



Launch by Adobe

Reference Architecture Guide for Standard Retail

by Adobe Customer Solutions

Version 1.0/March 13, 2017



Table of Contents

Reference Architecture Overview	4
Platform Scope	4
Guide Version History	4
Goal	4
Additional Reference Implementation Guides:	4
Assumptions	5
Tools you will need	5
Experience Cloud Debugger Chrome Extension	5
Launch Command Chrome extension	5
Browser Developer Tools	5
The Reference Site	5
General Known Platform Limitations	5
Experience Cloud ID Service Configuration & Tagging	6
Overview	6
Prerequisites & Product-Specific Setup	6
Add Data Elements for Authentication States and Customer Ids	6
Add data element to capture the authentication state	6
Collecting Customer Ids	6
Testing Considerations	8
Adobe Target Tagging & Configuration	11
Overview	11
Prerequisites & Product-Specific Setup	11
Using Customer IDs in Target	11
Parameters	12
Mbox parameters	12
Profile Parameters	12
Entity Parameters for Recommendations	12
Property Token for Enterprise User Permissions	14
Order Confirmation mbox	14
Custom mboxs	15
Library Header and Library Footer replacements	15
Testing Considerations	16



Known Issues or Limitations	17
Adobe Analytics Rule Creation	18
Overview	18
Prerequisites & Product-Specific Setup	18
Rule Creation	18
User Sign-in – Sign in Start.....	18
User Sign-in – Sign in Confirmation	20
Product Details.....	23
Shopping Cart Adds.....	25
Shopping Cart Removes.....	28
Checkout Initiation.....	30
Order Confirmation	32
Site Search	34
Rule Validation: Product, Cart, and Checkout flow	36
Adobe Audience Manager (AAM) Tagging & Configuration	39
Overview	39
Appendix: Data Layer Reference	40
Page and content identification:	40
Product information (product details pages)	40
Cart information and contents:	41
User information and status:	41
Appendix: Data Elements Reference.....	42
Appendix: Test Account Information	47
User account:	47
Order and address information:	47



Reference Architecture Overview

Platform Scope

This Reference Architecture Guide contains best practices for implementing the Adobe Experience Cloud in a website using Launch by Adobe. As a companion to this guide, a reference implementation using these best practices can be found at:

- <https://aem100-us.adobevelab.com/content/we-retail/us/en.html>

Prerequisite steps to guide you through the basic setup Launch properties, environments, and Adobe extensions, are contained in the Basic Setup guide, which can be found on the [Launch by Adobe Reference Architectures landing page](#).

This documentation includes example configurations for Adobe Audience Manager, Adobe Target, and Adobe Analytics. All three products can be configured, or each individually. For purposes of these examples, each configuration is independent, and the only prerequisites for any single deployment are those in the Assumptions section below.

Guide Version History

v1 March 13, 2018 —initial version

Goal

The goal is of this document is to guide you through implementing Adobe Analytics, Target, and Audience Manager using Launch by Adobe on a standard website. For the purposes of this guide, the demonstration website is a basic ecommerce platform, but the examples shown can be adapted for use on a variety of site types.

After completing this you will be able to:

- Identify pages and content
- Capture product SKU and performance
- Measure internal search performance
- Track cart, checkout, and purchase flows
- Identify authenticated users and track sign in behavior

Additional Reference Implementation Guides:

Additional reference implementation guides are available on the [Launch by Adobe Reference Architectures landing page](#), including the following:

1. Basic Setup with Launch by Adobe
2. AngularJS Implementation with Launch by Adobe
3. ReactJS with Redux Implementation with Launch by Adobe
4. Accelerated Mobile Pages (AMP) Implementation with Launch by Adobe



Assumptions

This guide assumes:

- Your Marketing Cloud Organization has been provisioned for Launch and you have full access.
- You have a working knowledge of our current tag management solution, DTM. While experience with DTM is not required to benefit from this Guide, there are many feature comparisons to DTM contained herein.
- You have already reviewed the [Launch Reference Architecture Guide – Basic Setup](#) document.

Tools you will need

This Guide will use all of the latest Adobe tools for Debugging and Quality Assurance (QA).

Experience Cloud Debugger Chrome Extension

The Experience Cloud Debugger is a Chrome extension with debugging features for all SaaS-based Experience Cloud Solutions. Get the extension here:

<https://chrome.google.com/webstore/detail/adobe-experience-cloud-de/ocdmogmohccmeicdhlhhgepeaijenapi>

Launch Command Chrome extension

This is a new extension to switch embed codes. Get the extension here:

<https://chrome.google.com/webstore/detail/launch-command/nkjhamgjeocefocmpbcjfmohkjgildki>

Once it's installed, you select "Options" to view the help docs if you have questions. Later in the Guide we will add Embed Codes to the extension.

Browser Developer Tools

The tools above are both Chrome extensions. If you prefer to use a different browser, make sure you have Developer Tools installed (some browsers require a separate download).

The Reference Site

Bookmark the reference site so you can refer to it later. If you wish, you can copy the source code and replace the Embed Codes with your own: <https://aem100-us.adobevelab.com/content/we-retail/us/en.html>

General Known Platform Limitations

None



Experience Cloud ID Service Configuration & Tagging

Overview

The Experience Cloud ID Service creates a single id for visitors which is then used to integrate the Adobe Experience Cloud solutions. The extension configuration was already covered in the Basic Setup guide, which can be found on the [Launch by Adobe Reference Architectures landing page](#). This section will cover a more advanced implementation scenario of collecting Customer IDs with the Service, which is used to power Customer Attributes and the Device Graph.

Prerequisites & Product-Specific Setup

Add the extension according to the steps in the Basic Setup guide, which can be found on the [Launch by Adobe Reference Architectures landing page](#).

The following Data Elements are used by the ID Service Implementation—see [Appendix: Data Elements Reference](#) for details on how each data element was defined:

- Authentication State—used to determine if the visitor is logged in so we can decide if we need to set a customer ID
- Email (Hashed)—captures the hashed version of the email address (used as the customer ID) from the data layer

Add Data Elements for Authentication States and Customer Ids

Add data element to capture the authentication state

- Click “Data Elements” in the top navigation

Collecting Customer Ids

The Experience Cloud ID Service includes an action called “Set Customer IDs” that sends the customer ID to Adobe. We need to create a rule that triggers this action appropriately:

1. Create a new rule called “All Pages - Top - Authenticated - 10” (this naming convention is to indicate we are implementing this rule at the top of all pages when the user is authenticated and it will have an order of “10”).
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type



- c. Specify the order “10” (we are lowering the order because we want to make sure we set the customer ID before some of the rules we are going to create later):

3. Create a new condition
 - a. Select the Core extension
 - b. Select the “Data Element” Condition Type

4. Set this Condition to check that the Data Element “Authentication State” has the following value: “logged out”

5. Create a new action
 - a. Select the “Experience Cloud ID Service” extension



- b. Select the “Set Customer IDs” Action Type

AGS Launch Demo >

AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension

Experience Cloud ID Service

Action Type

Set Customer IDs

Name

Experience Cloud ID Service - Set Custom..

