Adobe Experience Cloud

Discover Launch, by Adobe

student guide

ADOBE DIGITAL LEARNING SERVICES

Aleksander Dultsev

Discover Launch, by Adobe

The content of this guide is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this guide.

Please remember that existing artwork or images that you may want to include in your project may be protected under copyright law. The unauthorized incorporation of such material into your new work could be a violation of the rights of the copyright owner. Please be sure to obtain any permission required from the copyright owner.

Any references to company names in sample templates are for demonstration purposes only and are not intended to refer to any actual organization.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Notice to U.S. Government End Users. The Software and Documentation are “Commercial Items,” as that term is defined at 48 C.F.R. §2.101, consisting of “Commercial Computer Software” and “Commercial Computer Software Documentation,” as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished-rights reserved under the copyright laws of the United States. Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250, and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

04202018

Table of Contents

Launch, by Adobe: An Overview

Introduction

Launch, by Adobe is a tag management solution that enables you to build and manage extensions, define the way data is captured, and deploy web apps easily. It reduces your dependency on third-party vendors, speeds up time-to-market, reduces friction between IT and marketing teams, and helps gain accurate and consistent insights with unified data.

Objectives

After completing this course, you will be able to:

- Describe Launch capabilities
- Describe key concepts related to Launch
- Log on to Launch
- Set up Launch
- Install extensions
- Configure extensions
- Create data elements
- Implement rules

Launch: Capabilities

By using Launch, you can:

- Manage tags for Adobe solutions and third-parties in one place with little or no reliance on the IT team
- Build and manage extensions directly and independently within the Adobe Cloud platform
- Capture, define, and manage data consistently between marketing and advertising products from Adobe and other vendors
- Browse, deploy, and activate the right marketing tool quickly and easily. For example, you can implement a chat service quickly
- Share data across your teams, so that the right people have access to the right data, ensuring the delivery of better experiences
- Create unlimited development environments
- Automate deployments through APIs
- Integrate the Launch APIs with your internal systems

Launch: Key Concepts

The key concepts related to Launch are:

- Adapters
- Environment
- Data elements
- Rules
- Extensions
- Properties and Companies
- Publishing

Adapters

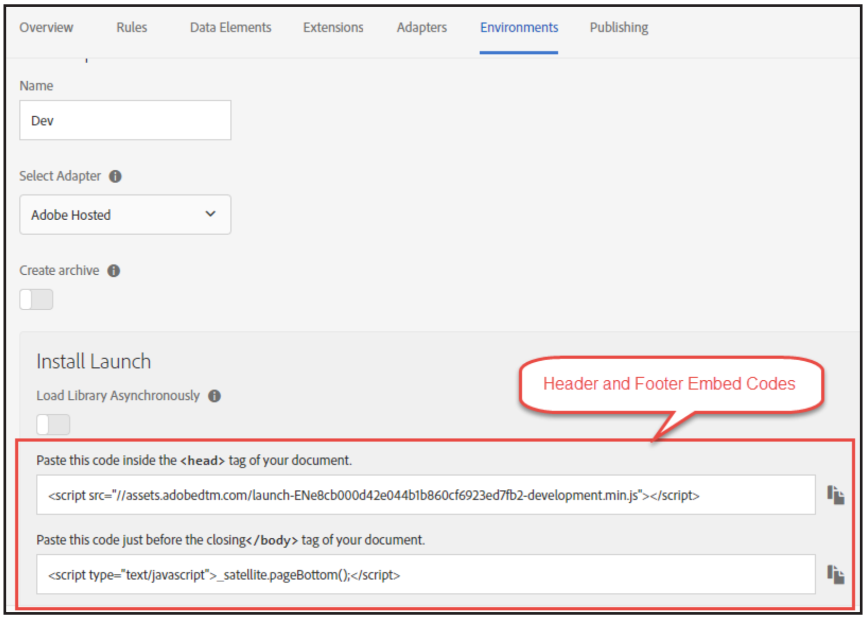
Adapters in Launch refer to the destination(s) where you want to deploy Launch libraries. You can deploy libraries using Akamai instance (using Adobe's Edge CDN, which sits on the Akamai architecture) or on your own web servers (using SFTP method). You have the flexibility to host your production libraries on your own domain, and use Adobe's Akamai instance for development environments.

Environment

An environment is a set of deployment instructions for build location and format. An environment has a one-to-one relationship with an embed code. The embed code:

- Consists of two <script> tags that you need to include within the HTML of your website
- Is automatically generated based on the environment configuration. You need to copy and paste the codes in web pages where you want to run Launch.
- Is used to call the JavaScript library code

Notice the header and footer codes in the image below:



Name: Dev

Select Adapter: Adobe Hosted

Create archive

Install Launch

Load Library Asynchronously

Paste this code inside the **<head>** tag of your document.

```
<script src="//assets.adobedtm.com/launch-ENe8cb000d42e044b1b860cf6923ed7fb2-development.min.js"></script>
```

Paste this code just before the closing**</body>** tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

Launch has three environments:

- Development environment: To develop and test libraries. Each development environment has its own embed code, which is deployed on your site(s). You can create unlimited development environments.
- Staging environment: To test and approve a library before it is published. You can have only one staging environment, and it is optional.
- Production environment: This environment is used for the live, customer-facing site. The production environment has an embed code, which should be deployed on the production site. You can have only one production environment.

Data Elements

Launch uses data elements to collect, organize, and deliver data across technologies. It is a built-in data dictionary that enables data reuse and multiple source inputs.

Data elements improve the efficiency of the tagging process. You can define data rules once and then reuse them in multiple places. For example, if you use a page name schema by referencing the document.title element, you can create a 'Page Name' data element in Launch and use it in any rule that needs to reference the page name. If you decide to change the way you refer the page name, you just need to change the reference once in the data element you created, and all rules that reference that data element are automatically updated.

In addition, you can:

- Map a data element value to URLs, cookie values, and JavaScript variables. You can reference this value with its variable name throughout Launch.
- Use the collection of data elements to build events, conditions, and actions, which can be shared across Launch for reuse.

The element type available to you is dependent on the extensions you have installed. For Launch Core extension, the following data element types are available:

- Cookie
- Custom Code
- DOM Attribute
- Local Storage
- Page Info
- Query String Parameter
- Random Number
- Session Storage
- JavaScript Variable

Rules

A rule is an "if this, then that" formula. If a specified event takes place, the rule is triggered, and specified actions take place. You can build rules to integrate disparate products into a single solution. Launch searches for user interaction and associated data, and when the criteria defined in your rules are met, extensions, scripts, or HTMLs are triggered.

A rule consists of two parts:

- Events (If): An event is what you want the rule to look for. You can specify one or more events that must take place to trigger the rule. Multiple events are joined by an AND operation. Optionally, you can narrow the event by configuring conditions that must be true for an event to trigger the rule, and/or specify exceptions that stop the rule from triggering, even if the events and conditions are met.
- Actions (Then): An action occurs after an event takes place and all conditions and exclusions are satisfied. A rule in Launch can trigger as many discrete actions as you want, and you can control the order in which these actions occur.

Extensions

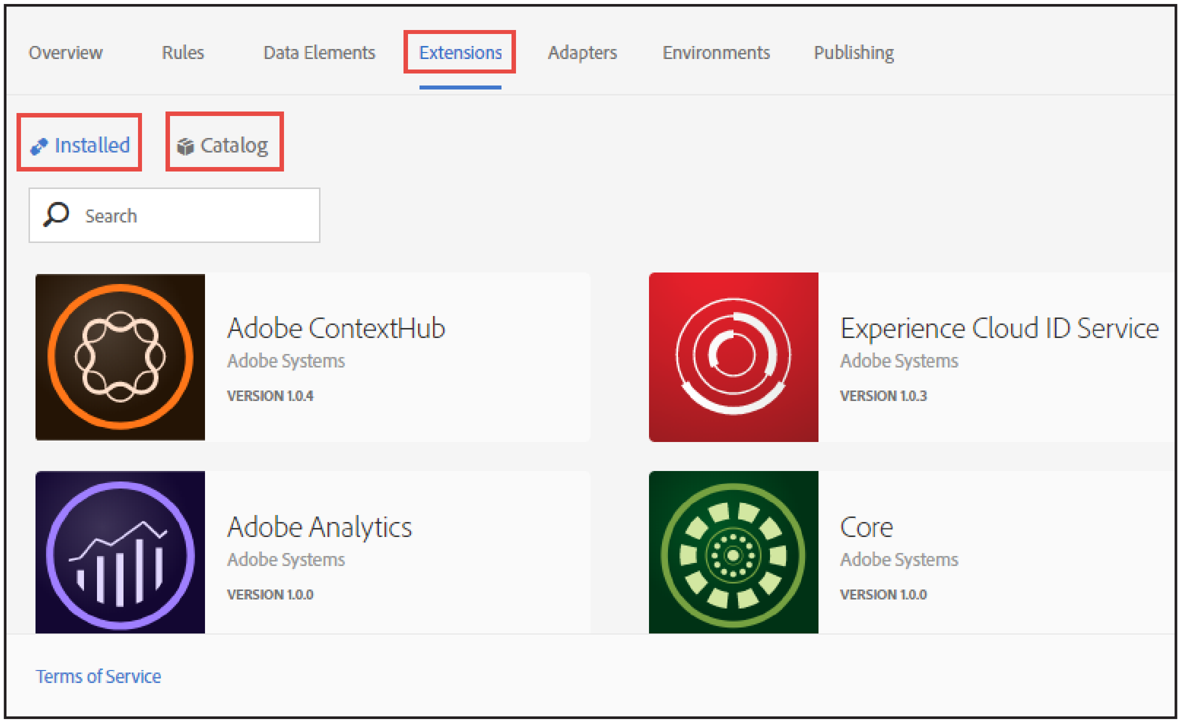
Extensions are like apps that run on the Launch platform. An extension is a packaged set of code that extends the Launch interface and the library functionality. The elements available to you for building rules and data elements are determined by extensions. When you add an extension, you get new data elements and new options for creating rules.

