



Launch by Adobe

Reference Architecture Guide for Single-Page Applications
(AngularJS)

by Adobe Customer Solutions

Version 1.0/February 21, 2018



Table of Contents

Reference Architecture Overview.....	4
Platform Scope	4
Other Launch Reference Architectures	4
Guide Version History:.....	4
Goal.....	4
Assumptions.....	4
Single-page Application Instructions & Setup	5
Load Launch Library Synchronously.....	5
Data Layer Recommendations.....	5
Data Layer Implementation	5
Data Layer Location.....	5
Updating Data Layer	5
Data Layer Events	5
AngularJS Instructions & Setup for Adobe Target and Analytics.....	6
AngularJS Environment	6
Page Load and View Change in SPA.....	6
Angular Event \$routeChangeSuccess	7
Dispatching Custom Event for View Start	7
Dispatching Custom Event for View End.....	8
Dispatching Custom Event for User Actions.....	8
General Launch Configuration & Settings.....	9
Rule Configuration	9
1. app load: initial app load	9
2. page load: event-view-start.....	10
3. page load: event-view-end	12
Adobe Target Configuration & Tagging.....	15
Overview.....	15
Prerequisites & Product-Specific Setup.....	15
Rule Configuration	15
Add Actions.....	15
Load Target Action.....	16



Add Custom Event Rule	17
Adobe Analytics Configuration & Tagging.....	20
Overview.....	20
Prerequisites & Product-Specific Setup.....	20
Extension Setup	20
Additional Configuration	20
Rule Configuration	20
Add Actions.....	20
Implementation Validation.....	25
Integrations.....	26
Integration Consideration 1: Analytics for Target (A4T).....	26
Integration Consideration 2: Visitor resetState Method.....	26
Integration Consideration 3: Visual Experience Composer (VEC).....	26
Experience Cloud ID Tagging & Configuration.....	27
Adobe Audience Manager (AAM) Tagging & Configuration	28
Overview.....	28
Server-side or Client-side Instructions:.....	28
Server-side forwarding.....	28
Client-side	28
Prerequisites & Product-Specific Setup – AAM Only – Client Side.....	28
Which Extension Do I Configure in Launch?	28
Build your Library.....	33
Publish your implementation of AAM DIL.....	33
Testing Considerations & Steps	33
To Validate AAM Only on Your Property.....	34
To See AAM Only on the AAM Only - SPA Demo Site.....	34



Reference Architecture Overview

Platform Scope

This Reference Architecture Guide contains Single Page Application setup best practices with focus on AngularJS framework for implementing the Adobe Experience Cloud via Adobe Launch. This guide provides integration steps for AngularJS version 1.x. As a companion to this guide, a reference implementation using these best practices can be found at:

<http://agslaunchdemo.businesscatalyst.com/angularjs/>

Other Launch Reference Architectures

[Additional Reference Implementation Guides](#) will show you how to:

1. Implement the Experience Cloud in a standard website (pre-requisite for this AngularJS Guide)
2. Implement more robust features of the Experience Cloud in a standard website
3. Implement the Experience Cloud in a Video player
4. Implement the Experience Cloud in Accelerated Mobile Pages (AMP)
5. Implement the Experience Cloud in Single Page Applications (SPAs) using the ReactJS framework

Guide Version History:

v1.0 February 21, 2018 – Initial version

Goal

The goal of this document is to aid you in configuring Launch to deploy the Adobe Experience Cloud ID Service, Adobe Target, Adobe Analytics, and Adobe Audience Manager in your AngularJS application.

Assumptions

This guide assumes:

- That you have working knowledge of the Adobe solutions you are implementing (i.e. Adobe Analytics, Adobe Target, Adobe Audience Manager). A technical background is helpful, but not required.
- Your Marketing Cloud Organization has been provisioned for Launch and you have full access.
- You are familiar with Launch by Adobe and already know how to deploy extensions, rules, data elements and publishing tags. It is assumed that you have already reviewed the basic website implementation guide located [here](#), and have configured:
 - Adapters
 - Environments
 - Adobe Target Extension
- You have an AngularJS environment up and running
- You have added Adobe tools for debugging and Quality Assurance (QA) as mentioned in Basic Setup guide.



Single-page Application Instructions & Setup

Load Launch Library Synchronously

Once you complete this section, you should have the Launch Library up and running on your website, you can view the source code of this page to see an example placement of the tags in the HTML document.

<http://agslaunchdemo.businesscatalyst.com/angularjs/>

Data Layer Recommendations

A data layer is a JavaScript object which is used to store page/visitor information and can be shared with 3rd party tools and applications running on your website. The information from the data layer can be read using Adobe Launch and then sent to Adobe Marketing Cloud tools/services and other 3rd party tools that are installed via Launch extensions.

A data layer is a requirement for this implementation and Adobe Customer Solutions strongly encourages you to leverage one in any of its tag management implementations.