6. Under “Integration Code”, set the “crm_id” with a value of your “Email (Hashed)” data element to the authenticated state

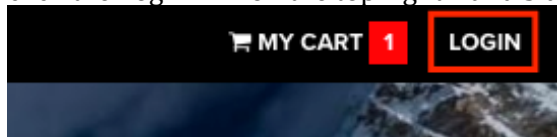
Integration Code ⓘ Value ⓘ Auth State

crm_id = %Email (Hashed)% Authenticated

Testing Considerations

To validate the setting of the customer ID:

1. click the Login link on the top right-hand side of the demo site:



2. Under the Login button, click the “new account” link
3. Create an account—you can use any email address (real or fake)
4. Click the Login link and log in
5. Return to the Homepage



6. If you inspect your demdex calls, you should see the `crm_id`, followed by the hashed email value, followed the authentication state of "1" (authenticated) at the end:

demdex

Query String Parameters

- d_visid_ver: 2.4.0
- d_fieldgroup: AAM
- d_rtbd: json
- d_ver: 2
- d_orgid: 1FD6776A524453CC0A490D44@AdobeOrg
- d_nsid: 0
- d_mid: 69919337458264615911242424200152831962
- d_blob: 6G1ynYcLPuiQxYZrsz_pkqfLG9yMXBpb2zX5dvJdYQJzPXImdj0y
- d_cid_lc: crm_id21232f297a57a5a743894a0e4a801fc31
- ts: 1511812818924

7. In the Debugger, it would look similar to this:

Marketing Cloud Visitor Service 13

Requests

Clear All Requests

1FD6776A524453CC0A490D44@AdobeOrg

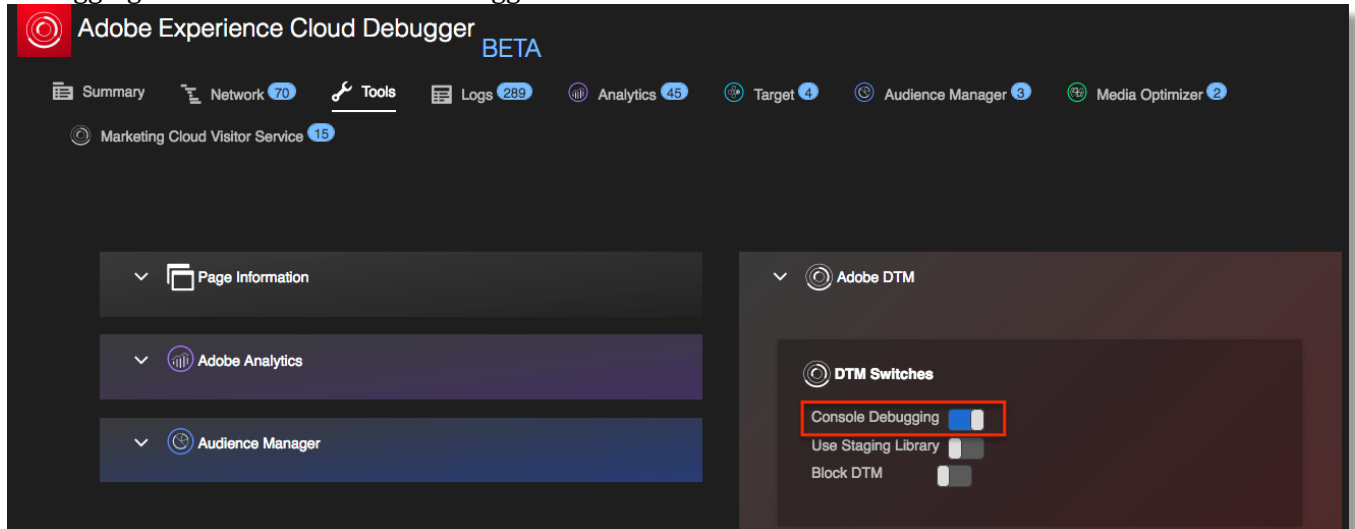
Property	Value
VisitorAPI.js version	2.4.0
Field Group	AAM
JSON Request	json
d_ver	2
Marketing Cloud Organization ID	1FD6776A524453CC0A490D44@AdobeOrg
d_nsid	0
Marketing Cloud Visitor ID	69919337458264615911242424200152831962
Audience Manager Blob	6G1ynYcLPuiQxYZrsz_pkqfLG9yMXBpb2zX5dvJdYQJzPXImdj0y
Customer ID	["crm_idu000121232f297a57a5a743894a0e4a801fc31"]
Timestamp	1511911919991
Customer ID - crm_id	21232f297a57a5a743894a0e4a801fc3-AUTH...

8. Note that you can confirm the hashed email value by viewing the source code of the page and looking at the username property:

```
87 "user": [  
88   {  
89     "profile": {  
90       {  
91         "profileInfo": {},  
92         "attributes": {  
93           "loggedIn": true,  
94           "username": "21232f297a57a5a743894a0e4a801fc3"  
95         }  
96       }  
97     }  
98   ],  
99 ]
```



9. Note that console logging can be activated for Launch in the same manner as DTM. You can run `_satellite.setDebug("true")` in your browser console, toggle it on with Launch Command, or turn on Console Debugging in the Tools tab of the Debugger:



10. Open your Developer Tools Console and you will see all of the Launch statements with the rocket icon in front of them:





Adobe Target Tagging & Configuration

Overview

This section demonstrates how to execute a robust Target implementation, including:

- Passing the Customer Id and Authentication State to enable CRM integrations such as Customer Attributes
- Parameters
 - Mbox parameters
 - Profile parameters
 - Entity parameters for Recommendations
- Order confirmation page mbox
- Custom mboxs
- Library Header and Library Footer alternatives

Prerequisites & Product-Specific Setup

- The Adobe Target extension has been implemented with default settings and the “All Pages – Top” rule (see Basic Setup guide on the [Launch by Adobe Reference Architectures landing page](#))
- The Experience Cloud Id Service extension has been implemented and Customer IDs are being set in “Library Loaded (Page Top)” rules for the Authenticated State as described in the previous section.

The following Data Elements are used by the Target Implementation—see [Appendix 1: Data Element definitions](#) for details on how each data element was defined:

- Page Name—useful for creating targeting conditions when building Target activities
- Authentication State—useful for creating targeting conditions when building Target activities
- Order Id—used to populate the orderId parameter in the order confirmation mbox
- Cart Amount—used to populate the orderTotal parameter in the order confirmation mbox
- Cart SKUs (Target)—used to populate the productPurchasedId parameter in the order confirmation mbox
- Product Category—used to populate the entity.categoryId parameter for the Recommendations catalog and to key Recommendations algorithms
- Product Description—used to populate the entity.message parameter for the Recommendations catalog
- Product Name—used to populate the entity.name parameter for the Recommendations catalog
- Product Path—used to populate the entity.pageUrl parameter for the Recommendations catalog
- Product Price—used to populate the entity.value parameter for the Recommendations catalog
- Product SKU (Target)—used to populate the entity.id parameter for the Recommendations catalog and to key Recommendations algorithms
- Product Thumbnail Path—used to populate the entity.thumbnailUrl parameter for the Recommendations catalog

Using Customer IDs in Target

When using the Experience Cloud ID service to enable integrations and core services such as Customer Attributes, it is imperative that the Customer ID is sent before firing the global mbox. To that end, make sure you have the following capabilities on your site:

- The customer ID must be available on the page before the Launch Embed Code
- The Experience Cloud ID Service extension must be installed
- You must use the “Set Customer IDs” action in a rule that fires at the “Library Loaded (Page Top)” event
- Use the “Fire global mbox” action in a rule that fires *after* the “Set Customer IDs” action



In the previous section, the “All Pages – Top – Authenticated - 10” rule was created to fire the “Set Customer ID” action. In the Basic Setup guide, we created a rule called “All Pages – Top” which loaded Target, set a `pageName` parameter, and fired the global mbox. Since that rule use the default “order” setting of “50”, it will fire after the “All Pages – Top – Authenticated - 10” rule, which is exactly what we want.