By default, Adobe provides extensions that support common integrations. You can add, customize, and configure extensions depending on your needs.

The Extensions page in Launch has two tabs:

- Installed tab: Shows all the installed extensions
- Catalog tab: Shows all the available extensions

Note: In Dynamic Tag Management (DTM), Extensions are referred to as Tools.



Adobe ContextHub
Adobe Systems
VERSION 1.0.4

Experience Cloud ID Service
Adobe Systems
VERSION 1.0.3

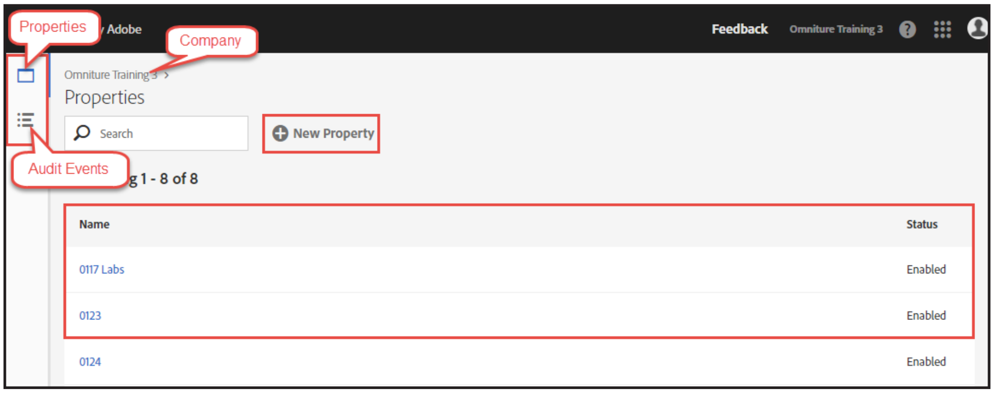
Adobe Analytics
Adobe Systems
VERSION 1.0.0

Core
Adobe Systems
VERSION 1.0.0

Properties and Companies

- A property is a collection of rules, data elements, configured extensions, environments, and libraries. Launch company accounts contain one or more properties. You can have one or several company accounts. Within a property, you can have one or more domains and subdomains.
- If a single property contains multiple domains and subdomains, add conditional logic to separate unique data collection and tracking needs that only apply to individual domains, subdomains, or subsets of domains and subdomains.
- You can choose to have one company that contains all properties or you can have a company account that contains several properties for each site you manage. For example, you can have three properties within your company account, where one contains all your blog sites, another contains all your ecommerce sites, and the third contains all your lead-generation sites. This enables you to effectively handle multiple geographies, business units, domains, and subdomains in Launch. The best practice is to have multiple properties in a single company.

Tip: You can access companies from the top-left section of the Launch UI, as shown in the image below. This is particularly useful if you manage multiple accounts. Select a company to see the properties and audit logs for that company. Click a property to see an overview of that property. The Overview tab displays activity performed on the property.



Name	Status
0117 Labs	Enabled
0123	Enabled
0124	Enabled

Publishing

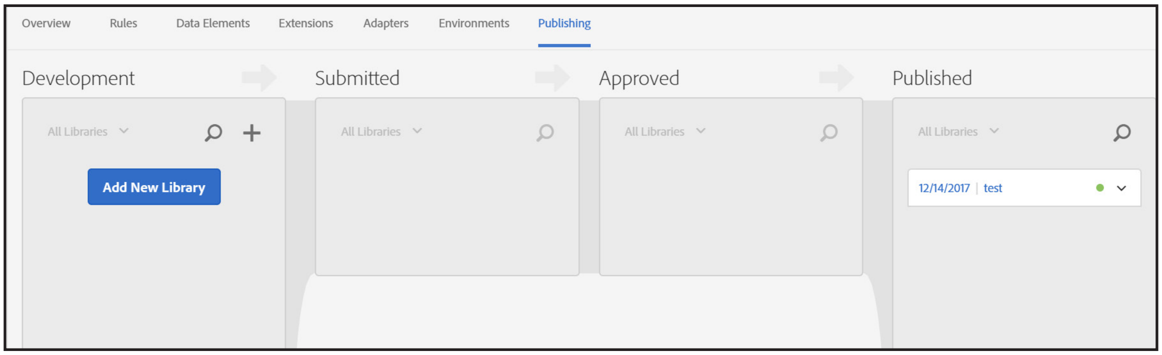
In Launch, after creating extensions, data elements, and rules, you need to publish them. It is important to understand the following components associated with the publishing process:

- Library: Is a set of instructions for how extensions, data elements, and rules interact with one another and with your website after they are deployed. When creating a library, you need to specify the changes you want to make to your library. Libraries are approved or rejected as a whole. You cannot approve or reject individual items within a library. Libraries are compiled into builds.
- Build: Is the actual set of files containing the code that is delivered to each user's browser when the user views your site. The build consists of one or more JavaScript files that reference each other. These files are delivered to your hosting location using the environment and adapter that you have chosen for the library. The embed code that you deploy on your site points to this same location so the files can load in a browser when a user accesses your site.
- Approval workflow: Refers to the process of creating libraries, testing builds, and approving them for production. There are three levels of permission in the approval workflow:
  - › The develop right: Includes the ability to create libraries, build for development, and submit for approval
  - › The approve right: Includes the ability to build for staging and approve
  - › The publish right: Includes the ability to publish an approved library

Note: The rights are not inclusive. If a single user needs to perform the workflow from start to finish, that user must be granted all three rights within a given property.

The publishing process consists of:

1. Creating and editing libraries.
2. Testing the functionality of libraries.
3. Deploying those libraries to your production site.

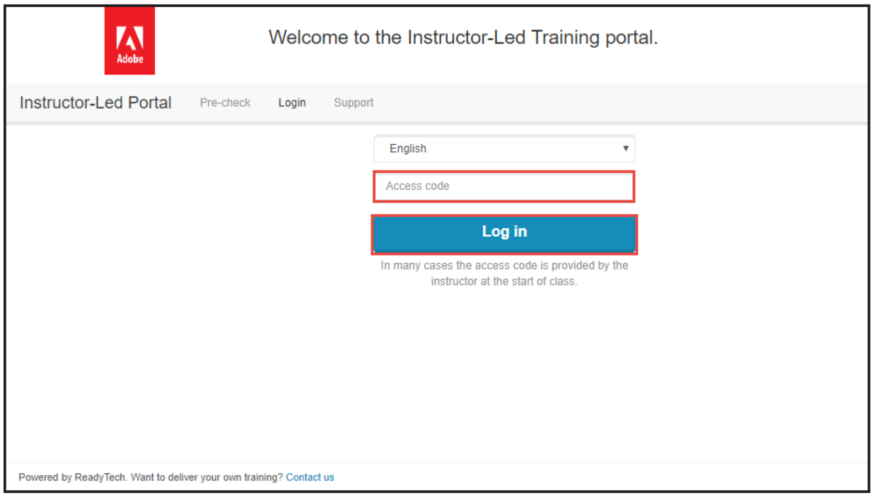


Exercise 1: Set up Launch

Task 1: Log on to Launch

In this training, you will use a ReadyTech training environment to access Launch. To log on to the ReadyTech environment:

1. Open a browser and navigate to https://adobe.instructorled.training to access the ReadyTech training site.
2. Enter the **Access Code** provided to you and click **Log in**, as shown**.** The Instructor-Led Training Portal screen opens.

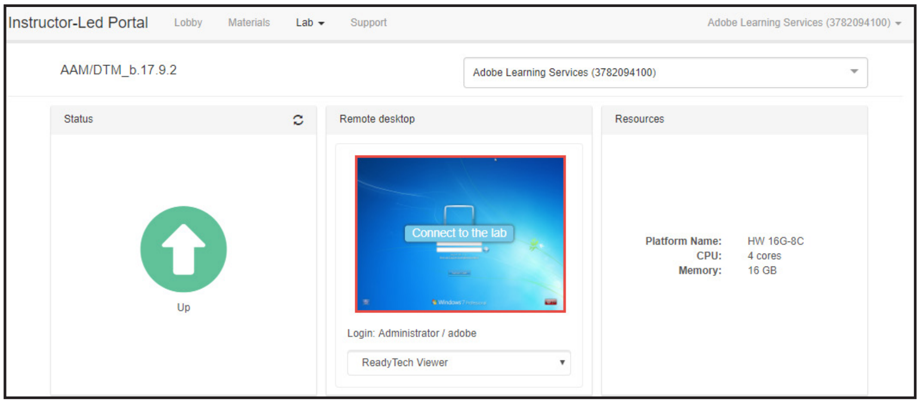


Welcome to the Instructor-Led Training portal.

In many cases the access code is provided by the instructor at the start of class.

Powered by ReadyTech. Want to deliver your own training? Contact us

3. Click **Connect to the lab**, as shown:



AAM/DTM_b.17.9.2

Adobe Learning Services (3782094100)

Login: Administrator / adobe

Platform Name: HW 16G-8C
CPU: 4 cores
Memory: 16 GB

4. Enter the login credentials provided to you by your instructor. The ReadyTech desktop is displayed.
5. Open Google Chrome in ReadyTech and navigate to http://launch.adobe.com
6. Enter the Launch login credentials provided to you by your instructor, and click **Sign in**.

Notice that when you log on to Launch for the first time, you cannot access any of its UI elements except for the Property icon on the left pane. This is because you have not created a property yet. You will first create a property, and within that property, you will create adapters, environments, data elements, rules, and extensions.

Best Practice: If the data you are collecting, variables, extensions, and tags, are similar across websites, group the sites into one property to avoid duplicating or copying rules from one property to another. If the properties are unique to each website, separate the sites in their own properties to have more control. In addition, assign different roles to different people for all your properties, or on a per-property basis. If someone has sufficient rights, that person can perform administrative actions across all the properties in that Launch company. All the other roles can be assigned on a per-property basis. You can even hide a property from certain users (non admins) by not giving them any role in that property.