Data Layer Implementation

Adobe Launch is compatible with any data layer schema. If you have not already implemented a data layer or are unfamiliar with them, Adobe Customer Solutions recommends the W3C standard. See example at <https://www.w3.org/2013/12/ceddl-201312.pdf>.

Data Layer Location

If you have any basic data layer structure and data that you wish to load during the initial page load, it is recommended that you load this before the Launch Library Header. This ensures that it is readily available for Launch when it executes the Library Loaded event or Page Top event rules.

Updating Data Layer

It is recommended to keep your Data Layer updated whenever a state change happens on your SPA page. Any information that you wish to relay to a 3rd party tool during the state change can be made available in the Data Layer. This includes page information, visitor information, event information that was triggered by the page or the user etc. All this information could be valuable inputs for your Analytics or Optimization tool.

Data Layer Events

We must add a minimum of three custom events to efficiently communicate state/view changes in a SPA application.

event-view-start: This event should fire on the view start of the view/state that is loading. All data layer variables needed for testing and optimization tools should be loaded or updated before this event. Every view change should be considered as a start of a new view. For a better user experience (minimal to no flicker), this event should be fired as early as possible once the data layer has been updated with new data for the new view.

event-view-end: This event should fire once the actual view/state has loaded and is ready for user interaction. This event should be fired even when a view/state change has occurred and all SPA components on the page have finished loading.



event-action-trigger: This event should fire when any event occurs on the page except view/state load. This could be a click event or a state change without a view change.

AngularJS Instructions & Setup for Adobe Target and Analytics

AngularJS Environment

The demo used in this example is using AngularJS version 1.2 and should work the same on all 1.x versions. Angular 2+ requires a different implementation technically but the idea remains the same – firing custom events on view start, view end, and user actions.

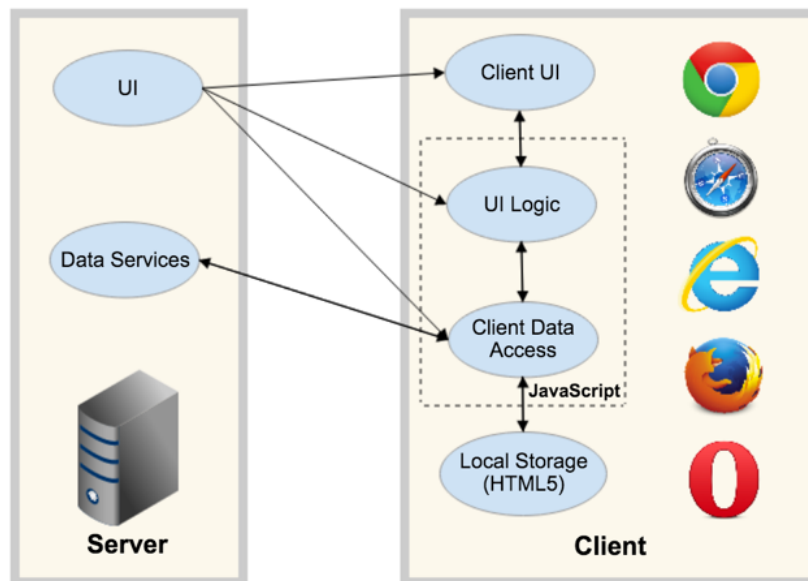
Refer to a fully functional demo for this guide at: <http://agslaunchdemo.businesscatalyst.com/angularjs/>.

Instructions below provide detailed step by step integration process for the above demo.

Page Load and View Change in SPA

The concept of “view change” in a Single Page Application is replacing the concept of page load. In SPA, we never reload the page to go to another one, but rather change the view, also called partial in Angular world. Therefore, view change needs to carry the same rules as page load.

Picture: Single Page Application Flow Diagram



In our demo as you browse the source code, Data Layer and Launch library are initially integrated in HTML head.

```
<script>
  digitalData = {
    pageInstanceID: "Angular-Demo-Localhost",
    page:{
      pageInfo:{
        pageID: document.title,
        pageName: 'angular-demo:'+document.title,
        destinationURL: encodeURIComponent(window.location.href)
      }
    }
  }
</script>
```



```

    },
    category:{
      primaryCategory: "Angular Demo - Localhost",
      subCategory1: "Angular for Launch",
      pageType: "Basic Demo"
    },
    attributes:{
      project: "Launch"
    }
  }
};
</script>
<script src="//assets.adobedtm.com/launch-EN843e9605001f429e9cfed58fa737f5cf.min.js"></script>

```

Angular Event \$routeChangeSuccess

On initial page load or when the view changes in your SPA, one of the first things that must happen is Data Layer update. This sets new information about the page on top of view before sending it to Target and Analytics.