Parameters

Parameters passed through mbox calls greatly enhance the targeting and reporting capabilities of Target. The Launch extension provides two actions to pass parameters:

1. Add Params to All Mboxes—this action includes the parameters in all mbox calls made in the scope of the page load, e.g. additional mbox calls made from Custom Code actions or hardcoded on your site. It is equivalent to using the [targetPageParamsAll\(\)](#) method in `at.js`.
2. Add Params to Global Mbox—this action includes the parameters only in mbox calls using the global mbox name in the extension configuration (e.g. “target-global-mbox”). It is equivalent to using the [targetPageParams\(\)](#) method in `at.js`.

Since most implementations only use the target-global-mbox for activity delivery, it usually doesn’t matter which action you use to pass parameters.

Mbox parameters

Mbox parameters are available for targeting and measurement in the scope of the mbox call in which they are passed. They are ideal for attributes that change frequently such as the page name, page template, etc.

Just give the parameter a simple name and map it to the relevant data element. In the reference implementation, “page” and “authState” parameters are passed to all mboxs in the “All Pages – Top” rule:

Name		Value		
page	=	%Page Name%		
authState	=	%Authentication State%		

Profile Parameters

Profile parameters passed through mbox calls are written the visitor profile on Target’s backend database. They available for targeting and measurement for as long as the profile is active. They are ideal for attributes that rarely change or are only available on certain pages.

Just give the profile parameter a name prefixed with “profile.” and map it to the relevant data element. In the reference implementation, we are not using any profile parameters, but this is what it would look like to set a profile parameter called “userType”:

Name		Value		
profile.userType	=	%User Type%		

Entity Parameters for Recommendations

Entity parameters passed through mbox calls are used in Recommendations implementations for two main reasons:



1. As a key to trigger product recommendations. For example, when using a recommendations algorithm like “People who viewed Product X, also viewed Y,” “X” is the “key” of the recommendation. It is usually the product sku (entity.id) or category (entity.categoryId) that you are currently viewing.
2. To populate the Recommendations catalog. Recommendations contains a database of all of the products or articles on your website, the details of which are used in an mbox response serving recommendations. For example, when recommending products, you typically want to display attributes like the product name (entity.name) and image (entity.thumbnailUrl). Some customers populate their catalog using backend feeds, but they can also be populated using entity parameters in mbox calls.

Just give the profile parameter a name prefixed with “entity.” and map it to the relevant data element. Note that some common entities have reserved names that must be used (e.g. entity.id for the product sku).

In the reference implementation, we have a rule called “Product Details – Top” that fires only on product detail pages which fires on the “Library Loaded – Page Top” event. The rule has a condition to only fire when the “Product SKU (Target)” data element has a value, defined by the regular expression “([^\s])”:

We then use the “Add Params to Global Mbox” action to set entity parameters:

Name		Value			
entity.id	=	%Product SKU (Target)%		+	
entity.name	=	%Product Name%		+	
entity.pageUrl	=	%Product Path%		+	
entity.categoryId	=	%Product Category%		+	
entity.thumbnailUrl	=	%Product Thumbnail Path%		+	

In order to deliver recommendations without flicker, it is critical to pass the entity.id and entity.categoryId parameters at the top of the page, in the global mbox. Since we have a pretty good data layer, we can also pass entities to populate the catalog.

However, if some of the catalog entities are not available until lower on the page, it is possible to pass them through a separate mbox call made later in the page load. In the reference implementation, there is another rule called “Product Details – Bottom” which fires on the “Library Loaded – Page Bottom” event using the same condition as the “Product Details – Top” rule. In order to avoid complications with A4T, we want to give this mbox a different name, so we use the Custom Code action of the Core Extension to fire an mbox called “recs-entities” with the following code:

```
adobe.target.getOffer({
  "mbox": "recs-entities",
  "params": {
    "entity.id": _satellite.getVar('Product SKU (Target)'),
    "entity.value": _satellite.getVar('Product Price'),
    "entity.message": _satellite.getVar('Product Description')
  },
  "success": function(offer) {
  },
}
```






```
"error": function(status, error) {  
    console.log('Error', status, error);  
}  
});
```

Property Token for Enterprise User Permissions

The property token is a relatively new type of parameter in Target and is used with the Enterprise User Permissions feature available to Target Premium customers. It is used to identify different web properties to which different users should have different permissions. Target properties are analogous to Launch properties and Analytics report suites. An enterprise with multiple brands, websites, and marketing teams might have use a different Target property, Launch property, and Analytics report suite for each website. Whereas the embed codes differentiate Launch properties, and report suite ids differentiate Analytics report suites, the `at_property` parameter is used to differentiate Target properties.

Just name the parameter “`at_property`” and paste in the value provided in the Target interface. In the reference implementation, the “`at_property`” parameter is passed to all mboxs in our Launch property. Note: if you were using multiple Target properties in a single Launch property, you could manage the `at_property` value via a data element:

Name		Value	
at_property	=	68d2e235-84ee-7570-e9f7-86cea171741e	  

Order Confirmation mbox

The order confirmation mbox (typically named “`orderConfirmPage`”) is a special type of mbox used to define revenue metrics in Target. The inclusion of three specific mbox parameters—`orderId`, `orderTotal`, and `productPurchasedId`—is what turns an mbox into an order mbox. In addition to reporting revenue, the order mbox also does the following:

1. De-duplicates accidental order resubmissions
2. Filter extreme orders (any order whose total was more than three standard deviations from the mean)
3. Uses a different algorithm behind the scenes to calculate statistical confidence
4. Creates a special, downloadable Audit report of individual order details

We recommend that all Target customers with order funnels implement the order confirmation mbox, even on non-retail sites. For example, lead generation sites with lead funnels that end with a lead id being generated should implement an order mbox (just use “1” or some other number for the `orderTotal`). Customers using the Analytics for Target (A4T) integration for most of their reporting should also implement the order mbox, since A4T is not compatible with all activity types (e.g. Automated Personaization, Auto Allocate, and Auto Target). Additionally, this mbox is used to power Recommendations algorithms based on purchase behavior.

The order confirmation mbox should fire from a rule that is only triggered on your order confirmation page. Often, it can be combined in a rule that also sets the Adobe Analytics purchase event. It must be configured by using the Custom Code action of the Core extension, using the appropriate data elements to set the `orderId`, `orderTotal`, and `productPurchasedId` parameters.



In the Reference site, we use a rule called “Order Confirmation Page - Bottom – 30” (order=30 because it sets some Analytics variables that want to set before the Analytics “Send Beacon” action which fires in a rule with order=5) and fire the order mbox using custom code:

```
adobe.target.getOffer({
  "mbox": "orderConfirmPage",
  "params":{
    "orderId": _satellite.getVar('Order Id'),
    "orderTotal": _satellite.getVar('Cart Amount'),
    "productPurchasedId": _satellite.getVar('Cart SKUs (Target)')
  },
  "success": function(offer) {
    adobe.target.applyOffer( {
      "mbox": "orderConfirmPage",
      "offer": offer
    } );
  },
  "error": function(status, error) {
    console.log('Error', status, error);
  }
});
```

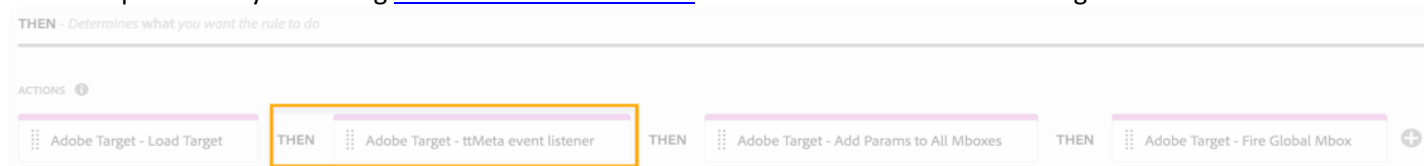
Custom mboxs

In the rare instance when you need to make mbox calls other than the global mbox, use the Custom Code action in the Core extension to an mbox, as described in the earlier sections [Entity Parameters for Recommendations](#) and [Order Confirmation mbox](#) which used [getOffer\(\)/applyOffer\(\)](#) and [trackEvent\(\)](#) methods. Just be sure to use the “Load Target” action before making mbox calls from custom code.