Task 2: Create a property

1. In the previous task, you logged on to Launch and accessed the Properties page. On the **Properties** page, click **Create New Property** to create a new property. The **Create Property** dialog box opens.

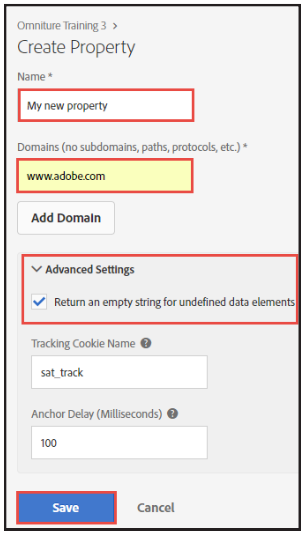
2. Update the property fields, as shown:

   a. In the **Name** field, enter **My new property**.

   b. In the **Domains** field, enter **www.adobe.com**.

   c. Under **Advanced Settings**, select the **Return an empty string for undefined data elements** if you want undefined values to be empty rather than default values.

   d. Optionally, you can specify a custom cookie name in the **Tracking Cookie Name** field. Launch uses the Tracking Cookie Name to track your opt-out status for receiving other cookies. Enter the delay time in the **Anchor Delay (milliseconds)** field. Anchor Delay is the delay time for tracking the link. Longer delays improve tracking accuracy. Any value under 500ms is recommended. For this exercise, retain the default values.



Omniture Training 3 >

Create Property

Name *

My new property

Domains (no subdomains, paths, protocols, etc.) *

www.adobe.com

Advanced Settings

☑ Return an empty string for undefined data elements

Tracking Cookie Name

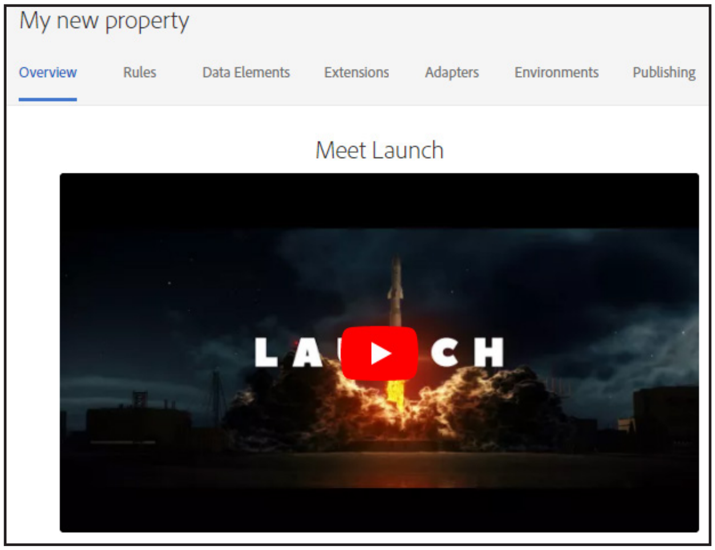
sat_track

Anchor Delay (Milliseconds)

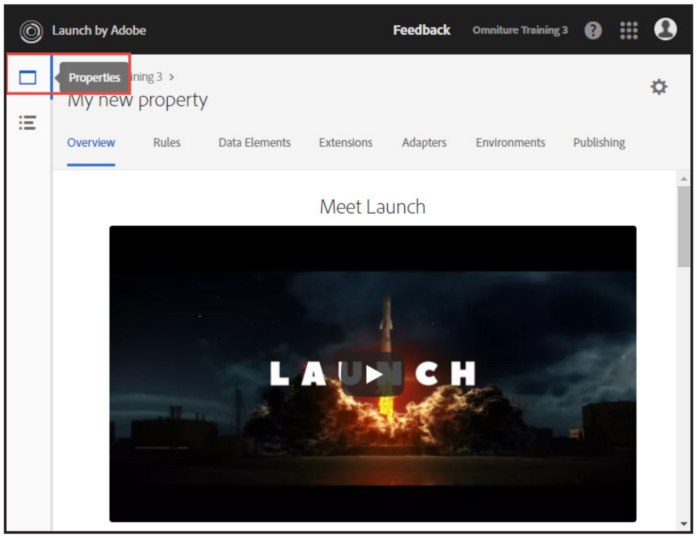
100

3. Click **Save**. Notice that the new property you created is listed on the **Properties** page.

4. Click the property you created. The **My new property** page opens. The **Overview** tab lists all your recent activities. As you have not created anything so far, your **Overview** tab should look similar to the one shown in the screenshot:



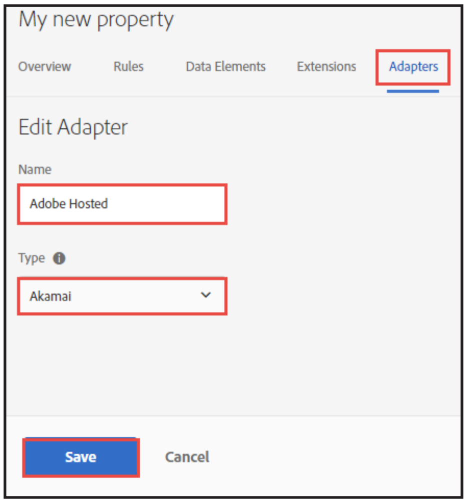
5. Click the **Properties** icon, as shown, to go back to the **Properties** page. Notice the newly created property on the **Properties** page.



Task 3: Create adapters

After creating a property, the next step is to set up adapters for hosting environments.

1. From the **Properties** page, click the property you created in the previous task to open it. The **My new property** page opens.
2. on the **Adapters** tab, click **Create a New Adapter**. The **Edit Adapter** page opens.
3. Enter a name for the adapter in the **Name** field. For example, enter Adobe Hosted.
4. From the **Type** drop-down list, select **Akamai,** as shown:



My new property

Edit Adapter

Name: Adobe Hosted

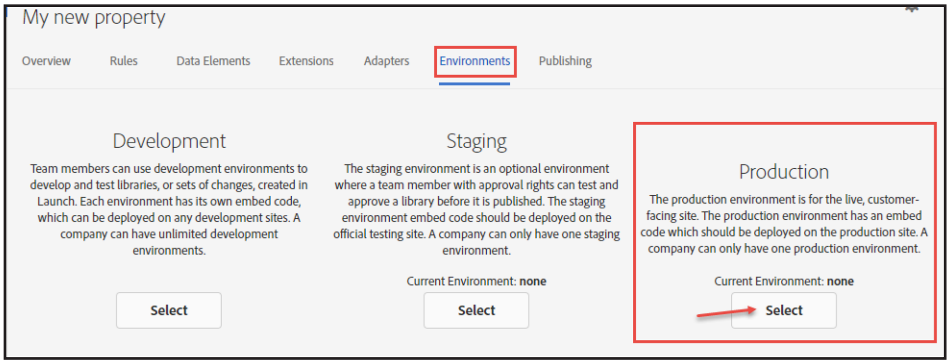
Type: Akamai

5. Click **Save** > **Close**.

NOTE: You can add multiple adapters by clicking the **Add Adapter** button. Additionally, you can create multiple FTP adapters for staging and production environments. In this training, you will be using Akamai.

Task 4: Setup environment

1. On the **Environments** tab, click **Create New Environment**.
2. To set up a production environment, click **Select** under **Production**, as shown:



Development

Team members can use development environments to develop and test libraries, or sets of changes, created in Launch. Each environment has its own embed code, which can be deployed on any development sites. A company can have unlimited development environments.

Staging

The staging environment is an optional environment where a team member with approval rights can test and approve a library before it is published. The staging environment embed code should be deployed on the official testing site. A company can only have one staging environment.

Current Environment: **none**

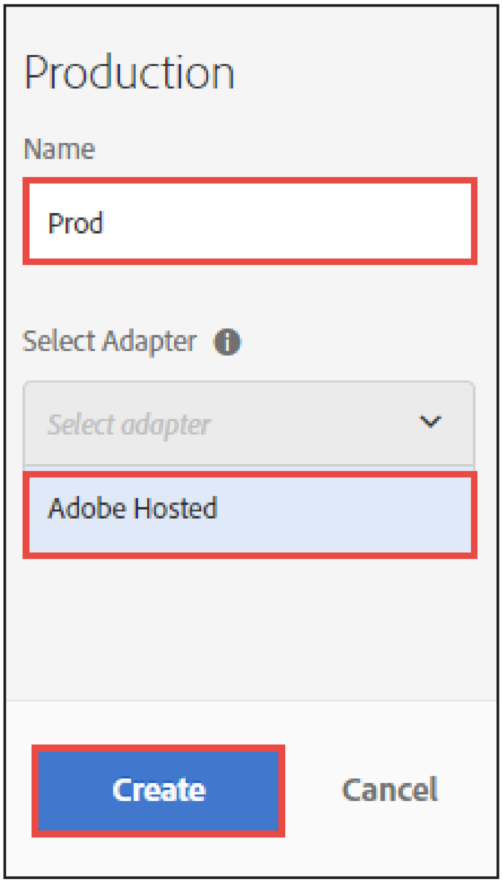
Production

The production environment is for the live, customer-facing site. The production environment has an embed code which should be deployed on the production site. A company can only have one production environment.

Current Environment: **none**

3. In the **Name** field, enter an appropriate name. For example, enter Prod.
4. From the **Select adapter** drop-down list, select the adapter you created in the previous task. For example, Adobe Hosted.