In AngularJS, one convenient place to do Data Layer update is an Angular event **\$routeChangeSuccess** ([https://docs.angularjs.org/api/ngRoute/service/\\$route#event-\\$routeChangeSuccess](https://docs.angularjs.org/api/ngRoute/service/$route#event-$routeChangeSuccess)). This event is broadcasted after a route change has happened successfully and we can safely update page info. Here is an example:

```

$rootScope.$on('$routeChangeSuccess', function() {
  // Update Data Layer
  digitalData.page.pageInfo.pageID = $route.current.title;
  digitalData.page.pageInfo.pageName = 'angular-demo:'+$route.current.title;
  digitalData.page.pageInfo.destinationURL = encodeURIComponent(window.location.href);
  //...
});

```

Dispatching Custom Event for View Start

When Data Layer is updated at top of the view, we now can send a Target request. The reason for request to be on top of the view is that response will likely deliver an alternate content to replace your view's default experience. To minimize a possible flicker effect when replacing the content, Adobe advises to fire the call as soon as possible, on the very top of the view rendering.

In current implementation, we use the technique of dispatching **CustomEvent** on top of the view, which would trigger a Target call. This event is fired directly from application code. "General Launch Configuration & Settings" section below describes how to set up custom event listeners which, when triggered, would fire a Target call.

Here is a code example to dispatch custom event named **"event-view-start"**:

```

$rootScope.$on('$routeChangeSuccess', function() {
  // Update Data Layer, etc.

  // Custom Event - Event View Start
  var evt=new CustomEvent('event-view-start');document.body.dispatchEvent(evt);
});

```



Dispatching Custom Event for View End

Next, fire one more custom event on view bottom to trigger an Analytics call. One convenient place to trigger this event is **\$viewContentLoaded** – another Angular event that is emitted every time the *ngView* content has been reloaded ([https://docs.angularjs.org/api/ngRoute/directive/ngView#event-\\$viewContentLoaded](https://docs.angularjs.org/api/ngRoute/directive/ngView#event-$viewContentLoaded)):

```
$rootScope.$on('$viewContentLoaded', function(event) {  
    // Custom Event - Event View End  
    var evt=new CustomEvent('event-view-  
end');document.getElementById('app').dispatchEvent(evt);  
});
```

Firing event **“event-view-start”** on top of the view and **“event-view-end”** on bottom of the view will trigger Target call and Analytics call respectively. Listeners for these events are set inside of Adobe Launch rules, which are covered in “General Launch Configuration & Settings” section below.

Dispatching Custom Event for User Actions

So far, we have covered 2 custom events that trigger Target call on view top and Analytics call on view bottom. There is one more custom event we need to implement to send user actions, such as clicks or other DOM events. To measure success or user interactions for Target with A4T, we need to fire **s.tl** call for Analytics. This is conveniently set up in the Launch rule as described in “General Launch Configuration & Settings” section below.

In our demo, below is an example of how we trigger custom event **“event-action-trigger”** when user clicks an element:

```
$('a.btn').click(function(){  
    var event=new CustomEvent('event-action-trigger',  
    {detail:  
        {action:'add-to-cart',  
        productID:'123987-MNHDE-3765',  
        pageName:document.title,  
        linkName: 'add-to-cart'  
        }  
    }  
    );  
    document.querySelector('a.btn').dispatchEvent(event);  
});
```

This event also sends additional information in the “detail” object that can be processed in the rule and sent to Analytics.



General Launch Configuration & Settings

Rule Configuration

A SPA configuration should consist of at least three rules:

1. **app load: initial app load** – triggered at the top of the page
2. **page load: event-view-start** – triggered by the custom event 'event-view-start' that is dispatched when the view (content) starts to render
3. **page load: event-view-end** – triggered by the custom event 'event-view-end' that is dispatched when the view (content) is finished rendering

1. app load: initial app load

This rule is used for anything that needs to be loaded or fired at the top of the page. For example, if you're using Adobe Target, you would add an action to load Adobe Target.

EVENTS

Under the Create New Rule page, provide an appropriate name for the rule, and click on the “Add” button (screenshot below) under the “EVENTS” section.

Overview Rules Data Elements Extensions Adapters Environments Publishing

Create New Rule

Name

app load: initial app load

If - Determines when you want the rule to fire

EVENTS ⓘ

+ Add

Under the Add Event Configuration page, follow the steps outlined below.



Event Configuration

Extension
Core 1.

Event Type
Library Loaded (Page Top) 2.

Name
Core - Library Loaded (Page Top)

1. Select "Core" from the Extension dropdown
2. Select "Library Loaded (Page Top)" from the Event Type dropdown

Keep Changes Cancel

Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Rule creation page.

CONDITIONS

None

EXCEPTIONS

None

ACTIONS

These will be added under the specific Experience Cloud solution sections later in this document.

Save Rule Cancel

Click on the Save Rule button on the bottom of the Rule creation page to Save the Rule.

2. page load: event-view-start

This rule is used for anything that needs to be loaded or fired after any libraries/tools have loaded, but before the view (content) has rendered. For example, if you're using Adobe Target, you would add an action to fire the global mbox.

