, dialog, addItemToCart);
    return;
}
// ExtendAnalytics Integration
if (window.EXT_A_EXT_GLOBAL_SWITCH) {
    var payload = createAddToCartAnalytics($form)
    trackAddToCart(payload)
}
```





```
2529
2530 /**
2531 * @description Handler to handle the add to cart event
2532 */
2533 var addToCart = function (e) {
2534   e.preventDefault();
2535   var $form = $(this).closest('form');
2536
2537   // Extend Integration
2538   if ($('#extend-offer').length || (window.EXT_GLOBAL_SWITCH && window.EXT_PDP_UPSELL_SWITCH)) {
2539     extendAddToCart($form, page, minicart, dialog, addItemToCart);
2540     return;
2541   }
2542
2543   // ExtendAnalytics Integration
2544   if (window.EXT_A_EXT_GLOBAL_SWITCH) {
2545     var payload = createAddToCartAnalytics($form)
2546     trackAddToCart(payload)
2547   }
2548
2549   addItemToCart($form).then(function (response) {
2550     |
```

5. External Interfaces

All requests are done through the Extend API. The full reference guide, along with the resource structure for requests, can be found at <https://docs.extend.com/>.

6. Firewall Requirements

None.





7. Orders API

7.1. Custom preferences

- i. Go to Business Manager → Merchant Tools → Site Preferences → Custom Preferences → Extend: Integration Preferences
- ii. Configure the provided Extend credentials in the following fields (refer to Section 2.2 for additional info):
 - Extend Access Token
 - Extend Store ID
- iii. Configure the following Orders API specific preferences:
 - **Extend Integration Method:** set this to Orders API on Order Create (please confirm with the Extend project team before changing versions).

7.2. Testing

i. LeadToken:

Go to PDP → Choose product variation → Click “Add to Cart” button without an Extend Protection Plan (LeadTokens are only generated for warrantable products if the customer did not elect to protect their purchase originally) → Go to cart → Go to checkout page → Make a purchase.

Result:

Go to Merchant Tools → Ordering → Orders → Choose Your Order → Shipment → Choose ProductLineItem → Attributes





General

Attributes

Attributes for "Black Flat Front Wool Suit"

On this page you can edit the attributes of the product line item. Fields with a red asterisk (*) are mandatory. Click **Apply** to save changes.

Extend

Extend Contract ID: No data is available

Extend Refund Statuses:

Lead Token: f102f17d9fda6fe8cd8a76962ae68dd9

ii. Extend Contract ID:

Go to PDP → Choose product variation → Choose an extension plan → Click “Add to Cart” button with an Extend Protection Plan → Go to cart → Go to checkout page → Make a purchase.

Result:

Go to Merchant Tools → Ordering → Orders → Choose Your Order → Shipment → Choose ExtendLineItem → Attributes





[Merchant Tools](#) > [Ordering](#) > [Orders](#) > Order: 00006006(SiteGenesis) > Shipment: 00032506 > Line Item: EXTEND-24

General

Attributes

Attributes for "Extend Product Protection: 2 years"

On this page you can edit the attributes of the product line item. Fields with a red asterisk (*) are mandatory. Click **Apply** to save changes.

Extend

Extend Contract ID: 764a42a5-c9f2-442d-9916-03061b22bf97

Extend Refund Statuses:

Lead Token:

iii. Refund status:

- Go to Merchant Tools → Ordering → Orders → Choose an Order with at least 1 created Extend Contract ID → Order Status → Choose “**Cancelled**” → Click “**Apply**” button.
- Then Go To Administration → Operations → Jobs → “**Create Refund from SFCC**” → Click “Run Now” button.

Result:

Go to Merchant Tools → Ordering → Orders → Choose Your Order → Shipment → Choose ExtendLineItem → Attributes:





Merchant Tools > Ordering > Orders > Order: 00006006(SiteGenesis) > Shipment: 00032506 > Line Item: EXTEND-24

General

Attributes

Attributes for "Extend Product Protection: 2 y"

On this page you can edit the attributes of the product line item. Fields with a red asterisk (*) are mandatory. Click **Apply** to save changes.