5. Click **Create**, as shown:



Production

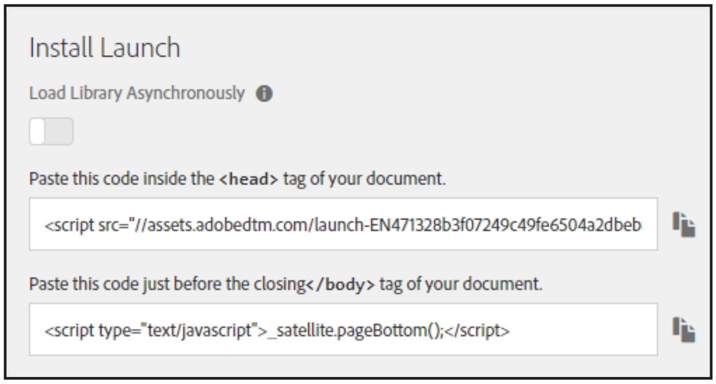
Name

Prod

Select Adapter

Adobe Hosted

Notice the header and footer embed codes at the bottom of the page. These codes determine the loading of JavaScript and page content on your site.



Install Launch

Load Library Asynchronously

Paste this code inside the **<head>** tag of your document.

```
<script src="//assets.adobedtm.com/launch-EN471328b3f07249c49fe6504a2dbeb
```

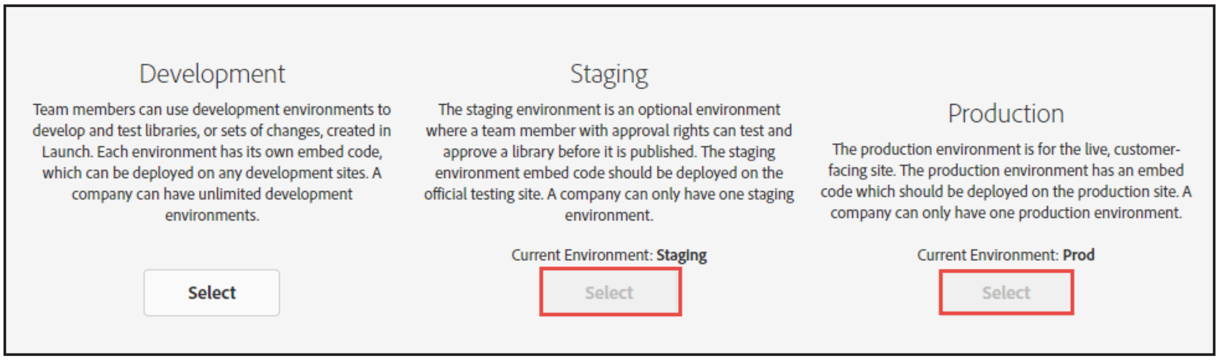
Paste this code just before the closing**</body>** tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

6. Click **Close**.
7. To set up a staging environment, click **Add Environment** and click **Select** under **Staging.**
8. In the **Name** field, enter an appropriate name. For example, enter **Staging**.
9. From the **Select adapter** drop-down list, select the adapter you created in the previous task. For example, select Adobe Hosted.
10. Click **Create.** Again, notice that the embed codes are displayed at the bottom of the page.

11. Click **Close**.
12. Click **Add Environment** to set up development environments.
13. Click **Select** under **Development.**
14. In the **Name** field, enter an appropriate name. For example, enter Dev.
15. From the **Select adapter** drop-down list, select the adapter you created in the previous task. For example, select Adobe Hosted.
16. Click **Create**.
17. Click **Close**.

Notice that the **Select** buttons under Staging and Production are grayed out. This is because, you can create only one staging environment and one production environment. However, you can create multiple development environments.



Development

Team members can use development environments to develop and test libraries, or sets of changes, created in Launch. Each environment has its own embed code, which can be deployed on any development sites. A company can have unlimited development environments.

Staging

The staging environment is an optional environment where a team member with approval rights can test and approve a library before it is published. The staging environment embed code should be deployed on the official testing site. A company can only have one staging environment.

Current Environment: **Staging**

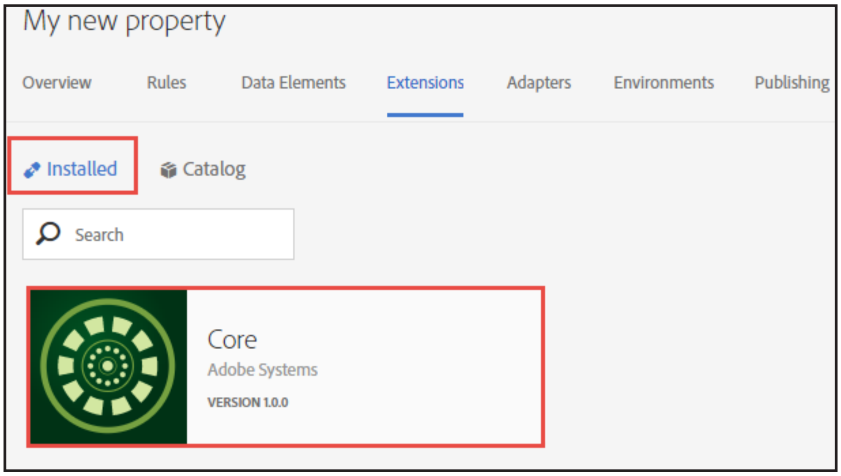
Production

The production environment is for the live, customer-facing site. The production environment has an embed code which should be deployed on the production site. A company can only have one production environment.

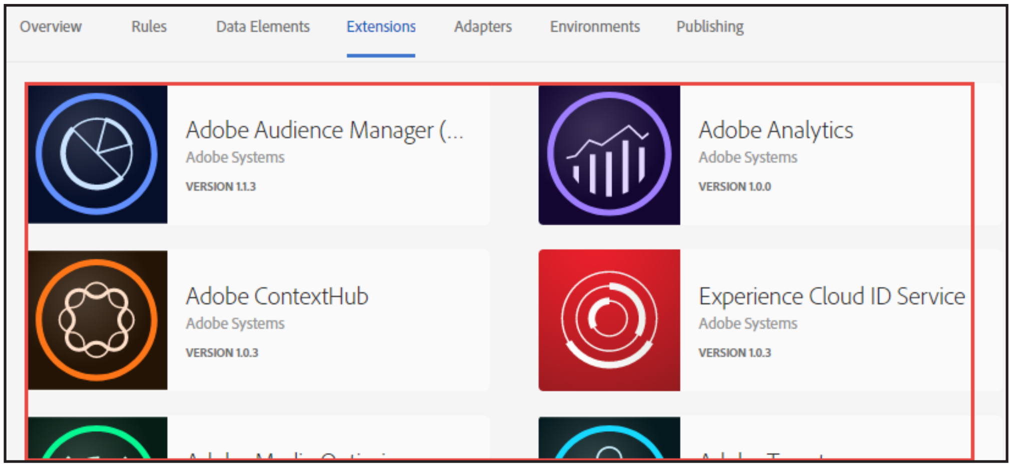
Current Environment: **Prod**

Task 5: Add extensions

1. Open the **Extensions** tab. Notice the default Core extension under the **Installed** tab.

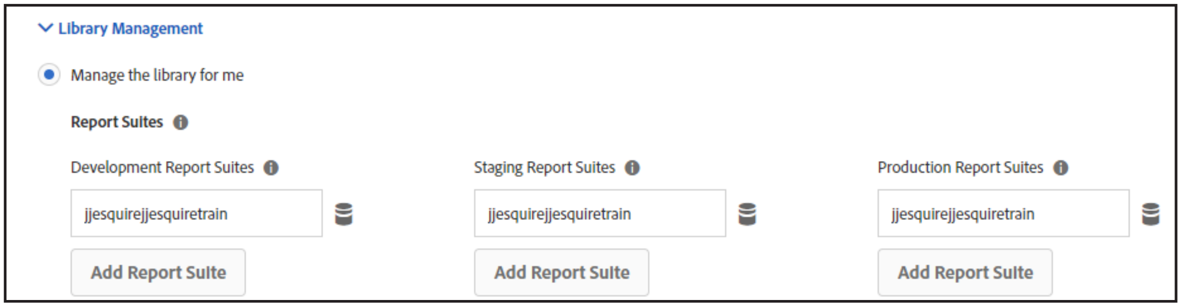


2. Click the **Catalog** tab to access and install additional extensions, as shown:



3. To install the Adobe Analytics extension, hover the cursor over the extension and click **Install**. The Analytics Extension configuration page opens.

4. Under **Library Management**, specify report suite to which your data should be sent. You can type the report suite ID in the field or click the database icon next to each report suite field to select a data element. You can add multiple report suites by clicking **Add Report Suite**. For this task, enter **jjesquirejjesquiretrain** for Development, Staging, and Production Report Suites fields, as shown:



Library Management

● Manage the library for me

Report Suites

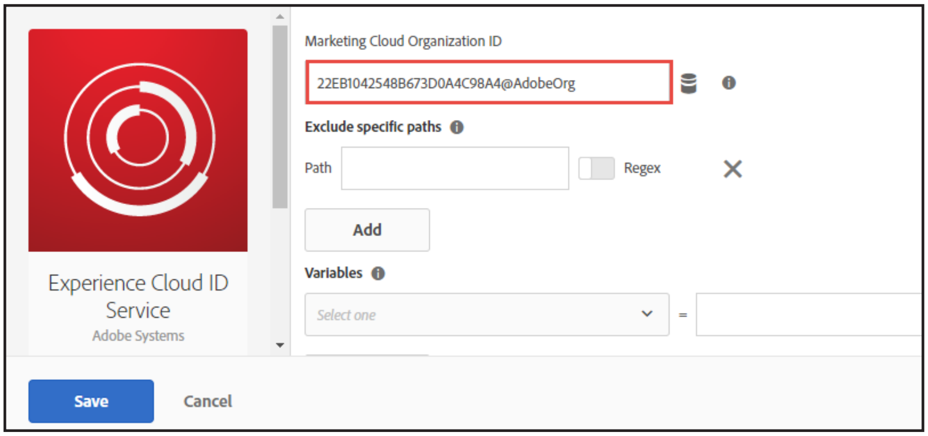
Development Report Suites: jjesquirejjesquiretrain

Staging Report Suites: jjesquirejjesquiretrain

Production Report Suites: jjesquirejjesquiretrain

5. Click **Save**.

6. Open the **Catalog** tab, hover your cursor over the **Experience Cloud ID Service**, and click **Install** to install the Experience Cloud ID Service extension. Notice that the **Marketing Cloud Organization ID** is automatically generated, as shown.