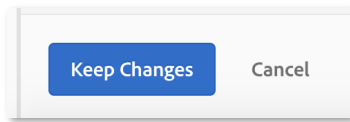
EVENTS



Under the Create New Rule page, provide an appropriate name for the rule, and click on the “Add” button (screenshot below) under the “EVENTS” section.

Under the Add Event Configuration page, follow the steps outlined below.

1. Select “Core” under the Extension dropdown.
2. Select “Custom Event” under the Event Type dropdown.
3. Enter “event-view-start” in the Custom Event Type input box.
4. Select Specific Elements radio button.
5. Enter “body” in the Elements matching the CSS selector input box. If you have a more specific selector such as a root DOM container ID, you could use that instead of “body”. Please ensure that you dispatch the custom event to the same selector mentioned here.



Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Rule creation page.

CONDITIONS

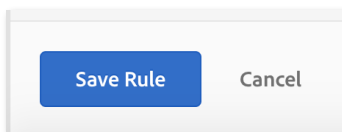
None

EXCEPTIONS

None

ACTIONS

These will be added under the specific Experience Cloud solution sections later in this document.



Click on the Save Rule button on the bottom of the Rule creation page to Save the Rule.

3. page load: event-view-end

This rule is triggered at the bottom of the page and is used for anything that needs to be loaded or fired after the view (content) has rendered.

EVENTS

Under the Create New Rule page, provide an appropriate name for the rule, and click on the "Add" button (screenshot below) under the "EVENTS" section.



Overview **Rules** Data Elements Extensions Adapters Environments Publishing

Create New Rule

Name

page load: event-view-end

If - Determines when you want the rule to fire

EVENTS ⓘ

+ Add

Under the Add Event Configuration page, follow the steps outlined below.

Overview **Rules** Data Elements Extensions Adapters Environments Publishing

Event Configuration

Extension

Core

Event Type

Custom Event

Name

Core - Custom Event

Custom Event Type

event-view-end

☒ Specific elements ☐ any element

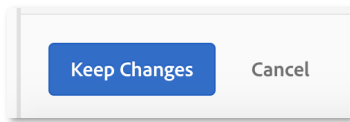
Elements matching the CSS selector

body

☐ and having certain property values...

> [Advanced](#)

1. Select "Core" under the Extension dropdown.
2. Select "Custom Event" under the Event Type dropdown.
3. Enter "event-view-end" in the Custom Event Type input box.
4. Select Specific Elements radio button.
5. Enter "body" in the Elements matching the CSS selector input box. If you have a more specific selector such as a root DOM container ID, you could use that instead of "body". Please ensure that you dispatch the custom event to the same selector mentioned here.



Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Rule creation page.

CONDITIONS

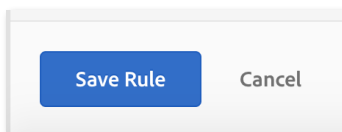
None

EXCEPTIONS

None

ACTIONS

These will be added under the specific Experience Cloud solution sections later in this document.



Click on the Save Rule button on the bottom of the Rule creation page to Save the Rule.



Adobe Target Configuration & Tagging

Overview

Adobe Target is the Adobe Marketing Cloud solution that provides everything you need to tailor and personalize your customers' experience so you can maximize revenue on your web and mobile sites, apps, social media, and other digital channels.

Prerequisites & Product-Specific Setup

In order to use Adobe Target, your contract with Adobe must be provisioned for it and you must have completed the Launch Basic setup and Target extension setup in the [basic guide](#).

Once you've confirmed this you can install the Adobe Target extension. To do this, follow the steps below.

Further documentation can be found at the following URL:

https://marketing.adobe.com/resources/help/en_US/experience-cloud/launch/c_extension-target.html

Rule Configuration

In order to properly load the Target library, we need to add an action to the 'app load: initial app load' rule we previously created.

Add Actions

Navigate to the 'app load: initial app load' rule and click the Add button in the Actions section of the Edit Rule page.



Name

app load: initial app load

If - Determines when you want the rule to fire

EVENTS ⓘ

Core - Library Loaded (Page Top) +

CONDITIONS ⓘ

+ Add

EXCEPTIONS ⓘ

+ Add

THEN - Determines what you want the rule to do

ACTIONS ⓘ

+ Add

Load Target Action

On the Action configuration page, follow the steps outlined below.

Action Configuration

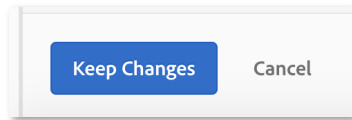
Extension

Adobe Target ▼ 1.

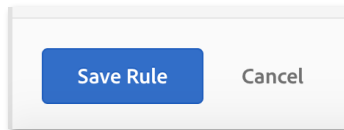
Action Type

Load Target ▼ 2.

1. Select 'Adobe Target' from the Extension dropdown
2. Select 'Load Target' from the Action Type dropdown



Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Edit Rule page.



Click on the Save Rule button on the bottom of the Rule creation page to Save the Rule.

Add Custom Event Rule

Create a new rule “page load: event-view-start” to listen to Custom Event dispatching from the application.