Library Header and Library Footer replacements

The Edit at.js screen in the Target user interface has locations in which you can paste custom JavaScript that will execute immediately before or after the at.js file. The Library Header is sometimes used to override at.js settings via the [targetGlobalSettings\(\)](#) function, while the Library Footer is sometimes used to add [at.js library extensions](#) or [at.js custom event](#) listeners.

To replicate this capability in Launch, just use the Custom Code action in the Core extension and sequence the action before (Library Header) or after (Library Footer) the Load Target action. The reference architecture replicates Library Footer capabilities by delivering [the ttMeta event listener](#) custom code after the Load Target action:

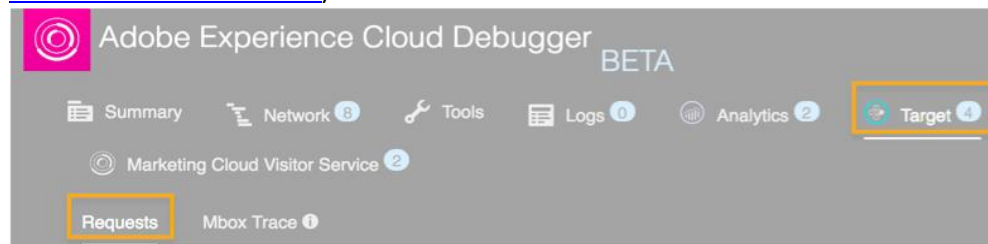




Testing Considerations

Using the Experience Cloud Debugger, we can quickly confirm the correct implementation of many of the Target concepts described above.

For example, open the Debugger's Target->Requests tab on one of the reference architecture product detail pages (<https://aem100-us.adobevelab.com/content/we-retail/us/en/products/women/shirts/devi-sleeveless-shirt.html#wootsudet-XXS>):



We can confirm the correct passing of the customer ids, mbox and entity parameters, plus the custom mbox:

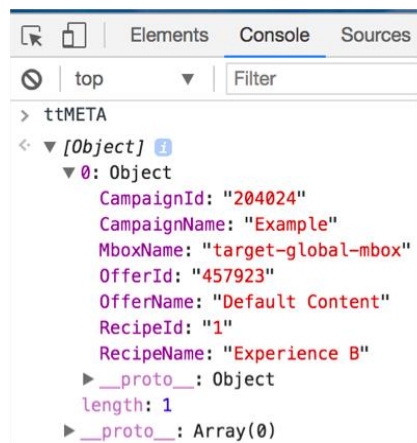
Requests		
Clear All Requests		
▼ agslaunchdemo		
Request Type	json	json
Analytics Visitor ID		
Marketing Cloud Visitor ID	78981395896656326371279784785836906234	78981395896656326371279784785836906234
Audience Manager Location Hint	9	9
Audience Manager Blob	RKhpRz8krg2tLO6pguXWp50kAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp50kAcUniQYPHa...
Supplemental Data ID	36117EE4F7EC1BAF-7BEFAB6EF7468209	36117EE4F7EC1BAF-7BEFAB6EF7468209
Mbox Name	target-global-mbox	recs-entities
Mbox Count	1	2
Mbox Session	3d976117060f4ddb8b69cc668d3b06ec	3d976117060f4ddb8b69cc668d3b06ec
Mbox PC	5be526e8ad44948e57cdc16ffa5d70.17_43	5be526e8ad44948e57cdc16ffa5d70.17_43
Mbox Host	or1010051030128.corp.adobe.com	or1010051030128.corp.adobe.com
Mbox Page	2a798a61a5cd4b6082d7a58d7f8948b2	2a798a61a5cd4b6082d7a58d7f8948b2
Mbox URL	http://or1010051030128.corp.adobe.com:4503...	http://or1010051030128.corp.adobe.com:4503...
Mbox Referrer	http://or1010051030128.corp.adobe.com:4503...	http://or1010051030128.corp.adobe.com:4503...
Mbox Time	1513801270539	1513801271694
Mbox Version	1.2.0	1.2.0
Screen Height	900	900
Screen Width	1440	1440
Browser Height	780	780
Browser Width	1404	1404
Browser Time Offset	-300	-300
Color Depth	24	24
Entity Param: id	wootsudet	
Entity Param: name	Devi+Sleeveless+Shirt	
Entity Param: pageUrl	/content/we-retail/us/en/products/women/shirts...	
Entity Param: categoryId	shirt	
Entity Param: thumbnailUrl	/content/we-retail/us/en/products/men/pants/d...	
Entity Param: value		45.00
Entity Param: message		Street-style+inspiration+with+performance+de...
Param: vat_crm_id	21232297a57a5a74a99d4a8d4a811c...	91999997a57a5a74a99d4a8d4a811c3
Param: vat_crm_id_authState	1	
Param: page	content/we-retail/us/en/products/women/shirts...	content/we-retail/us/en/products/women/shirts...
Param: authState	logged-in	



We can place a test order (first name and last name are the only required fields on the reference implementation) to verify that our order confirmation mbox is firing and with the required parameters:

Request Type	i	json	json
Analytics Visitor ID	i		
Marketing Cloud Visitor ID	i	78981395896656326371279784785836906234	78981395896656326371279784785836906234
Audience Manager Location Hint	i	9	9
Audience Manager Blob	i	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...
Supplemental Data ID	i	743A7E19A42AD11C-285692D2479DDA83	743A7E19A42AD11C-285692D2479DDA83
Mbox Name	i	target-global-mbox	orderConfirmPage
Mbox Count	i	1	2
Mbox Session	i	3d976117080f4ddb8b69cc668d3b06ec	3d976117080f4ddb8b69cc668d3b06ec
Mbox PC	i	5be526eec8ad44948e57cdc16ffa5d70.17_43	5be526eec8ad44948e57cdc16ffa5d70.17_43
Mbox Host	i	or1010051030128.corp.adobe.com	or1010051030128.corp.adobe.com
Mbox Page	i	1516eef405584306b8892c509d7cc119	1516eef405584306b8892c509d7cc119
Mbox URL	i	http://or1010051030128.corp.adobe.com:4503...	http://or1010051030128.corp.adobe.com:4503...
Mbox Referrer	i	http://or1010051030128.corp.adobe.com:4503...	http://or1010051030128.corp.adobe.com:4503...
Mbox Time	i	1513801913609	1513801913631
Mbox Version	i	1.2.0	1.2.0
Screen Height	i	900	900
Screen Width	i	1440	1440
Browser Height	i	780	780
Browser Width	i	1404	1404
Browser Time Offset	i	-300	-300
Color Depth	i	24	24
Param: vst.crm_id.id	i	21232f297a57a5a743894a0e4a801fc3	21232f297a57a5a743894a0e4a801fc3
Param: vst.crm_id.authState	i	1	1
Param: page	i	content:we-retail:us:en:user:checkout:order:th...	content:we-retail:us:en:user:checkout:order:th...
Param: authState	i	logged+in	logged+in
Param: orderId	i		7bf2fba4-c821-41b5-b9f1-d5e6069a9dce
Param: orderTotal	i		45
Param: productPurchasedId	i		wootsudet-xxs

Library header and footer equivalents will require different validation techniques depending on what you are doing in the Custom Code. The ttMeta event listener used in the reference architecture—when combined with Target’s [Response Token](#) capability and a live activity—will expose details of the activity when you run ttMETA in the browser console:



Known Issues or Limitations

None



Adobe Analytics Rule Creation

Overview

In the [Launch Reference Architecture Guide – Basic Setup](#) document, you implemented Adobe Analytics with a global page view beacon with the Page Name variable.