Marketing Cloud Organization ID: 22EB1042548B673D0A4C98A4@AdobeOrg

Exclude specific paths

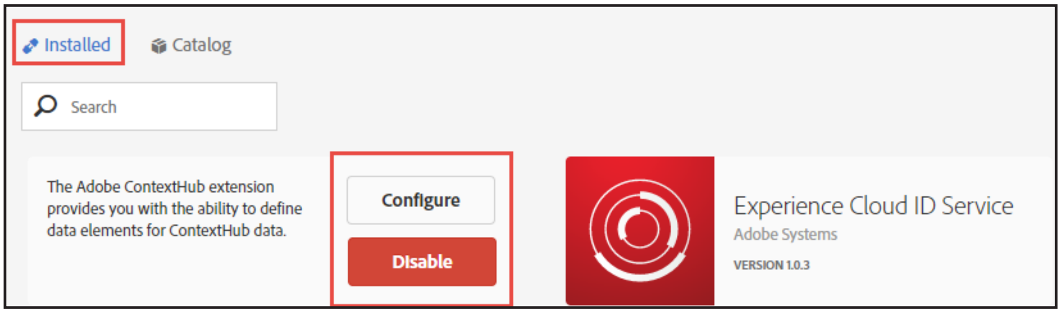
Path: Regex

Variables: Select one =

7. Click **Save**.

8. Similarly, add the **Adobe ContextHub** extension.

9. To configure or disable an extension, hover the cursor over an extension in the **Installed** tab and click **Configure** or **Disable**, as required.

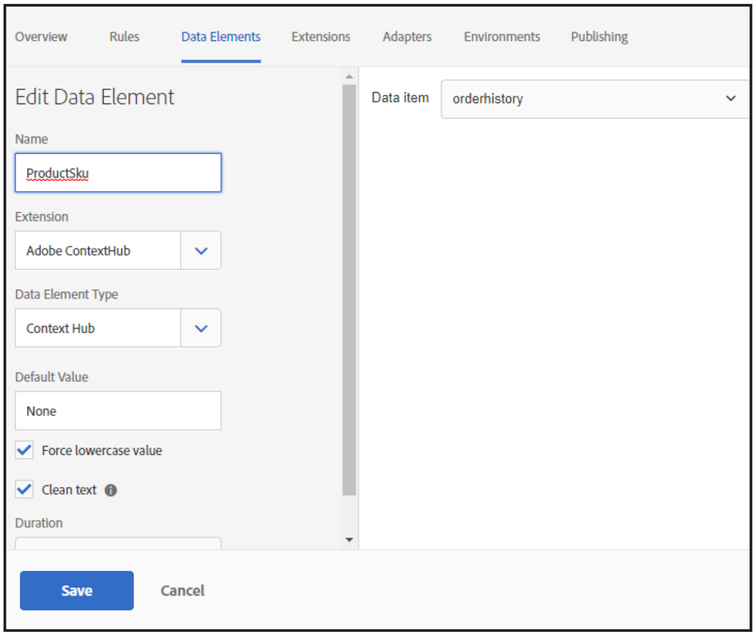


The Adobe ContextHub extension provides you with the ability to define data elements for ContextHub data.

Task 6: Create data elements

1. On the **Data Elements** tab, click **Create New Data Element**. The **Edit Data Element** page opens.
2. Add a data element for product SKU, as shown:
   a. In the **Name** field, enter ProductSku.
   b. From the **Extension** drop-down, select **Adobe ContextHub** extension.
   c. From the **Data Element Type** drop-down, select **Context Hub** parameter.
   d. From the **Data Item** drop-down on the right, select a value. For example, select orderhistory.
   e. Optionally, enter a value in the **Default value** field.
   f. Select the **Force lowercase value** and **Clean text** boxes. This option causes Launch to remove beginning and ending whitespace and replace all successive spaces in the middle of return values with a single space. This option is useful to normalize data element values for easier matching.
   g. For **Duration**, retain **Pageview** as a value. Duration here refers to browser-specific time. That is, for how long do you want Launch to remember this value so that you can act on it later.

h. Click **Save**. This will automatically pull order history into the data layer.



Edit Data Element

Name: ProductSku

Extension: Adobe ContextHub

Data Element Type: Context Hub

Default Value: None

☑ Force lowercase value

☑ Clean text

Duration

Data item: orderhistory

3. Similarly, add a page name data element:
   a. Click **Add Data Element** on the **Data Elements** tab.
   b. In the **Name** field, enter **PageName**.
   c. From the **Extension** drop-down list, select the **Adobe ContextHub** extension.
   d. From the **Data Element Type** drop-down list, select the **Context Hub** parameter.
   e. From the **Data Item** drop-down list on the right, select a value. For example, select **pagedata**.**hits**.
   f. Optionally, enter a value in the **Default value** field.
   g. Select the **Force lowercase value** and **Clean text** check boxes.
   h. For **Duration**, retain **Pageview**.
   i. Click **Save**.

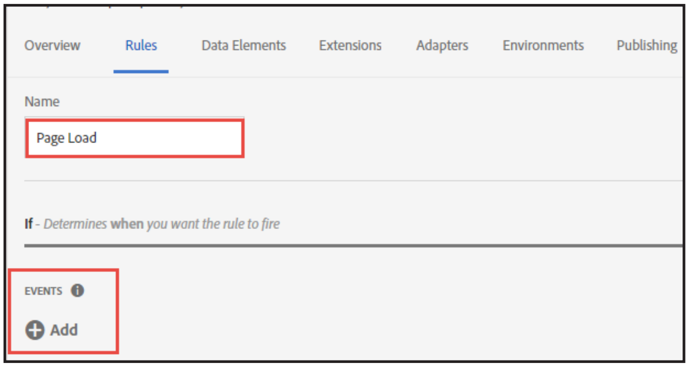
Task 7: Build rules

1. On the **Rules** tab, click **Create New Rule**. The **Create New Rule** page opens.

2. In the **Name** field, type a name. For example, type **Page Load**.

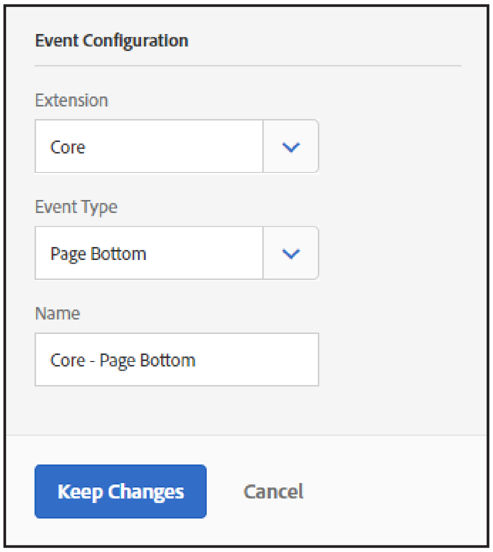
To determine when you want the rule to fire, define events, conditions, and exceptions (IF condition).

3. Click **Add** under **EVENTS,** as shown. The **Event Configuration** page opens.



Name: Page Load

a. For **Extension**, retain the **Core** extension.
b. From the **Event Type** drop-down list, select **Page Bottom**.
c. Optionally, change the **Name**, as required.
d. Click **Keep Changes**.

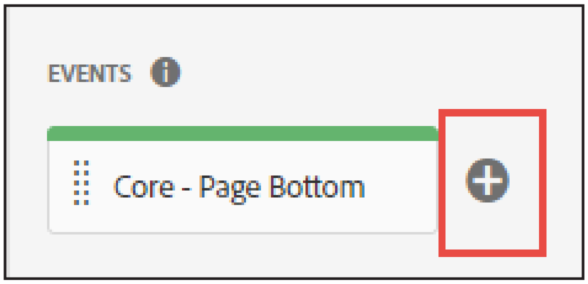


Extension: Core

Event Type: Page Bottom

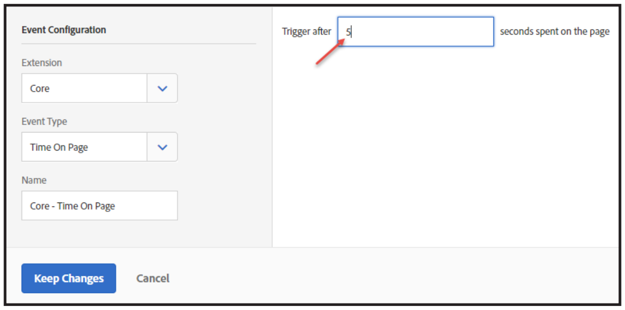
Name: Core - Page Bottom

4. Add another event by clicking the plus sign next to the event you defined now, as shown. The **Event Configuration** page opens.



On this page, enter the following values, as shown:

a. For **Extension**, retain the **Core** extension.
b. From the **Event Type** drop-down list, select **Time on Page**.
c. Enter a value in the **Trigger after...seconds spent on the page** box on the right. For example, enter 5.
d. Optionally, change the name, as required.
e. Click **Keep Changes**.