Name
page load: event-view-start

If - Determines when you want the rule to fire

EVENTS ⓘ

Core - Custom Event +

CONDITIONS ⓘ

+ Add

EXCEPTIONS ⓘ

+ Add

THEN - Determines what you want the rule to do

ACTIONS ⓘ

Adobe Target - Add Params to Global Mbox THEN Adobe Target - Fire Global Mbox +

The event configuration “Core - Custom Event” is shown below:

Event Configuration

Extension
Core ▼

Event Type
Custom Event ▼

Name
Core - Custom Event

Order ⓘ
50

Custom Event Type event-view-start ⓘ

☒ specific elements ☐ any element

Elements matching the CSS selector body ⓘ

☐ and having certain property values...

> Advanced

2 actions are configured as shown below:



- 1) Action “Adobe Target – Add Params to Global Mbox” adds parameters to Target Global Mbox from data value.

The screenshot shows the configuration for the 'Add Params to Global Mbox' action. On the left, the 'Action Configuration' panel has 'Extension' set to 'Adobe Target', 'Action Type' set to 'Add Params to Global Mbox', and 'Name' set to 'Adobe Target - Add Params to Global Mbox'. The main panel displays a table with two rows of parameters:

Name	Value
pagename	%page: name%
pagetype	%page: type%

Each row has a database icon and a close 'X' button. An 'Add' button is located at the bottom of the table.

- 2) Action “Adobe Target – Fire Global Mbox” fires a Target Global Mbox call.

The screenshot shows the configuration for the 'Fire Global Mbox' action. On the left, the 'Action Configuration' panel has 'Extension' set to 'Adobe Target', 'Action Type' set to 'Fire Global Mbox', and 'Name' set to 'Adobe Target - Fire Global Mbox'. The main panel shows the 'Global Mbox Name' as 'target-global-mbox'. Below this, there are two settings: 'Body Hiding' set to 'Disabled' and 'Body Hidden style' set to 'body {opacity: 0}'.



Adobe Analytics Configuration & Tagging

Overview

Adobe Analytics is an industry-leading solution that empowers you to understand your customers as people and steer your business with customer intelligence.

Prerequisites & Product-Specific Setup

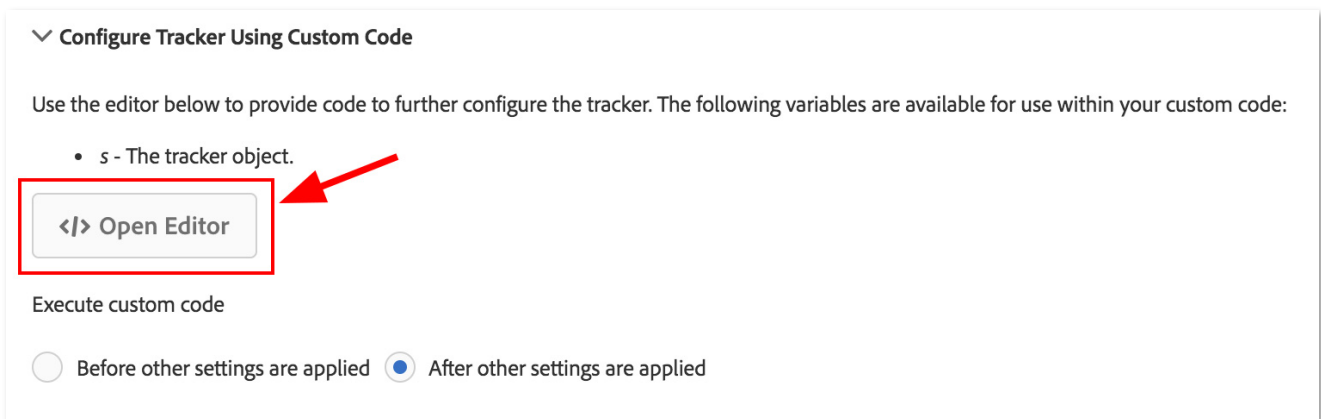
This guide assumes you have already installed and configured the Adobe Analytics extension covered in the [Basic Setup Guide](#).

Extension Setup

Additional Configuration

1. Configure Tracking Using Custom Code

Under the Configure Tracking Using Custom Code section, follow the instructions outlined below.



▼ Configure Tracker Using Custom Code

Use the editor below to provide code to further configure the tracker. The following variables are available for use within your custom code:

- s - The tracker object.

</> Open Editor

Execute custom code

☐ Before other settings are applied ☒ After other settings are applied

Click the Open Editor button. This is where you'll add any plugins you plan to use in your implementation. For SPA implementations, it is a good idea to use the following plugins:

- apl
- inList

Contact your Adobe consultant to obtain the latest versions of any plugins you may need.

Rule Configuration

In order to capture and send Analytics data to the Adobe servers, we need to add actions to the 'event-view-end' rule we previously created.

Add Actions

Navigate to the 'event-view-end' rule and click the Add button in the Actions section of the Edit Rule page.