The Adobe Analytics extension supports client-side Analytics implementations using AppMeasurement.js. The Analytics extension is one of the few Adobe extensions which allows the customer to completely replace the underlying solution library with a custom variation to support unique tracking scenarios.

Prerequisites & Product-Specific Setup

Prior to continuing, ensure that the following steps have been completed:

1. The Adobe Analytics extension should be created and working as specified in the Basic Setup guide on the [Launch by Adobe Reference Architectures landing page](#).
 - a. If applicable, the edits made for the Audience Manager integration should also be completed
2. The “All Pages – Bottom” rule should be created and working as specified in the Basic Setup guide on the [Launch by Adobe Reference Architectures landing page](#).
3. Be prepared for validation. For the examples below, the tracking request made to Adobe servers will be reviewed, and the query strings containing tracking information will be examined. Screenshots provided will be of the Network tab in a Chrome browser’s developer tools, but the principle is the same for any debugging tool.

Rule Creation

The sections outlined below contain instructions for creating the rules which track user behavior on the reference site. A variety of rule types have been created to provide examples of different methods of tracking, as well as to demonstrate common events and actions that might be required.

User Sign-in – Sign in Start

This rule reports when a user begins the sign in process on the reference site.

1. Create a new rule called “Sign In – Start”
2. Create a new Event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type



- c. Name the new event "Core – Library Loaded (Page Top)"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Event Configuration

Extension
Core

Event Type
Library Loaded (Page Top)

Name
Core - Library Loaded (Page Top)

3. Create a new condition

- Select the Core extension
- Select the "Path Without Query String" Condition Type
- Name the new condition "Core – Path Without Query String"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Condition Configuration

Extension
Core

Condition Type
Path Without Query String

4. Set the Path field to the following string: /content/we-retail/us/en/community/signin.html

Path /content/we-retail/us/en/commur ☐ Regex

Add Pattern

5. Create a new action

- Select the "Adobe Analytics" extension
- Select the "Set Variables" Action Type



- c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension
Adobe Analytics

Action Type
Set Variables

Name
Adobe Analytics - Set Variables

6. Under “Events”, set event2:

Events

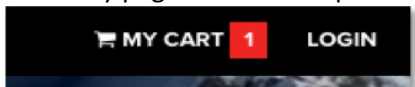
event2

Serialize from value (optional)

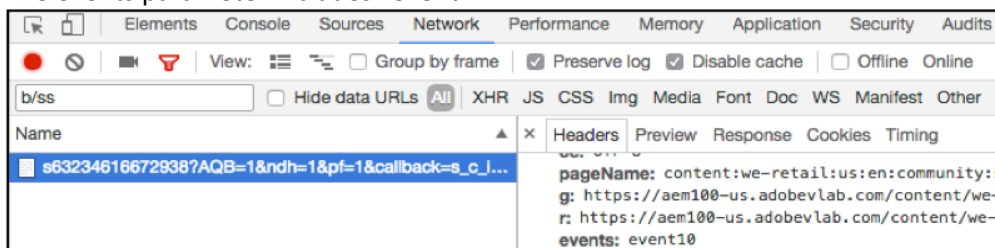
7. Save the new “Sign In – Start” rule

To validate this rule, take the following steps:

1. Ensure your debugger or the Network tab of your browser’s developer tools is open. Click the “preserve log” option, if available.
2. From any page on the example site, find the “Login” link in the upper right corner of the page and click it:



3. In your debugger, verify that an image request was made to the Adobe tracking server.
4. In the query parameters of that image request, verify the following values:
 - a. The `pageName` parameter is set to “content:we-retail:us:en:community:signin”
 - b. The `events` parameter includes “event2”



User Sign-in – Sign in Confirmation

7. Create a new rule called “Sign In”
8. Create a new event
 - a. Select the Core extension



- b. Select the “Data Element Change” event type
- c. Name the new event "Core – Data Element Change"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Event Configuration

Extension
Core

Event Type
Data Element Change

Name
Core - Data Element Change

9. Create a new condition
- a. Select the Core extension
 - b. Select the “Data Element” Condition Type
 - c. Name the new condition “Core – Data Element”

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extension

Condition Configuration

Extension
Core

Condition Type
Data Element

Name
Core - Data Element

10. Set this Condition to check that the Data Element “Authentication State” has the following value: “logged out”

Data element Authentication State has the value logged out

☐ Regex

11. Create a new action
- a. Select the “Adobe Analytics” extension
 - b. Select the “Set Variables” Action Type



- c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension
Adobe Analytics

Action Type
Set Variables

Name
Adobe Analytics - Set Variables

12. Under “Events”, set event3:

Events

event3

Serialize from value (optional)

13. Create a (second) new action

- Select the “Adobe Analytics” extension
- Select the “Send Beacon” action type
- Name the new action “Adobe Analytics – Send Beacon”

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Action Configuration

Extension
Adobe Analytics

Action Type
Send Beacon

14. Under “Tracking”:

- Select “s.tl()”
- Set Link Type to “Custom Link”



- c. Set Link Name to “signin”:

Tracking

☐ s.t(): Send data to Adobe Analytics and treat it as a page view

☒ s.tl(): Send data to Adobe Analytics and do not treat it as a page view

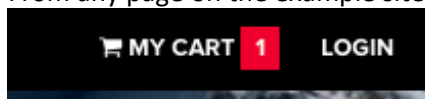
Link Type Link Name

Custom Link signIn

15. Save the new “Sign In” rule

To validate this rule, take the following steps:

1. Ensure your debugger or the Network tab of your browser’s developer tools is open. Click the “preserve log” option, if available.
2. From any page on the example site, find the “Login” link in the upper right corner of the page and click it:



3. Sign in to the site (see [Appendix: Test Account](#) for information on creating an account)
4. When the following page loads, in your debugger, verify that an image request was made to the Adobe tracking server.
5. In the query parameters of that image request, verify the following values:
 - a. The events parameter includes “event3”

Product Details

For Product Details pages, two rules are required: one at the page top, and one at the bottom. The “Product Details - Bottom” rule is used for Target deployments (see above).

For Analytics, the “Product Details – Top - 20” rule is configured as follows.