Event Configuration

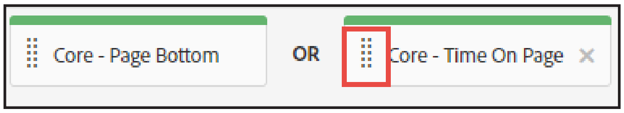
Extension: Core

Event Type: Time On Page

Name: Core - Time On Page

Trigger after 5 seconds spent on the page

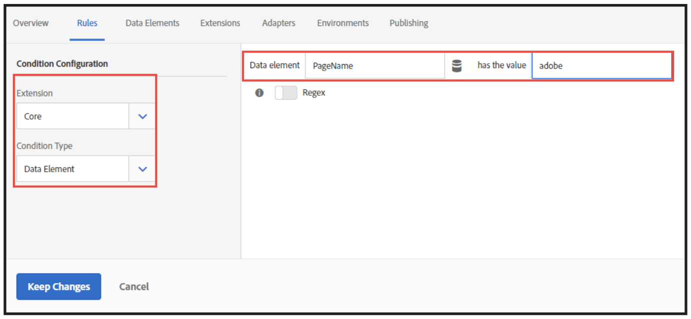
5. Hover the cursor over the dotted lines on the **Core - Time on Page** event. When the cursor turns into a cross with four arrows, drag and drop the event in place of the **Core - page Bottom** event. You can reorganize the events as per your requirement, as shown:



6. To add a condition, click **Add** under **CONDITIONS**. The **Condition Configuration** page opens. On this page, enter the following, as shown:

a. For **Extension**, retain the **Core** extension.
b. From the **Condition Type** drop-down list, select **Data Element.**
c. Click the database icon on the right to select a date element. For example, select **PageName**.
d. Specify a text value in the **has the value** box. For example, specify adobe.

e. Click **Keep Changes**.



Condition Configuration

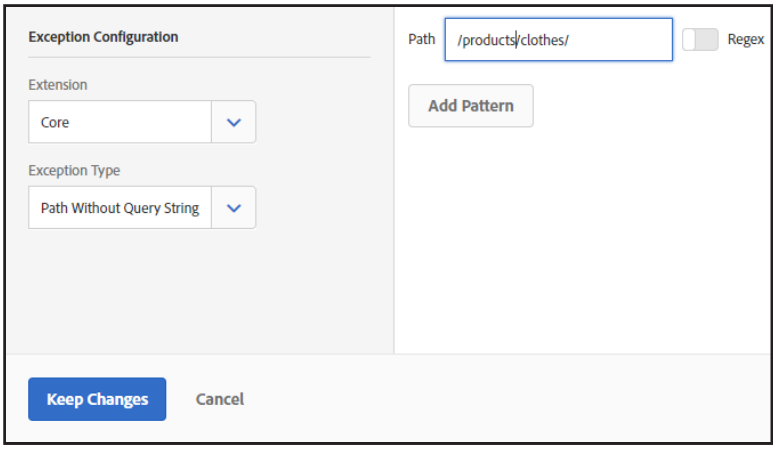
Extension: Core

Condition Type: Data Element

Data element PageName has the value adobe

Regex

7. Next, add an exception, as shown. Click **Add** under **EXCEPTIONS**. The Exceptions Configuration page opens. Enter the following values, as shown:
   a. For **Extension**, retain the **Core** extension.
   b. From the **Exception Type** drop-down, select **Path Without Query String**.
   c. Specify a path on the right. For example, /products/clothes/
   d. Click **Keep Changes**.



Exception Configuration

Extension: Core

Exception Type: Path Without Query String

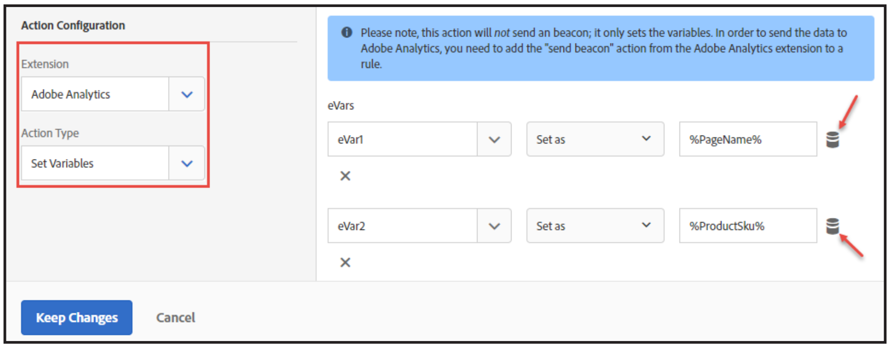
Path: /products/clothes/

Regex

Now that you have specified events, a condition, and an extension, the next step is to specify the actions (THEN condition), as shown.

8. Click **Add** under **ACTIONS**. The Actions Configuration page opens. Specify the following details, as shown:
   a. From the **Extension** drop-down list, select **Adobe Analytics**.
   b. From the **Action Type** drop-down list, select **Set Variables**.
   c. Under **Variables** section on the right, select **eVar1** from the **eVar** drop-down list.
   d. Click the database icon on the right and select **pageName**.
   e. Click **Add eVar**.

   f. Select **eVar2** from the **eVar** drop-down list, click the database icon on the right, and select **ProductSku**.
   g. Click **Keep Changes**.



Action Configuration

Extension: Adobe Analytics

Action Type: Set Variables

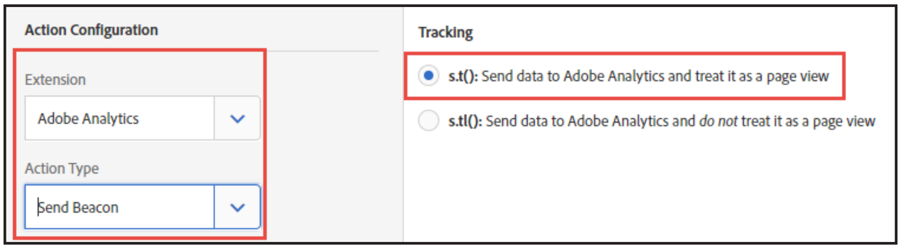
Please note, this action will *not* send an beacon; it only sets the variables. In order to send the data to Adobe Analytics, you need to add the "send beacon" action from the Adobe Analytics extension to a rule.

eVars

eVar1 | Set as | %PageName%

eVar2 | Set as | %ProductSku%

9. Click the plus sign next to the action you defined just now to add another action. The Action Configuration page opens. Specift the following details, as shown:
   a. From the **Extension** drop-down list, select **Adobe Analytics**.
   b. From the **Action Type** drop-down list, select **Send Beacon**.
   c. Under **Tracking**, select the **s.t(): Send data to Adobe Analytics and treat it as a page view** option button.
   d. Click **Keep Changes**.



Action Configuration

Extension: Adobe Analytics

Action Type: Send Beacon

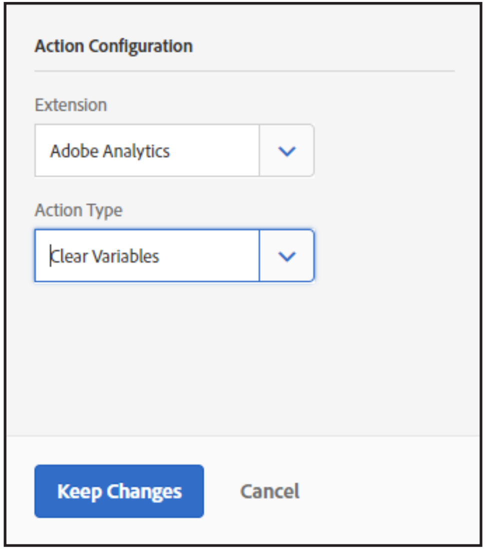
Tracking

● **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

10. Click the plus sign next to the **Send Beacon** action you defined now to add another action.
    a. From the **Extension** drop-down list, select **Adobe Analytics**.
    b. From the **Action Type** drop-down list, select **Clear Variables**.

c. Click **Keep Changes**.

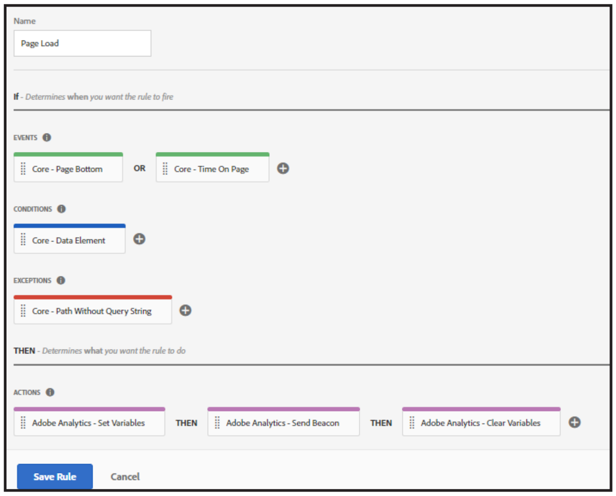


Action Configuration

Extension: Adobe Analytics

Action Type: Clear Variables

Your rule should look similar to the one shown:



Name: Page Load

If - Determines when you want the rule to fire

EVENTS: Core - Page Bottom OR Core - Time On Page

CONDITIONS: Core - Data Element

EXCEPTIONS: Core - Path Without Query String

THEN - Determines what you want the rule to do

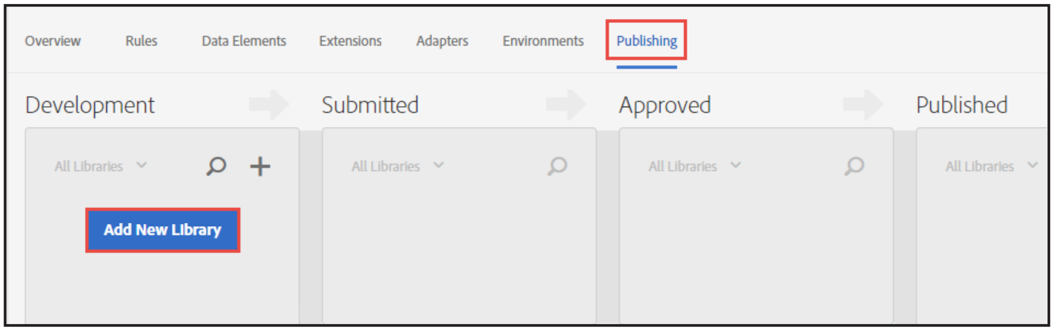
ACTIONS: Adobe Analytics - Set Variables THEN Adobe Analytics - Send Beacon THEN Adobe Analytics - Clear Variables

11. Click **Save Rule**.

Task 8: Publish in Launch

The first step in the publishing process is to add a new library.