Extend

Extend Contract ID: 764a42a5-c9f2-442d-9916-03061b22bf97

Extend Refund Statuses: {"764a42a5-c9f2-442d-9916-03061b22bf97":"refund_paid"}

Lead Token:

Go to Merchant Tools → Ordering → Orders → Choose Your Order → Attributes:





[Merchant Tools](#) > [Ordering](#) > [Orders](#) > Order: 00006006(SiteGenesis)

General

Attributes

Payment

Notes

History

Attributes for Order '00006006'

On this page you can edit the attributes of the order. Fields with a red asterisk (*) are mandatory. Click

Extend

Extend Refund Status:

Refund Statuses:

refund_quoted: Refund amounts have been previewed by merchant/customer at least once, but merchant has not reported that they have refunded the customer yet.

refund_paid: Refund has been reported as paid by the merchant. Contract is canceled. Finance will credit the merchant in the next monthly invoice.

refund_denied: Customer is not eligible for a refund. Customer has consumed all entitlements associated with the contract. Get Refund via Contract ID.

ERROR: service call error.





8. Extend SDK analytics

8.1. Custom Preferences

- i. Go to Business Manager → Merchant Tools → Site Preferences → Custom Preferences → Extend: Storefront Preferences
- ii. Configure the provided Extend credentials in the following fields (refer to Section 2.2 for additional info):
 - Extend Access Token
 - Extend Store ID
- iii. Ensure “Extend Analytics” is set to “Yes”.

Extend Analytics Switch	Yes
(extendAnalyticsSwitch)	

8.2. Tracking events

i. Extend.trackOfferViewed:

This method is triggered when an Extend warranty offer is rendered on the screen to a user.

Name	x Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> Product-Productnav?lang=en_US&start=1&pid=25686514&format/ajax <input type="checkbox"/> offers?storeId=5bcd9cf9-807a-4389-ba29-6d38daf7d395&productId=25686... <input type="checkbox"/> Product-Variation?pid=25686514&dwvar_25686514_size...WL&Quantity=1&... <input type="checkbox"/> offers?storeId=5bcd9cf9-807a-4389-ba29-6d38daf7d395&productId=25686... <input type="checkbox"/> Product-Variation?pid=25686514&dwvar_25686514_widt...WL&Quantity=1... <input type="checkbox"/> tracking		<p>Request Payload view source</p> <pre>▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "offer_viewed",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {...} eventName: "offer_viewed" platform: "extend-sdk-client" ▶ properties: {productId: "750518548203", offerType: {area: "product_page", component: "buttons"}} ▶ offerType: {area: "product_page", component: "buttons"} productId: "750518548203" ▶ sessionState: {products: [], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643290228798 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				

ii. Extend.trackProductAddedToCart:

This method is triggered when an user adds a (non-Extend) product to the cart. This event accepts the product's “productId” and the number of units (quantity) added to the cart.





Name	Headers	Payload	Preview	Response	Initiator	Timing
https://api-demo.helloextend.com/tracking?_t=1&pid=25686514&format=ajax	Request Payload view source					
<input type="checkbox"/> offers?storeId=5bcd9cf9-807a-4389-ba29-6d38daf7d395&productId=25686...	{type: "analytics",...} object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "product_added_to_cart",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" context: {,...} eventName: "product_added_to_cart" platform: "extend-sdk-client" properties: {productId: "750518548203", productQuantity: 1} productId: "750518548203" productQuantity: 1 sessionState: {products: [{productId: "750518548203", productQuantity: 1}], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643290587268 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"					

iii. Extend.trackOfferAddedToCart

This method is triggered when an user adds an Extend warranty to the cart. This event accepts the product's "productId", the number of units (quantity), and the Extend warranty's "planId".