1. Create a new Rule, called “Product Details - Top - 20”
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type
 - c. Name the new event “Core – Library Loaded (Page Top)”



- d. Set Order to 20

AGS Launch Demo >

AEM 6.3 Demo

Overview Rules Data Elements

Event Configuration

Extension

Core

Event Type

Library Loaded (Page Top)

Name

Core - Library Loaded (Page Top)

Order

20

- 3. Create a new condition
 - a. Select the Core extension
 - b. Select the “Data Element” Condition Type
 - c. Name the new condition “Core – Data Element”:

AGS Launch Demo >

AEM 6.3 Demo

Overview Rules Data Elements Extension

Condition Configuration

Extension

Core

Condition Type

Data Element

Name

Core - Data Element

- 4. Set this condition as follows:
 - a. Check that the Data Element “Product SKU (Target) has the following value: ([^\s])



- b. Enable the Regex flag

Data element has the value

☒ Regex [Test](#)

5. Create a new action

- a. Select the “Adobe Analytics” extension
- b. Select the “Set Variables” Action Type
- c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo >
AEM 6.3 Demo

Overview **Rules** Data Elements Extensions

Action Configuration

Extension
 ▼

Action Type
 ▼

Name

6. In the Custom Code editor of the Action, enter the following code:

```
s.products=_satellite.getVar("Product SKU (Analytics)");
```

(see the Appendix entry for [Product information \(product details pages\)](#) for details on configuring this data element)

7. Under “Events”, set prodView:

Events

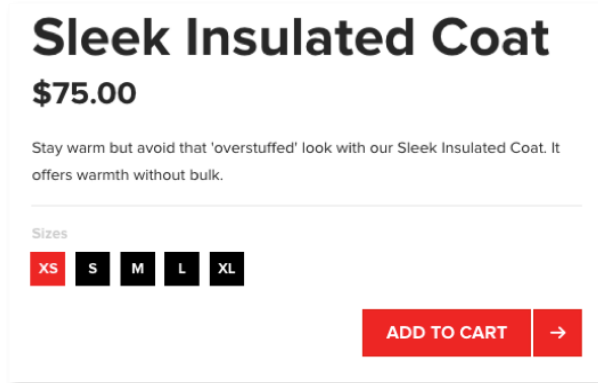
▼

8. Save the new “Product Details – Top” rule.

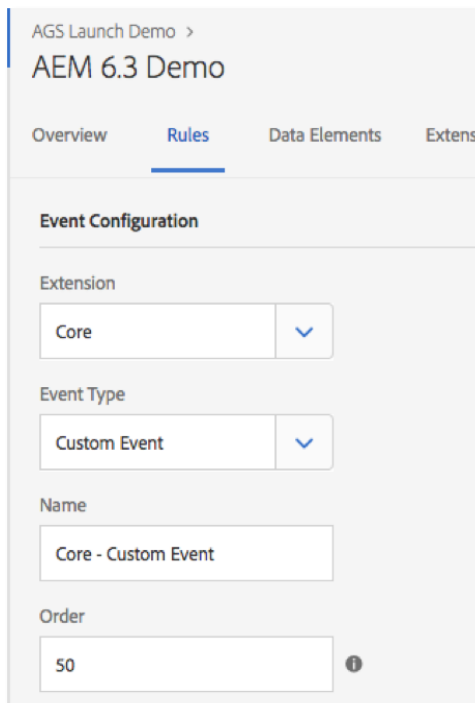
To validate this rule, follow the steps in the [Rule Validation: Product, Cart, and Checkout Flow](#) section below.

Shopping Cart Adds

This rule assumes a custom JavaScript event, “cart-add”, has been created and executes on the submit action of the add-to-cart process. Adding a rule to the click of an add-to-cart button or form may result in less accurate reporting than using an event connected with the CMS or ecommerce platform.



1. Create a new Rule called "Cart Adds"
2. Create a new event
 - a. Select the Core extension
 - b. Select the "Custom Event" event type
 - c. Name the new event "Core – Custom Event"



3. In the Event Configuration:
 - a. Set Custom Event Type to "cart-add"
 - b. Select the "specific elements" option



- c. Set the CSS selector to “body”

Custom Event Type ⓘ

☒ specific elements ☐ any element

Elements matching the CSS selector ⓘ

☐ and having certain property values...

> Advanced

4. Create a new action

- Select the “Adobe Analytics” extension
- Select the “Set Variables” Action Type
- Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo >

AEM 6.3 Demo

Overview **Rules** Data Elements Extensions

Action Configuration

Extension

▼

Action Type

▼

Name

5. Create a new action

- Select the “Adobe Analytics” extension
- Select the “Send Beacon” Action Type
- Name the new action “Adobe Analytics – Send Beacon”

Overview **Rules** Data Elements

Action Configuration

Extension

▼

Action Type

▼

Name

6. Under Tracking:



- a. Select s.tl()
- b. Set Link Type to “Custom Link”
- c. Set Link Name to “scAdd”:

Tracking

☐ s.t(): Send data to Adobe Analytics and treat it as a page view

☒ s.tl(): Send data to Adobe Analytics and do not treat it as a page view

Link Type Link Name

Custom Link scAdd

To validate this rule, follow the steps in the [Rule Validation: Product, Cart, and Checkout Flow](#) section below.

Shopping Cart Removes

This rule assumes a custom JavaScript event, “cart-remove”, has been created and executes on the submit action of the add-to-cart process. Adding a rule to the click of an add-to-cart button or form may result in less accurate reporting than using an event connected with the CMS or ecommerce platform

1. Create a new Rule called "Cart Removes"
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Custom Event” event type
 - c. Name the new event "Core – Custom Event"

AGS Launch Demo >

AEM 6.3 Demo

Overview Rules Data Elements Extensions

Event Configuration

Extension

Core

Event Type

Custom Event

Name

Core - Custom Event

Order

50

3. In the Event Configuration:
 - a. Set Custom Event Type to “cart-remove”



- b. Select the “specific elements” option
- c. Set the CSS selector to “body”.

Custom Event Type ⓘ

☒ specific elements ☐ any element

Elements matching the CSS selector ⓘ

☐ and having certain property values...

> Advanced

- 4. Create a new action
 - a. Select the “Adobe Analytics” extension
 - b. Select the “Set Variables” Action Type
 - c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo >

AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension
 ▼

Action Type
 ▼

Name

- d. Add “scRemove” to the Events section:

Events

<input data-bbox="269 1486 521 1528" type="text" value="scRemove"/> ▼	Serialize from value (optional) <input data-bbox="867 1486 1101 1539" type="text"/>	⌵	×
<input data-bbox="269 1581 521 1623" type="text" value="Select event"/> ▼	Serialize from value (optional) <input data-bbox="867 1581 1101 1633" type="text"/>	⌵	×

- 5. Create a (second) new action
 - a. Select the “Adobe Analytics” extension
 - b. Select the “Send Beacon” Action Type



- c. Name the new action “Adobe Analytics – Send Beacon”

Overview Rules Data Elements

Action Configuration

Extension
Adobe Analytics

Action Type
Send Beacon

Name
Adobe Analytics - Send Beacon

- d. Under Tracking, select `s.tl()`, set Link Type to “Custom Link”, and set Link Name to “scRemove”:

Tracking

☐ `s.t()`: Send data to Adobe Analytics and treat it as a page view

☒ `s.tl()`: Send data to Adobe Analytics and *do not* treat it as a page view

Link Type Link Name

Custom Link scRemove

To validate this rule, follow the steps in the [Rule Validation: Product, Cart, and Checkout Flow](#) section below.