1. On the **Publishing** tab, click **Add New Library** under **Development,** as shown. The **Create New Library** page opens.

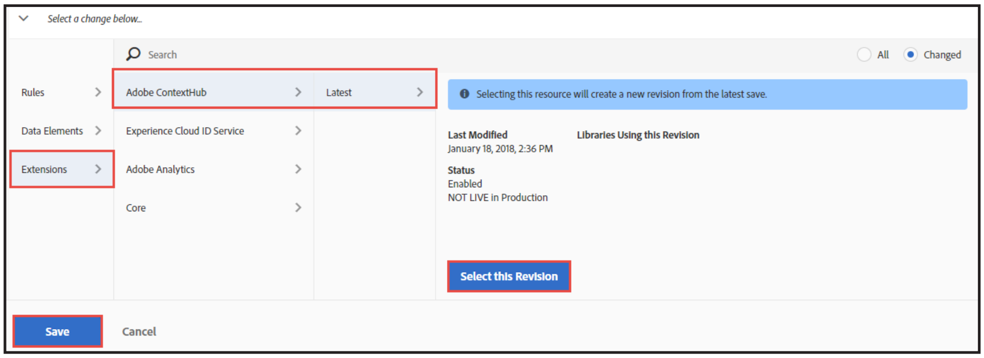


2. In the **Name** field, enter a suitable name. For example, enter New Library.
3. From the **Environment** drop-down list, select one of the development environments you created earlier. For example, select Dev1.

This library contains all rules, data elements, and extensions. You can add or change resources under **NEW RESOURCES** section.

4. To add an updated extension:
   a. Click **Add a Resource**.
   b. In the left panel, click **Extensions** > **Adobe ContextHub** > **Latest** > **Select this Revision**. This creates a new revision that includes all latest updates.

c. Click **Save** to save the library, as shown:



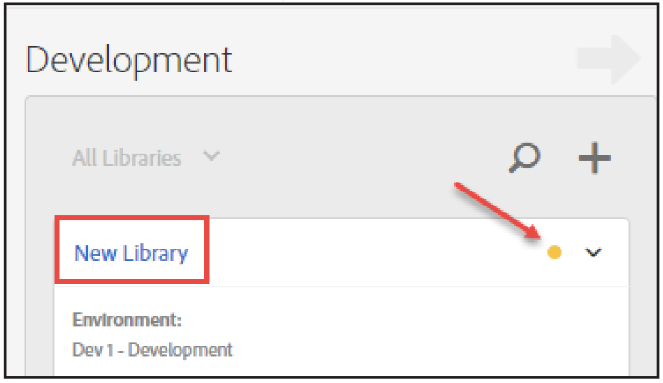
Selecting this resource will create a new revision from the latest save.

Last Modified
January 18, 2018, 2:36 PM

Libraries Using this Revision

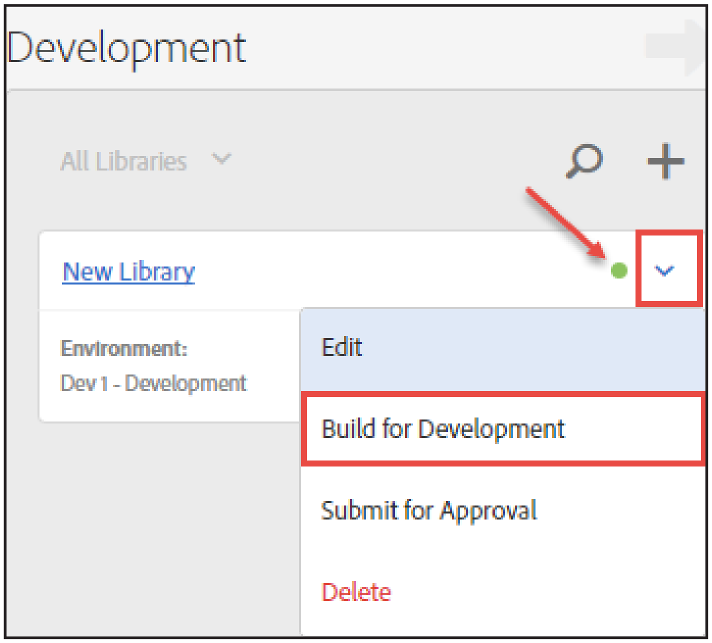
Status
Enabled
NOT LIVE in Production

Notice that the library is saved in the **Development** section, as shown. The yellow dot next to the library name indicates that there is no build for development. The next step is to create a build. This will send the code to the development server.



Environment:
Dev 1 - Development

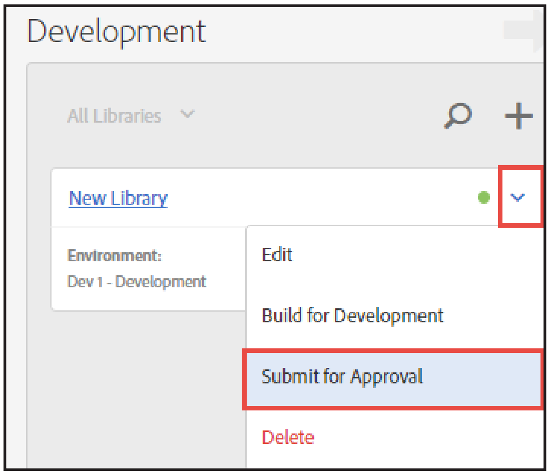
5. To create a build, click the **New Library** drop-down arrow, as shown.
6. Click **Build for Development,** as shown.



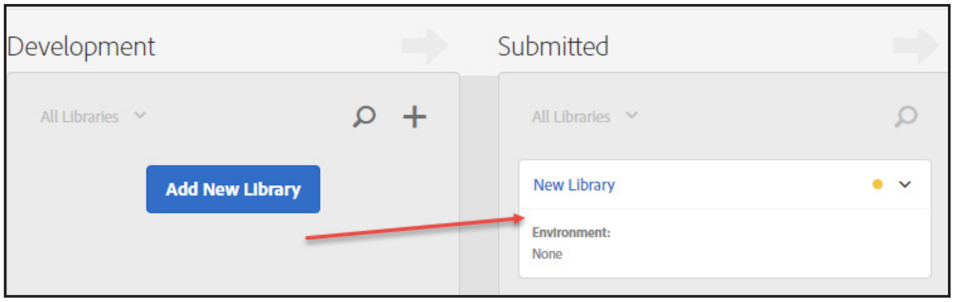
Environment:
Dev 1 - Development

Notice that the yellow dot has now turned green. This indicates that the library now has a build.

7. Submit the library for approval by clicking **New Library** > **Submit for Approval** > **Submit**, as shown:



Notice that your library moved from **Development** step to the **Submitted** step, as shown. You can have only one library at a time in the **Submitted** step.



Additional Information

Configure Extensions in Launch

Introduction

An extension is an integration that adds new options for the tags you deploy to your sites. You can use extensions to run surveys, manage privacy, and gain specific insights into user behavior. Launch includes extensions built by Adobe and by third-parties, such as Facebook, Twitter, and LinkedIn.

Objectives

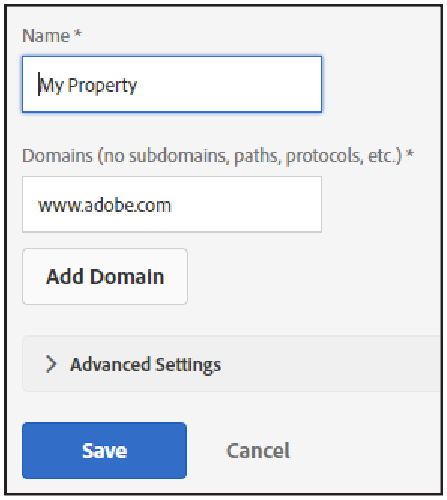
After completing this chapter, you will be able to:

- Configure data elements and rules using the Core extension
- Install Analytics extension
- Configure rules using the Analytics extension
- Configure rules using the Experience Cloud ID extension

Demo: Configure data elements and rules using the Launch Core extension

The element types available to you when building data elements and rules depend on the extensions you have installed. The Core extension provides a robust set of default event conditions and data element types, which you can use to configure data elements and rules. The Core extension is preinstalled in every new property.

1. Log in to Launch using the credentials provided to you by your instructor. The **Properties** page opens.
2. Click **Create a New Property**. The **Create Property** page opens.
3. In the **Name** field, enter a name, as shown. For example, enter My Property.
4. In the **Domains** field, enter a domain, as shown. For example, enter, www.adobe.com.
5. Click **Save**. The property you created is listed on the **Properties** page.