Name

page load: event-view-end

If - Determines *when* you want the rule to fire

EVENTS ⓘ

Core - Custom Event +

CONDITIONS ⓘ

+ Add

EXCEPTIONS ⓘ

+ Add

THEN - Determines *what* you want the rule to do

ACTIONS ⓘ

+ Add

Set Variables Action

On the Action Configuration page, follow the steps outlined below.



1. Select 'Adobe Analytics' from the Extension dropdown
2. Select 'Set Variables' from the Action Type dropdown
3. Configure your page load variables per your implementations requirements

Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Edit Rule page.

Send Beacon Action

Click the 'plus' icon in the Actions section

On the Action Configuration page, follow the steps outlined below.



1. Select 'Adobe Analytics' from the Extension dropdown
2. Select 'Send Beacon' from the Action Type dropdown
3. Select the 's.t()' radio button in the Tracking section

Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Edit Rule page.

Clear Variables Action


This is a new feature in Adobe Launch and extremely helpful in SPA implementations. In a traditional implementation where the browser reloads between pages, the JavaScript object that stores all of the Analytics variables is automatically cleared out and reset when a new page is visited. This doesn't happen in a SPA because the browser doesn't reload between view/state changes. Therefore, we have to explicitly tell Launch when to clear the object. If we don't, variables will persist between image requests and skew our data.


Click the 'plus' icon in the Actions section

On the Action Configuration page, follow the steps outlined below.





Action Configuration



Extension
Adobe Analytics  1.

Action Type
Clear Variables  2.

1. Select 'Adobe Analytics' from the Extension dropdown
2. Select 'Clear Variables' from the Action Type dropdown

Click the Keep Changes button on the bottom of the page to Save the configuration and return to the Edit Rule page.

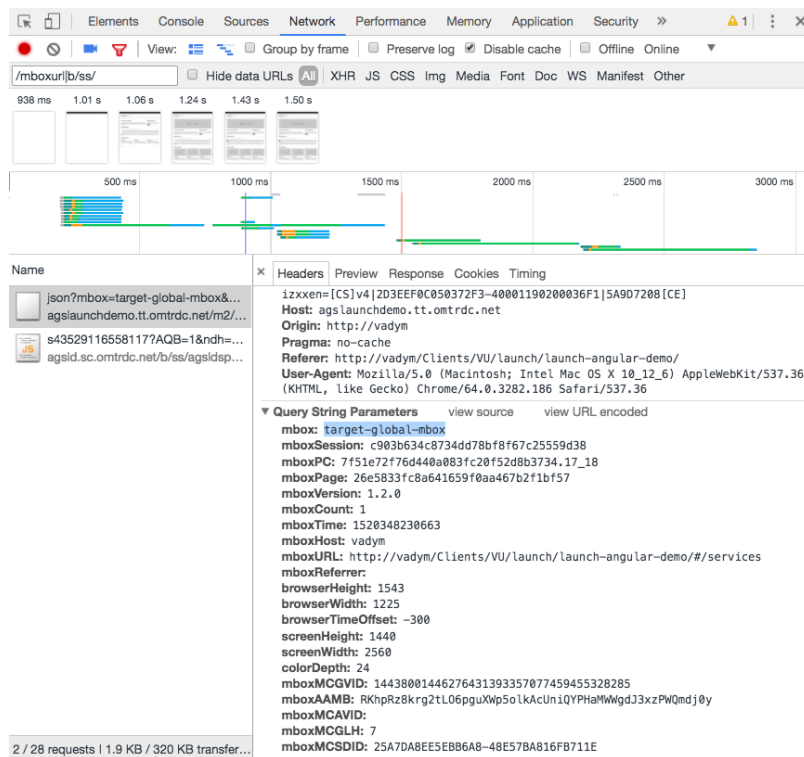
 

Click on the Save Rule button on the bottom of the Rule creation page to Save the Rule.



Implementation Validation






If you followed this guide to set up your own implementation, at this moment, you should have a working AngularJS environment with Launch library and 3 custom events firing on every view change and on user clicks. In Launch, each of these custom events would trigger Target and Analytics calls. To validate your implementation, please open your browser's Developer Console Network tab to look for Target and Analytics calls. Apply filter to display only the calls we need as shown on the screenshot below.



Step 1: Ensure your initial page load fires Target call first, then followed by Analytics call. The `json` endpoint is a Target call, `b/ss` is an Analytics call – `s.t()`.

Step 2: Ensure your click tracking Analytics call is firing. Current implementation listens to navigation menu clicks and fires a click tracking call `s.tl()`. Screenshot below shows the third (selected) tracking call triggered just before the next set of Target/Analytics calls for the new view load.