Checkout Initiation

1. Create a new rule called “Checkout Page - Top”
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type



- c. Name the new event "Core – Library Loaded (Page Top)"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Event Configuration

Extension
Core

Event Type
Library Loaded (Page Top)

Name
Core - Library Loaded (Page Top)

3. Create a new condition

- a. Select the Core extension
- b. Select the "Path Without Query String" Condition Type
- c. Name the new condition "Core – Path Without Query String"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Condition Configuration

Extension
Core

Condition Type
Path Without Query String

- d. Set the Path field to the following string: checkout.html

Path checkout.html ☐ Regex Test

Add Pattern

4. Create a new action

- a. Select the "Adobe Analytics" extension
- b. Select the "Set Variables" Action Type



- c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo > AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension

Adobe Analytics

Action Type

Set Variables

Name

Adobe Analytics - Set Variables

5. Under “Events”, set scCheckout:

Events

scCheckout

Serialize from value (optional)

6. Save the new “Checkout Page - Top” rule

To validate this rule, follow the steps in the [Rule Validation: Product, Cart, and Checkout Flow](#) section below.

Order Confirmation

1. Create a new rule called “Order Confirmation Page - Bottom - 30 ”
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type
 - c. Name the new event "Core – Library Loaded (Page Top)".



- d. Set the Order to 30.

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Event Configuration

Extension
Core

Event Type
Page Bottom

Name
Core - Page Bottom

Order
30

3. Create a new condition

- Select the Core extension
- Select the "Path Without Query String" Condition Type
- Name the new condition "Core – Path Without Query String"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements

Condition Configuration

Extension
Core

Condition Type
Path Without Query String

- d. Set the Path field to the following string: thank-you.html.html

Path ☒ Regex Test

Add Pattern

4. Create a new action

- Select the "Adobe Analytics" extension
- Select the "Set Variables" Action Type



- c. Name the new action “Adobe Analytics - Set Variables”

AGS Launch Demo > AEM 6.3 Demo

Overview Rules Data Elements Extensions

Action Configuration

Extension

Adobe Analytics

Action Type

Set Variables

Name

Adobe Analytics - Set Variables

- d. Under “Events”, set “purchase”:

Events

purchase

Serialize from value (optional)

5. Save the new “Order Confirmation Page - Top” rule

To validate this rule, follow the steps in the [Rule Validation: Product, Cart, and Checkout Flow](#) section below.

Site Search

1. Create a new rule named “Search Results – Top”
2. Create a new event
 - a. Select the Core extension
 - b. Select the “Library Loaded (Page Top)” event type



- c. Name the new event "Core – Library Loaded (Page Top)"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Event Configuration

Extension
Core

Event Type
Library Loaded (Page Top)

Name
Core - Library Loaded (Page Top)

3. Create a new condition
- Select the Core extension
 - Select the "Query String Parameter" condition type
 - Name the new condition "Core - Query String Parameter"

AGS Launch Demo >
AEM 6.3 Demo

Overview Rules Data Elements Extensions

Condition Configuration

Extension
Core

Condition Type
Query String Parameter

Name
Core - Query String Parameter

- d. Set this Condition as follows:
- Check that the URL Parameter Name "searchText" has the Regex wildcard "." (to match any single character)
 - Enable the Regex flag:

URL Parameter Name searchText URL Parameter Value . ☒ Regex Test

4. Create a new action
- Select the "Adobe Analytics" extension
 - Select the "Set Variables" Action Type



- c. Name the new action “Adobe Analytics - Set Variables”
- d. In the new action, set eVar2 to %search term%:

eVars

eVar2 ▼ Set as ▼ %search term%

5. Save the “Search Results – Top” rule.

To validate this rule:

1. Ensure your debugger or the Network tab of your browser’s developer tools is open. Click the “preserve log” option.
2. Click the Search icon, located near the upper right of any page:



3. Enter a search term, and press return to go to the search results page:



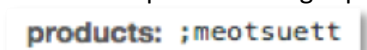
4. In your debugger, verify that an image request was made to the Adobe tracking server.
5. In the query parameters of that image request, verify the following values:
 - a. The events parameter includes “event2”
 - b. The eVar2 parameter includes the search term you entered. In the example above, eVar2 should be set to “jacket”.

Rule Validation: Product, Cart, and Checkout flow

The steps below validate each step of the product purchase flow rules defined above. Before beginning the validation process, ensure your debugger or the Network tab of your browser’s developer tools is open. Click the “preserve log” option, if available.

Product Details Page:

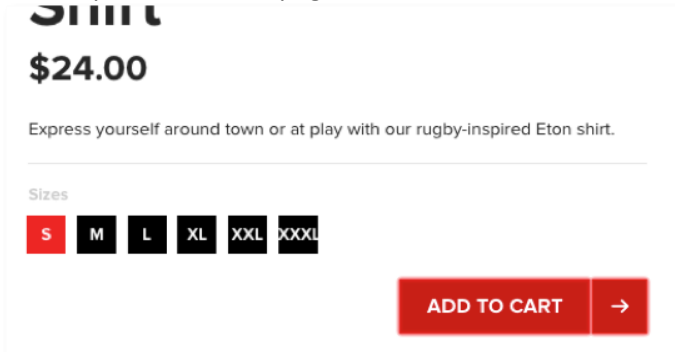
1. Browse to a product details page, or use the following example page:
<https://aem100-us.adobeavl.com/content/we-retail/us/en/products/men/shirts/eton-short-sleeve-shirt.html#meotsuett-S>
2. In your debugger, verify that an image request was made to the Adobe tracking server. In the query parameters of that image request, verify the following values:
 - a. The events parameter includes “prodView”
 - b. The correct product string is present. For the example page above, that value should be “;meotsuett”





Cart Adds:

1. From a product details page, click the Add To Cart button



2. In your debugger, verify that an image request was made to the Adobe tracking server.
3. In the query parameters of that image request, verify the following values:
 - a. The events parameter includes “scAdd”
 - b. The correct product string is present. For the example page above, that value should be “;meotsuett”

`products: ;meotsuett`

Cart Removes:

1. Assuming the item from the previous step has been added to your cart, view your shopping cart by clicking the “My Cart” link in the upper right of the page:



2. In the cart slide-out widget, click the “x” icon next to the product added above.

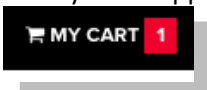


3. In your debugger, verify that an image request was made to the Adobe tracking server.
4. In the query parameters of that image request, verify the following values:
 - a. The events parameter includes “scRemove”
 - b. The correct product string is present. For the example page above, that value should be “;meotsuett”

`products: ;meotsuett`

Checkout Initiation:

1. Repeat the steps above to add a product to the shopping cart.
2. View your shopping cart by clicking the “My Cart” link in the upper right of the page:



... or visit the following link:

<https://aem100-us.adobevelab.com/content/we-retail/us/en/user/cart.html>



- Click the “check out” button:



- Upon reaching the checkout page, in your debugger, verify that an image request was made to the Adobe tracking server.
- In the query parameters of that image request, verify the following values:
 - The events parameter includes “scCheckout”
 - The correct product string is present. For the example page above, and assuming a quantity of 1 item, that value should be “;meotsuett-S”

```
events: scCheckout,event10  
products: ;meotsuett-S
```

Order Confirmation:

- From the Checkout page (link below), enter address and payment information. This demonstration site accepts fake information, as referenced in [Appendix: Test Account Information](#). (please don't submit personal data). For example:

Shipping Address	Shipping method
TEST	<input checked="" type="radio"/> Standard Shipping: 5-14 business days - \$5.00
ORDER	<input type="radio"/> Ground Shipping: 3-7 business days - \$10.00
123 Fake ST	<input type="radio"/> Priority Shipping: 2 business days - \$15.00
Address line 2	<input type="radio"/> Express Shipping: 1 business day - \$25.00
Springfield	Payment method
75081	<input checked="" type="radio"/> Credit or Debit card
Uruguay	<input type="radio"/> PayPal
	4111111111111111
	2/20
	123



NOTE: Please do not send any personal information to the demonstration site.