Name	Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> tracking	Request Payload view source					
<input type="checkbox"/> Extend-AddExtendProduct	{type: "analytics",...} object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "offer_added_to_cart",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" context: {,...} eventName: "offer_added_to_cart" platform: "extend-sdk-client" properties: {productId: "750518548203", productQuantity: 1, warrantyQuantity: 1,...} offerType: {area: "cart_page", component: "modal"} planId: "10001-misc-elec-base-replace-3y" productId: "750518548203" productQuantity: 1 warrantyQuantity: 1 sessionState: {products: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643290855612 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"					

iv. Extend.trackOfferRemovedFromCart

This method is triggered when an user removes an Extend warranty from the cart. This event accepts the Extend warranty's "planId" and the associated "productId" of the product the warranty would have covered.

Name	Headers	Payload	Preview	Response	Initiator	Timing
https://api-demo.helloextend.com/tracking?_t=1&offerId=596859ab60bb9ab25a226641f	Request Payload view source					
<input type="checkbox"/> tracking	{type: "analytics",...} object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "offer_removed_from_cart",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" context: {,...} eventName: "offer_removed_from_cart" platform: "extend-sdk-client" properties: {productId: "750518548203", planId: "10001-misc-elec-base-replace-3y"} planId: "10001-misc-elec-base-replace-3y" productId: "750518548203" sessionState: {products: [{productId: "750518548203", productQuantity: 1}], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643291063318 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"					





v. Extend.trackOfferUpdated

This method is triggered when an user increments or decrements the quantity of a warranty that has already been added to the cart. This event takes the Extend warranty's "planId", the associated "productId" of the product the warranty covers, as well as an update object containing the set of updates to apply to the warranty offer. **If the quantity of warranty is updated to 0, the a Extend.trackOfferRemoved event is called**, and further updates to this planId/productId will result in a no-op until it is re-added via Extend/trackOfferAddedToCart.

Name	x Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> Extend.IsEligibleForWarranty?uuid=7896859ab60bb9ab25a226641f <input type="checkbox"/> Extend.IsEligibleForWarranty?uuid=eeeab5379a83081e350fc5f3aa <input checked="" type="checkbox"/> tracking		<pre>▼ Request Payload view source ▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "offer_updated",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {,..} eventName: "offer_updated" platform: "extend-sdk-client" properties: {productId: "750518548203", planId: "10001-misc-elec-base-replace-ly",...} ▶ newQuantities: {warrantyQuantity: 2, productQuantity: 2} ▶ oldQuantities: {productQuantity: 1, warrantyQuantity: 1} planId: "10001-misc-elec-base-replace-ly" productId: "750518548203" ▶ sessionState: {products: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643291432640 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				

vi. Extend.trackProductRemovedFromCart

This method is triggered when an user removes a product from the cart that **does not** have a warranty offer associated with it. This event takes the "productId" of the product being removed from the cart.

Name	x Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> tracking		<pre>▼ Request Payload view source ▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "product_removed",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {,..} eventName: "product_removed" platform: "extend-sdk-client" properties: {productId: "750518548203"} productId: "750518548203" ▶ sessionState: {products: [], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643291664323 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				

vii. Extend.trackProductUpdated

This method is triggered when an user increments or decrements the quantity of a product that has already been added to the cart. This event takes the "productId" of the product being updated, as well as an update object





containing the set of updates to apply to the product. The product being updated must **not** be associated with a warranty offer.

If the “productQuantity” passed into this method is 0, then Extend.trackProductRemoveFromCart is fired.

Name	Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> Extend-IsEligibleForWarranty?uuid=f48cc344ec25b9d94f038586e8 <input type="checkbox"/> tracking <input type="checkbox"/> tracking		<pre>▼ Request Payload view source ▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "product_updated",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {...} eventName: "product_updated" platform: "extend-sdk-client" ▼ properties: {productId: "750518699608M", oldProductQuantity: 1, newProductQuantity: 2} newProductQuantity: 2 oldProductQuantity: 1 productId: "750518699608M" ▶ sessionState: {products: [{productId: "750518699608M", productQuantity: 2}], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643292010508 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				

viii. Extend.trackCartCheckout

This method is triggered when a customer completes a purchase.