Name
 json?mbox=target-global-mbox&... agslaunchedemo.tt.omtrdc.net/m2/...
 s49630519499684?AQB=1&ndh=... agsld.sc.omtrdc.net/b/ss/agsldsp...
 s44847895922444?AQB=1&ndh=... agsld.sc.omtrdc.net/b/ss/agsldsp...
 json?mbox=target-global-mbox&... agslaunchedemo.tt.omtrdc.net/m2/...
 s46233273937789?AQB=1&ndh=... agsld.sc.omtrdc.net/b/ss/agsldsp...

Step 3: if you are using Analytics for Target (A4T), the call order is especially important. Target call always must fire before Analytics call. All Target calls and the first Analytics call will contain a generated Supplemental Data ID (SDID), where the value must be the same per page lifecycle. Target uses a parameter **mboxMCSDID** and Analytics uses a parameter **sdid** to pass this value. If values match between Target and Analytics calls, data will be stitched on Adobe servers' backend for Analytics reports. The new page or view will generate a new value for SDID.

If you have observed Target and Analytics calls on page load and view change, as well as tracking call was successfully triggered on your tracking elements – your implementation is verified.

Integrations

Integration Consideration 1: Analytics for Target (A4T)

In addition to implementing the Experience Cloud Id Service, A4T requires that Target loads before Analytics. If you fire a global mbox in a “Library Loaded (Page Top)” rule and the Analytics page view beacon in a “Page Bottom” rule as configured in this reference architecture, you will be all set. You should always see matching supplemental data ids (SDIDs) in both calls—these ids stitch hits together in the Analytics processing for A4T.

Integration Consideration 2: Visitor resetState Method

In Single Page Applications, we may fire more than just one global mbox per view. For example, you can fire regional mboxes in addition to global mbox to apply alternate content to smaller sections of the view. Visitor ID Service introduced resetState method to help solve issues related to working with IDs on single page sites/screens or apps. Use this method to reset SDID values before the first Target and following Analytics call:

```
var visitor = Visitor.getInstance("Marketing_Cloud_organization_ID@AdobeOrg");
if(typeof visitor.resetState==='function'){
    visitor.resetState();
}
```

Integration Consideration 3: Visual Experience Composer (VEC)

Adobe Target VEC is supported with AngularJS framework. However, in more complicated use cases, you may consider using Custom Code editor of VEC to apply AngularJS specific code when modifying sections that use such AngularJS



functionality. For example, *ng-click*, etc. You may need to consult with your AngularJS developer on how to correctly apply such alternate content to the page.

Experience Cloud ID Tagging & Configuration

Configure the Experience Cloud ID Service according to the documentation in the [Basic Setup Guide](#).



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 or Client-side Instructions:

Server-side forwarding

Server-side forwarding is implemented on this demo site because it also has Adobe Analytics. The instructions on how it was implemented, and how you can implement server-side forwarding on your site are found in the <https://helpx.adobe.com/experience-manager/kt/integration/using/launch-reference-architecture-guides.html> document. Please complete the steps in that document to implement server-side forwarding of data from Adobe Analytics to Adobe Audience Manager.

Client-side

Client-side implementation options are presented to you below, in case that better represents your environment.

Prerequisites & Product-Specific Setup – AAM Only – Client Side

Client-Side DIL – this solution covers anyone who has neither Adobe Analytics nor Google Analytics. DIL code (Data Integration Library Code) sends data from the web page into AAM directly. Hence, there is no need to enable any server-side forwarding.

Which Extension Do I Configure in Launch?

In this case, we will configure and use the Adobe Audience Manager extension and will pull variables out of the page's data layer to be sent directly to AAM.


Steps to set up AAM DIL in Launch

1. Add the extension for Adobe Audience Manager



AGS Launch Demo >
SPA Demo with AAM Only

Overview Rules Data Elements **Extensions** Adapters Environments Publishing



Adobe Audience Manager (DIL)
Adobe Systems
VERSION 1.1.3

DIL version: 6.12

Note: This extension is not meant to be used for server-side forwarding (SSF) of Adobe Analytics data. For SSF, use the Adobe Analytics extension.

[DIL Documentation](#)

Exclude Specific Paths ⓘ


Add Path

2. Within the configure extension there is nothing to add to the configuration once the extension is enabled. Verify the partner ID and the OrgID (these will be automatically added, and you should not need to change them).



AGS Launch Demo >
SPA Demo with AAM Only

Overview Rules Data Elements **Extensions** Adapters Environments Publishing



Adobe Audience Manager (DIL)
Adobe Systems
VERSION 1.1.3

This is the Adobe Audience Manager (AAM) Data Integration Library (DIL v6.12) extension (client-side implementation). Note: This extension is not meant to be used for server-side forwarding (SSF) of Adobe...

DIL version: 6.12

Note: This extension is not meant to be used for server-side forwarding (SSF) of Adobe Analytics data. For SSF, use the Adobe Analytics extension.