- Once information has been entered, click Continue.
- On the Review Order page, click “Place Order”.
- Upon reaching the order confirmation page, in your debugger, verify that an image request was made to the Adobe tracking server.
- In the query parameters of that image request, verify the following values:
 - The events parameter includes “purchase”
- The correct product string is present, including units and price. For the example product used here, and assuming a quantity of 1 item, that value should be “;meotsuett-S;1;24”

```
events: purchase,event10  
products: ;meotsuett-S;1;24
```



Adobe Audience Manager (AAM) Tagging & Configuration

Overview

There are two ways to implement AAM code, server-side forwarding and client-side DIL.

Server-Side Forwarding (SSF) – If clients have Adobe Analytics this solution forwards Adobe Analytics data to AAM on the back end, allowing for one less pixel on the page. This also enables key integration features, and conforms with our best practices for AAM code implementation and deployment.

Client-Side DIL – This solution covers anyone who does not have Adobe Analytics. DIL code (Data Integration Library Code) sends data directly from the web page into AAM. This is also utilized for clients who have Google Analytics.

Server-side forwarding is implemented on this demo site because it also has Adobe Analytics. Detailed steps to implement and validate server-side forwarding can be found in the Basic Setup guide available on the [Launch by Adobe Reference Architectures landing page](#). Client-side implementation options are in the Appendix of this document, in case that better represents your environment.



Appendix: Data Layer Reference

This section contains examples of data layer elements used by this reference architecture, which are based on the [W3C data layer standard](#) (PDF).

Page and content identification:

```
window.digitalData = {
  "page": {
    "pageInfo": {
      "breadcrumbs": "we-retail:us:en",
      "pageShortName": "we-retail:[...]:en",
      "pageTitle": "content:we-retail:us:en",
      "destinationURL": "http://or1010051030128.corp.adobe.com/content/we-
retail/us/en.html",
      "isIframe": false,
      "contentIframe": false,
      "hierarchiel": "we-retail:us:en",
      "title": "English",
      "internalPageName": "en",
      "pageID": "or1010051030128.corp.adobe.com:content:we-retail:us:en",
      "tagging": "",
      "server": "or1010051030128.corp.adobe.com",
      "urlShortcut": ""
    },
    "category": {
      "type": ":conf:we-retail:settings:wcm:templates:hero-page",
      "version": "2016-6-29"
    },
    "attributes": {},
    "components": {}
  }
}
```

Product information (product details pages)

```
window.digitalData = {
  "product": [
    {
      "productInfo": {
        "sku": "mehiwiext",
        "title": "Expedition Tech Long-Sleeved Shirt",
        "description": "shirt"
      }
    }
  ],
}
```




Cart information and contents:

```
window.digitalData = {
  "cart": {
    "productsInCart": 2,
    "orderId": "7e9daf14-cb8a-45eb-8206-99aed1a7026a",
    "cartAmount": "194",
    "cartEntries": [
      {
        "qty": "1",
        "sku": "meskwilt.1-XS",
        "title": "El Gordo Down Jacket Green",
        "formattedPrice": "$119.00",
        "price": "119"
      },
      {
        "qty": "1",
        "sku": "woskwilt-XS",
        "title": "Sleek Insulated Coat",
        "formattedPrice": "$75.00",
        "price": "75"
      }
    ]
  }
},
```

User information and status:

```
window.digitalData = {
  "user": [
    {
      "profile": [
        {
          "profileInfo": {},
          "attributes": {
            "loggedIn": true,
            "username": "21232f297a57a5a743894a0e4a801fc3"
          }
        }
      ]
    }
  ]
},
```



Appendix: Data Elements Reference

This appendix contains all of the data element definitions used in the Reference implementation. For details on how to create Data Elements in Launch, see the Basic Setup guide.

Data Element Name	Authentication State
Extension	Core
Data Element Type	Custom Code
Default Value	N/A
Force lowercase value	No
Clean text	No
Duration	Pageview
Custom Code	<pre>if (digitalData.user[0].profile[0].attributes.loggedIn) return "logged in" else return "logged out"</pre>

Data Element Name	Cart Amount
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	No
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.cart.cartAmount

Data Element Name	Cart SKUs (Target)
Extension	Core
Data Element Type	Custom Code



Default Value	N/A
Force lowercase value	Yes
Clean text	Yes
Duration	Pageview
Custom Code	<pre>var targetProdSkus=""; for (var i=0; i<digitalData.cart.cartEntries.length; i++) { if(i>0) { targetProdSkus = targetProdSkus + ","; } targetProdSkus = targetProdSkus + digitalData.cart.cartEntries[i].sku; //for debugging purposes, to run for each loop _satellite.notify(targetProdSkus,1); } return targetProdSkus;</pre>

Data Element Name	Email (Hashed)
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	No
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.user.0.profile.0.attributes.username

Data Element Name	Order Id
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A



Force lowercase value	No
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.cart.orderId

Data Element Name	Page Name
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	Yes
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.page.pageInfo.pageName

Data Element Name	Product Category
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	No
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.product.0.productInfo.description

Data Element Name	Product Description
-------------------	---------------------



Extension	Core
Data Element Type	DOM Attribute
Default Value	N/A
Force lowercase value	No
Clean text	Yes
Duration	Pageview
CSS Selector	.we-Product-description
Insert the value of	text

Data Element Name	Product Name
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	No
Clean text	Yes
Duration	Pageview
Path to variable	digitalData.product.0.productInfo.title

Data Element Name	Product Path
Extension	Core
Data Element Type	Page Info
Default Value	N/A
Force lowercase value	Yes
Clean text	No



Duration	Pageview
Attribute	Pathname

Data Element Name	Product Price
Extension	Core
Data Element Type	Custom Code
Default Value	N/A
Force lowercase value	No
Clean text	No
Duration	Pageview
Custom Code	<code>return document.querySelectorAll(".we-Product-price")[0].textContent.slice(1);</code>

Data Element Name	Product SKU (Target)
Extension	Core
Data Element Type	JavaScript Variable
Default Value	N/A
Force lowercase value	Yes
Clean text	Yes
Duration	Pageview
Path to variable	<code>digitalData.product.0.productInfo.sku</code>

Data Element Name	Product Thumbnail Path
Extension	Core
Data Element Type	Custom Code



Default Value	N/A
Force lowercase value	No
Clean text	No
Duration	Pageview
Custom Code	<pre>return "/content/we-retail/us/en/products/men/pants/" + digitalData.page.pageInfo.internalPageName + "/jcr:content/root/product/image.thumbnail.160.png/1473680874024.png";</pre>

Appendix: Test Account Information

User account:

To create a user account on this reference site:

1. Click on the “login” link in the upper right, and then click the “new account” link, or go directly to the Sign Up page:
<https://aem100-us.adobevelab.com/content/we-retail/us/en/community/signup.html>
2. Enter test account information on the following page. An email address validation is not performed, so a test email address will work here.

Order and address information:

This demo site allows for fake data to be entered on the Payment Information page for the purposes of sending test orders. All that needs to be entered is a first and last name. Any text can be entered.



NOTE: Please do not send any personal information to the demonstration site.