- After the Checkout flow is complete (at Order Confirmation page), there fires two tracking events - offer_sold, and total_cart_revenue

Name	Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...ess2=&country... <input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...ountryCode=u... <input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...s&stateCode=... <input type="checkbox"/> COBilling-SelectCreditCard?creditCardUUID=3a1e6870902235f9c0771e4bf1 <input type="checkbox"/> tracking <input type="checkbox"/> tracking		<pre>▼ Request Payload view source ▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "offer_sold",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {...} eventName: "offer_sold" platform: "extend-sdk-client" ▼ properties: {offerType: {area: "product_page", component: "buttons"}, planId: "10001-misc-elec-base-replace-3y",...} offerType: {area: "product_page", component: "buttons"} planId: "10001-misc-elec-base-replace-3y" productId: "75051894485" productQuantity: 1 warrantyQuantity: 1 ▶ sessionState: {products: [], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643292435532 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				

Name	Headers	Payload	Preview	Response	Initiator	Timing
<input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...ess2=&country... <input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...ountryCode=u... <input type="checkbox"/> COShipping-GetApplicableShippingMethods.JSON?address...s&stateCode=... <input type="checkbox"/> COBilling-SelectCreditCard?creditCardUUID=3a1e6870902235f9c0771e4bf1 <input type="checkbox"/> tracking <input type="checkbox"/> tracking		<pre>▼ Request Payload view source ▼ {type: "analytics",...} ▼ object: {userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7", eventName: "total_cart_revenue",...} cartId: "0BF2F4B4-7570-4A4E-99E5-416E0840CA88" ▶ context: {...} eventName: "total_cart_revenue" platform: "extend-sdk-client" ▼ properties: {cartTotal: 409.47} cartTotal: 409.47 ▶ sessionState: {products: [], productsWithWarranties: [],...} storeId: "5bcd9cf9-807a-4389-ba29-6d38daf7d395" timestamp: 1643292435534 userId: "1B5D4861-6DE1-4552-96E6-65624D3E8EB7" type: "analytics"</pre>				





- After the Checkout flow is complete (at Order Confirmation page), there fires two tracking events called - product_sold and total_cart_revenue

Name	x Headers	Payload	Preview	Response	Initiator	Timing
tracking						
tracking						

Request Payload view source

```
{
  "type": "analytics",
  "object": {
    "userId": "1B5D4861-6DE1-4552-96E6-65624D3E8EB7",
    "eventName": "product_sold",
    "cartId": "669FA4D1-70F4-4463-B816-28855403B9C5",
    "context": {},
    "eventName": "product_sold",
    "platform": "extend-sdk-client",
    "properties": {
      "productId": "750518894485",
      "productQuantity": 1
    },
    "productId": "750518894485",
    "productQuantity": 1,
    "sessionState": {
      "products": [
        {
          "productId": "750518894485",
          "productQuantity": 1
        }
      ],
      "productsWithWarranties": []
    },
    "storeId": "5bcd9cf9-807a-4389-ba29-6d38daf7d395",
    "timestamp": 1643292733992,
    "userId": "1B5D4861-6DE1-4552-96E6-65624D3E8EB7",
    "type": "analytics"
  }
}
```

Name	x Headers	Payload	Preview	Response	Initiator	Timing
tracking						
tracking						

Request Payload view source

```
{
  "type": "analytics",
  "object": {
    "userId": "1B5D4861-6DE1-4552-96E6-65624D3E8EB7",
    "eventName": "total_cart_revenue",
    "cartId": "669FA4D1-70F4-4463-B816-28855403B9C5",
    "context": {},
    "eventName": "total_cart_revenue",
    "platform": "extend-sdk-client",
    "properties": {
      "cartTotal": 325.48
    },
    "cartTotal": 325.48,
    "sessionState": {
      "products": [],
      "productsWithWarranties": []
    },
    "storeId": "5bcd9cf9-807a-4389-ba29-6d38daf7d395",
    "timestamp": 1643292733993,
    "userId": "1B5D4861-6DE1-4552-96E6-65624D3E8EB7",
    "type": "analytics"
  }
}
```

9. Locale restrictions.

None.

10. Failover/Recovery process.

The cartridge's operation is based on exchanges of data via Extend APIs. Please reach out to your Extend support team for any integration questions and issues.