[DIL Documentation](#)




Exclude Specific Paths ⓘ

☐ Use DIL Site Catalyst Module

Add a note:

☐ Use DIL Google Analytics Module

DIL.create Initialization Properties ⓘ

partner	=	agslaunchdemo	
secureDataCollection	=	false	<input checked="" type="radio"/> False
visitorService	=	Use the subproperties or select a data elem	
Subproperties for visitorService Object ⓘ			
namespace	=	68825AEC5A0DC5490A495ESA@AdobeOrg	

3. Create one Rule, name the rule "event-view-end"

AGS Launch Demo >
SPA Demo with AAM Only

Overview **Rules** Data Elements Extensions Adapter

Status ▾

Displaying 1 rule

<input type="checkbox"/>	Name
<input type="checkbox"/>	event-view-end

4. Add and event within Rules and select Extension "Core", Action Type "Custom Event", Name "Core – Custom Event"



AGS Launch Demo >
SPA Demo with AAM Only

Overview Rules Data Elements Extensions Adapters Environments Publishing

Event Configuration

Extension
Core

Event Type
Custom Event

Name
Core - Custom Event

Custom Event Type event-view-end

☒ specific elements ☐ any element

Elements matching the CSS selector #app

☐ and having certain property values...

> Advanced

5. Add and Action within the Rules and select Extension "Adobe Audience Manager", Action Type "Run Custom Code", Name "Audience Manager (DIL) – Run Custom Code" select "Code Editor"

AGS Launch Demo >
SPA Demo with AAM Only

Overview Rules Data Elements Extensions Adapters Environments Publishing

Action Configuration

Extension
Adobe Audience Manage...

Action Type
Run Custom Code

Name
Adobe Audience Manager (DIL) - I

Run Custom Code

Code Editor

Add a note:

6. Paste Code into the code editor that will pull information off your page, presumably from a data layer. Use the example code to know how to set your custom code. In the following example, we are pulling data from a "digitalData" data layer. You can use the format to pull data from your data layer or from other variables on your page.



```
/*
  Use the variable `dilInstance`

  Example:

    dilInstance.api.signals({
      c_zip: variableOnPage
    }).submit();

    or

    DIL.modules.GA.submitUniversalAnalytics(ga, dilInstance);
*/
dilInstance.api.signals({
  c_pagename: digitalData.page.pageInfo.pageID,
  c_category: digitalData.page.category.subCategory1,
  c_attributes: digitalData.page.attributes.project
}).submit();
```

7. Confirm you code looks like this in the editor:

```
1  /*
2    Use the variable `dilInstance`
3
4    Example:
5
6      dilInstance.api.signals({
7        c_zip: variableOnPage
8      }).submit();
9
10     or
11
12     DIL.modules.GA.submitUniversalAnalytics(ga, dilInstance);
13  */
14  dilInstance.api.signals({
15    c_pagename: digitalData.page.pageInfo.pageID,
16    c_category: digitalData.page.category.subCategory1,
17    c_attributes: digitalData.page.attributes.project
18  }).submit();
```

8. The completed rule for "Insert Custom Variable will have Events "Core – Library Loaded (Page Top) and Actions "Adobe Audience Manager(DIL) – Run Customs Code"



Overview **Rules** Data Elements Extensions Adapters

Edit Rule

Name

event-view-end

If - Determines when you want the rule to fire

EVENTS ⓘ

Core - Custom Event +

CONDITIONS ⓘ

+ Add

EXCEPTIONS ⓘ

+ Add

THEN - Determines what you want the rule to do

ACTIONS ⓘ

Adobe Audience Manager (DIL) - Run Custom Code +

Build your Library

9. Create a new library in the Development 1 environment
10. Add the changes to the Analytics extension and build the library
11. Open the Demo site
12. Make sure that Launch Command is on and set to load the Development 1 embed code

Publish your implementation of AAM DIL

Approve and Publish the library to Production

Testing Considerations & Steps

You have two options here for testing and validation:

1. If you have followed these steps above on your own site and your own Launch property, you can validate your site
2. If you want to see this use case on our demo site, you can use the Launch Command extension for Chrome to see it implemented



To Validate AAM Only on Your Property

Assumptions:

1. For this step, we will assume that you have implemented as described above, and that you have pushed your library through to production
2. We will also assume that you have place the production embed code on your site

Validation:

1. Open a debugger on your machine (E.g. Charles, or the Chrome debugger, etc)
2. To see the AAM hit, filter the debugger to "demdex"
3. Run/refresh your page
4. In the debugger, you should have a hit that has "event" in it. Select that hit
5. You should be able to see the variables that were pulled from your page in the AAM hit

To See AAM Only on the AAM Only - SPA Demo Site

For this, you should use the Launch Command extension for Chrome

1. Open the Launch Command Extension
2. Type "Adobe SPA Demo - AAM Only" – Production" in the first box

Paste the following embed code for the environment into the second box

```
<script src="//assets.adobedtm.com/launch-EN94c632ce757541e183fe3ae50af84cd9.min.js"></script>
```

3. Check the debugger for the call by searching for "demdex" call (make sure the page is refreshed). You should see the hits described above in the call.
4. Verify that there is an AAMUUID in the response