Testing

1. Enable Extend

- Ensure the following switches are set to Yes
 - Enable Extend
 - Enable PDP Offers
 - Enable Modal Offers
 - Enable Cart Offers
- Ensure the product catalog has been exported to Extend and confirm with the Extend project team that warranty plans are enabled in the system.

2. PDP

2.1. Extend Plan

Navigate to any simple or variant product that has an Extend Protection Plan enabled. Select a variation and confirm an Extend offer is shown.

The screenshot shows a product detail page for a "Must Have Washable No-Iron Georgette Blouse". The page includes a navigation bar with links for NEW ARRIVALS, WOMENS, MENS, ELECTRONICS, and TOP SELLERS. A search bar and a shopping cart icon (with 6 items) are also present. The main content features a large image of a woman wearing the blouse, its name, item number, price (\$69.00), color selection (Ivory), size selection (6, 8, 10, 12), availability (In Stock), and an "Add product protection from Extend" button with options for 1, 2, or 3 years. Social sharing icons and wishlist/gift registry buttons are at the bottom.

FREE 2-Day SHIPPING FOR ORDERS OVER \$300

salesforce commerce cloud

Enter Keyword or Item No.

NEW ARRIVALS WOMENS MENS ELECTRONICS TOP SELLERS

Must Have Washable No-Iron Georgette Blouse

Item No. 008885538465M

\$69.00

SELECT COLOR

IVORY

SELECT SIZE

6 8 10 12 8

AVAILABILITY

In Stock

Add product protection from Extend [Learn more](#)

1 Year - \$6.99 2 Years - \$12.99 3 Years - \$17.99

QTY 1 **ADD TO CART**

[f](#) [t](#) [g+](#) [p](#) [e](#)

[Add to Wishlist](#) [Add to gift registry](#)





2.2. Add to Cart

Select an Extend Protection Plan and click Add to Cart. Confirm 2 products were added to the cart – the SFCC catalog product along with the Extend Protection Plan.

2.3. Up-sell Modal

Navigate to any simple or variant product that has an Extend Protection Plan enabled and click Add to Cart on the PDP without selecting any Extend Protection Plan. Confirm the Extend Interstitial Modal is presented. Selecting “Add Protection” will add the Extend Protection Plan to the cart, along with the catalog product. Selecting “No, thanks” will only add the catalog product to the cart.

The screenshot shows a product detail page for a "Must Have Washable No-Iron Georgette Blouse" priced at \$69.00. The modal, titled "Protect your purchase from drops, breaks, spills, and defects with Extend.", offers three protection plan options:

Protection Plan	Price	Status
1 Year Protection Plan	\$6.99	
2 Year Protection Plan	\$11.99	Best Seller
3 Year Protection Plan	\$17.99	Best Value

Buttons at the bottom of the modal include "No, thanks" and "Add protection". A woman smiling and holding a package is visible on the right side of the screen.





3. Cart

3.1. Extend Plans in Cart

Confirm an Extend plan has been added to the cart.

FREE 2-DAY SHIPPING FOR ORDERS OVER \$300

salesforce commerce cloud Enter Keyword or Item No.

NEW ARRIVALS WOMENS MENS ELECTRONICS TOP SELLERS

« Continue Shopping **CHECKOUT**

PRODUCT	DELIVERY OPTIONS	QTY	PRICE	TOTAL
Must Have Washable No-Iron Georgette Blouse Item No.: 008885538465M Color Ivory Size 8 Edit Details	Home Delivery	<input type="text" value="1"/> In Stock Remove Add to Wishlist	\$69.00	\$69.00
 Extend Product Protection: 2 years for Must Have Washable No-Iron Georgette Blouse Item No.: EXTEND-24 Extend Term: 2Y	Home Delivery	<input type="text" value="1"/> In Stock Remove	\$12.99	\$12.99

Enter coupon code **Apply** **Update Cart**

Subtotal \$81.99
Shipping -
Sales Tax -
Estimated Total \$81.99

« Continue Shopping **CHECKOUT**

The Extend Product Protection offer is highlighted with a red box.

3.2. Up-sell button for eligible products in cart

Add an eligible catalog product to the cart, without selecting any Extend Protection Plan for it. Navigate to the cart page and confirm the cart offer button is presented.



PRODUCT	DELIVERY OPTIONS	QTY	PRICE	TOTAL
 <p>Must Have Washable No-Iron Georgette Blouse Item No.: 008885538465M Color Ivory Size 8 Edit Details</p> <p>Add product protection from \$6.99</p>	Home Delivery	<input type="text" value="1"/> In Stock Remove Add to Wishlist	\$69.00	\$69.00
<input type="text" value="Enter coupon code"/> Apply Update Cart			Subtotal \$69.00 Shipping -	\$69.00

3.3. Up-sell modal in cart

Click on the up-sell cart offer button and confirm the Extend Interstitial Modal is presented. Selecting “Add Protection” will add the corresponding Extend Protection Plan to the cart. Selecting “No, thanks” will close the modal.


Must Have Washable No-Iron Georgette Blouse
Item No.: 008885538458
Color Ivory
Size 6
[Edit Details](#)

[Add accident protection for \\$6.99](#)

[Apply](#)

 + 

Protect your purchase from drops, breaks, spills, and defects with Extend.

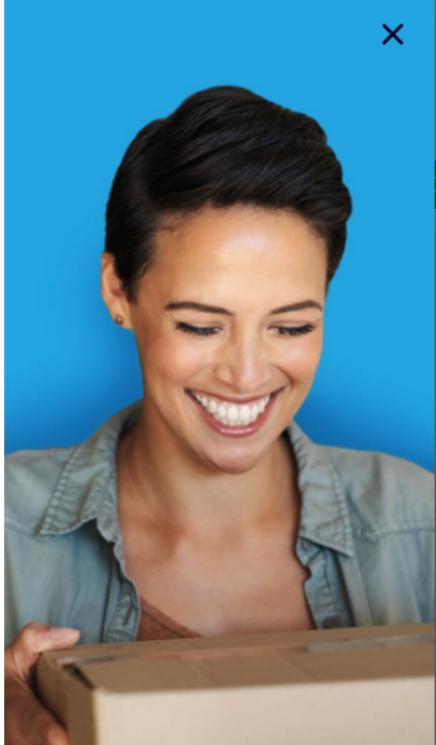
This plan covers:

- Fast and free product replacements
- Accidental damage such as breaks, drops, and spills
- Extended malfunction and wear-and-tear protection

[See plan details ↗](#) Offered and sold by Extend

1 Year Protection Plan \$6.99	2 Year Protection Plan \$11.99 <small>Best Seller</small>	3 Year Protection Plan \$17.99 <small>Best Value!</small>
----------------------------------	--	--

[No, thanks](#) [Add protection](#)



FEATURED PRODUCTS





4. Business Manager

4.1. ExtendOrderQueue custom object

This custom object is used to track which SFCC order has not been processed yet.

Orders API on Schedule creates a custom queue record (*ExtendOrderQueue*) upon order creation in the SFCC Business Manager, but will not integrate with Extend until the Job is run manually or on a schedule. This job will consume the custom queue.

Steps:

1. Buying the product with/without any extensions.
2. In the case of **Orders API on Schedule** method is chosen the custom objects created.
3. There are examples below:

Custom Object Search				Simple	Advanced
Object Type:	ExtendOrderQueue	Object ID:	Find		
Select All	Extend Order Number (OrderNo)	Scope	Last Modified	Expires On	
<input type="checkbox"/>	00010601	Site	8/8/22 3:58:48 am	11/6/22 2:58:48 am	
Edit All	Edit Selected			New	Delete

Extend Order Number (OrderNo) - Order number for further processing

List Modified - Creation Date

Expires On - Date when the object will be deleted if it is not processed

There is an order information below:





General

Manage '00010601' (ExtendOrderQueue)

Fields with a red asterisk (*) are mandatory. You can view and edit the name and description in other languages, if required. Click A

Extend Order Data	
Extend Order Number: *	00010601
Creation Date: *	08/08/2022 : 7:58 am MM/dd/yyyy h:mm a
Last Modified: *	08/08/2022 : 7:58 am MM/dd/yyyy h:mm a
Order Customer:	{"phone": "3333333333", "email": "test@gmail.com", "name": ""}
Order Total:	4,829.00 (Number)
Site currency:	USD
Order Shipping Address:	{"address1": "1404 S Federal ST", "address2": null, "city": "C", "state": "CA", "zip": "95014", "country": "US"}
Log information:	

Extend Order Number - SFCC Order Number

Order Customer - Buyer Information

Order Total - Order Summary Total

Site Currency - Currency

Order Shipping Address - Shipping Address

Log information - Log field for error message

All the fields (besides three first) are custom. Could be changed/deleted/added the new fields.

4. Then, if necessary, using the [job](#) all the custom objects are processed. There are two cases possible:

- In case of buying without any extensions the lead token is created. Objects are deleted after processing



4.2. In case of buying with an extension the contractId is created. Objects are deleted after processing.





4.3. ExtendContractsQueue custom object

Contracts API on Schedule is a legacy feature whereby only Contract creation occurs (requires Extend API Version 2021-04-01 or earlier). Integration with Contracts API uses a Job that consumes a custom queue (ExtendContractQueue) when run manually or on a schedule. This job will consume the custom queue.

The custom objects are created ONLY in case of buying the product with an extension.

Steps:

1. Buying the product **WITH** an extension.
2. In the case of **Contracts API on Schedule** method is chosen the custom objects created.
3. There are examples below:

Custom Object Search				Simple	Advanced
Object Type:	Line Item UUID (LUUID)	Scope	Last Modified	Expires On	
Select All	4f79c2527fed471a9c4f647662-1	Site	8/8/22 4:36:12 am	11/6/22 3:36:12 am	
Edit All	Edit Selected				New Delete

Line Item UUID - Unique identifier for the extension

List Modified - Creation Date

Expires On - Date when the object will be deleted if it is not processed

There is an extension information below:

Manage '4f79c2527fed471a9c4f647662-1' (ExtendContractsQueue)

Fields with a red asterisk (*) are mandatory. You can view and edit the name and description in other languages, if required. Click **Apply** to save the details.

Contract Data	
Site Currency:	USD
Extend Warranty Plan:	{"purchasePrice":3099,"planId":"10001-misc-elec-base-re
SFCC Product:	{"referenceId":"750518894553M","purchasePrice":29999}
Last Modified:	08/08/2022 : 8:36 am MM/dd/yyyy h:mm a
Creation Date:	08/08/2022 : 8:36 am MM/dd/yyyy h:mm a
Order Customer:	{"phone":"3333333333","email":"test@gmail.com","name"
Log:	
Line Item UUID:	4f79c2527fed471a9c4f647662-1
Order Total:	35,802.00 (Number)
Order Number:	00010604
Product Shipping Address:	{"address1":"1404 S Federal ST","address2":null,"city":"C





Site Currency - Currency

Extend Warranty Plan - Warranty Plan Information

SFCC Product - Information about the product for which the warranty was purchased

Order Customer - Buyer Information

Log - Log field for error message

Line Item UUID - Unique identifier for the extension

Order Total - Order Summary Total

Order Number - Order Number

Product Shipping Address - Shipping Address

All the fields (besides marked* fields) are custom. Could be changed/deleted/added new fields.

4. Then, if necessary, using the [job](#) all the custom objects are processed. ContractID's are created for all the extensions. Object are deleted after processing.





Questions? Feedback?

At Extend, customer experience is our North Star, and that goes for both our merchant customers (you!) and your store's customers as they buy your products and our protection plans. So, we want to do anything we can to make your Extend integration successful and to make your customers' experiences outstanding.

If you have any questions, please don't hesitate to reach out to us any time or contact integrations@extend.com for help. And if you have any feedback about our documentation, integration process, or our APIs and SDK, we are all ears!