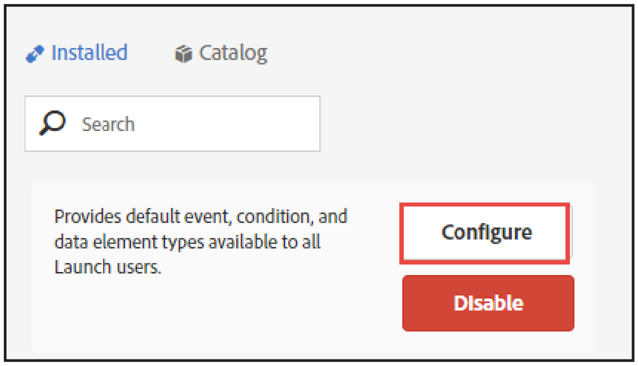


Name *: My Property

Domains (no subdomains, paths, protocols, etc.) *: www.adobe.com

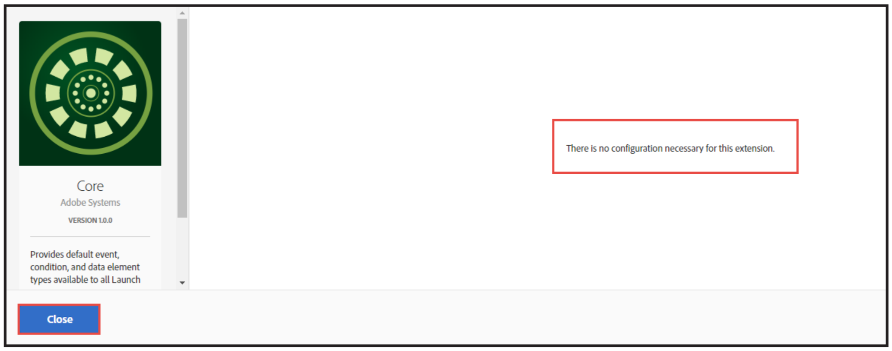
6. Click **My Property** to open it. The **Overview** page opens.
7. Click the **Extensions** tab. Notice that the Core extension is preinstalled.

8. Hover your cursor over the Core extension and click **Configure**, as shown. The **Core Extension** page opens.



Provides default event, condition, and data element types available to all Launch users.

9. Notice that this extension does not require additional configuration.



Provides default event, condition, and data element types available to all Launch

There is no configuration necessary for this extension.

10. Click **Close** to return to the **Extensions** tab.

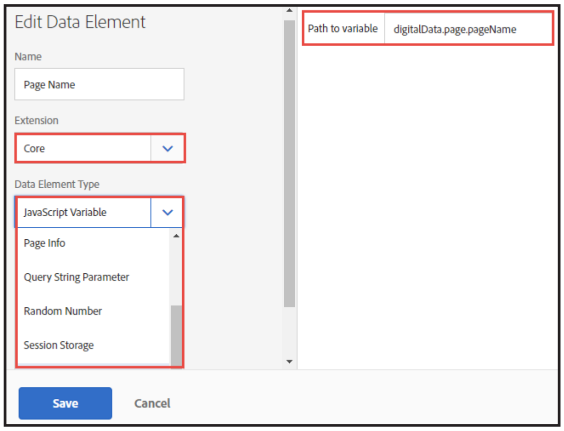
Launch uses data elements to collect, organize, and deliver data. The core extension offers various data element types, which you will use to create data elements.

11. On the **Data Elements** tab, click **Create New Data Element**. The **Create New Data Element** page opens.
12. In the **Name** field, enter a name. For example, enter **Page Name**.
13. Notice that the **Extension** field is already populated with the Core extension. Retain the same.
14. Click the **Data Element Type** drop-down arrow. Notice different element types. Depending on your requirement, you will select and configure one of the data types. The following table describes data element types and settings available:

Data Element Type	Settings
Cookie	Provide a cookie name.
Custom Code	Enter the code in the editor.
DOM Attribute	Indicate the DOM element matching the CSS selector.
Local Storage	Specify the local storage item name.
Page Info	Select an attribute from the drop down list.
Query String Parameter	Specify the URL query string parameter name.
Random Number	Indicate minimum and maximum values.
Session Storage	Specify the session storage item name.
JavaScript Variable	Indicate the path to variable.

15. For this task, select **JavaScript Variable**.

16. In the **Path to Variable** field, type digitalData.page.pageName, as shown:



Edit Data Element

Name: Page Name

Extension: Core

Data Element Type: JavaScript Variable

Path to variable: digitalData.page.pageName

17. Retain Pageview in the **Duration** field.

18. Click **Save**. The data element is created.

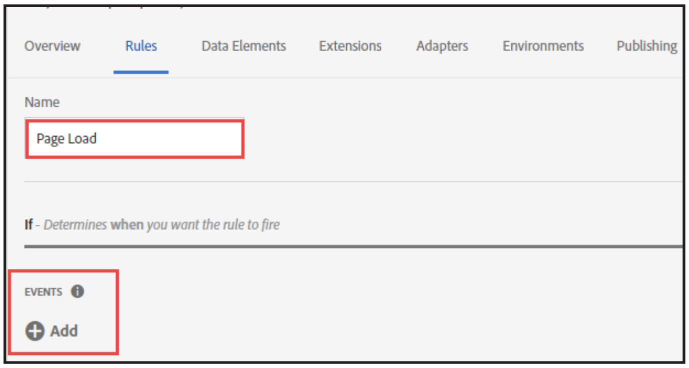
The Core extension offers various event types, condition types, exception types, and action types, which you will use to build rules. In Launch, rules are used to integrate disparate products into a single solution. Launch searches for user interaction and associated data, and when the criteria defined in your rules are met, extensions, script, or HTML are triggered.

19. On the **Rules** tab, click **Create New Rule**. The **Create New Rule** page opens.

20. In the **Name** field, type a name. For example, type **Page Load**.

To determine when you want the rule to trigger, define events, conditions, and exceptions (the IF part).

21. Click **Add** under **EVENTS,** as shown. The **Event Configuration** page opens.



Name: Page Load

If - Determines when you want the rule to fire

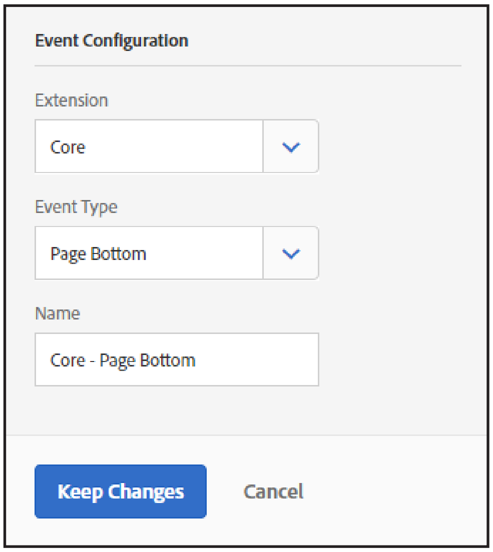
a. Notice that the **Extension** field is prepopulated with the **Core** extension. Retain the same.

b. Click the **Event Type** drop-down list. Scroll down and notice the various event types available under the Core extension. You will use one of these types based on your requirement. All event types available under the Core extension and their settings are described in the Appendix section of this document.

c. For this task, select **Page Bottom**.

d. Optionally, change the **Name**, as per your preference.

e. Click **Keep Changes,** as shown:



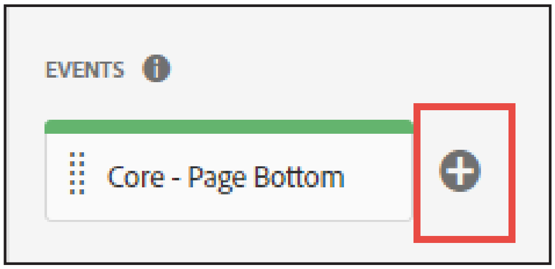
Event Configuration

Extension: Core

Event Type: Page Bottom

Name: Core - Page Bottom

22. Add another event by clicking the plus sign next to the event you defined now, as shown. The **Event Configuration** page opens.



Enter the following values, as shown:

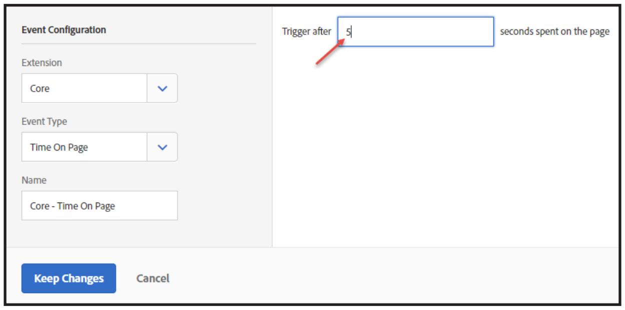
a. For **Extension**, retain the **Core** extension.

b. From the **Event Type** drop-down list, select **Time on Page**.

c. Enter a value in the **Trigger after...seconds spent on the page** box on the right. For example, enter 5.

d. Optionally, change the name, as per your preference.

e. Click **Keep Changes**.



Event Configuration

Extension: Core

Event Type: Time On Page

Name: Core - Time On Page

Trigger after 5 seconds spent on the page

23. To add a condition, click **Add** under **CONDITIONS**. The **Condition Configuration** page opens. Enter the following details, as shown:

a. Notice that the **Extension** field is prepopulated with the **Core** extension. Retain the same.

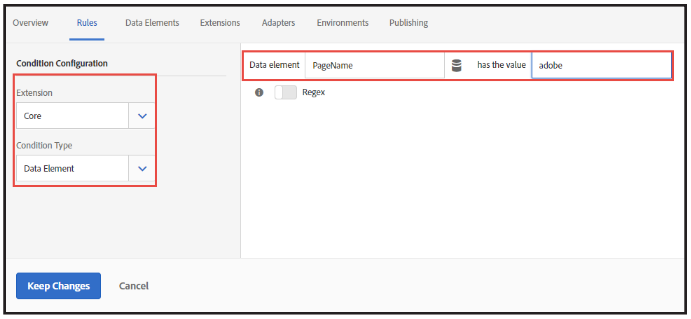
b. Click the **Condition Type** drop-down list. Scroll down and notice the different condition types available under the Core extension. You will use one of these types based on your requirement. All condition types available under the Core extension and their settings are described in the Additional Information section of this document.

c. For this task, select **Data Element.**

d. Click the database icon on the right to select a date element. For example, select **PageName**.

e. Specify a text value in the **has the value** box. For example, specify adobe.

f. Click **Keep Changes,** as shown:



Condition Configuration

Extension: Core

Condition Type: Data Element

Data element PageName has the value adobe

Regex

24. Add an exception, as shown. Click **Add** under **EXCEPTIONS**. The **Exceptions Configuration** page opens. Enter the following values, as shown:

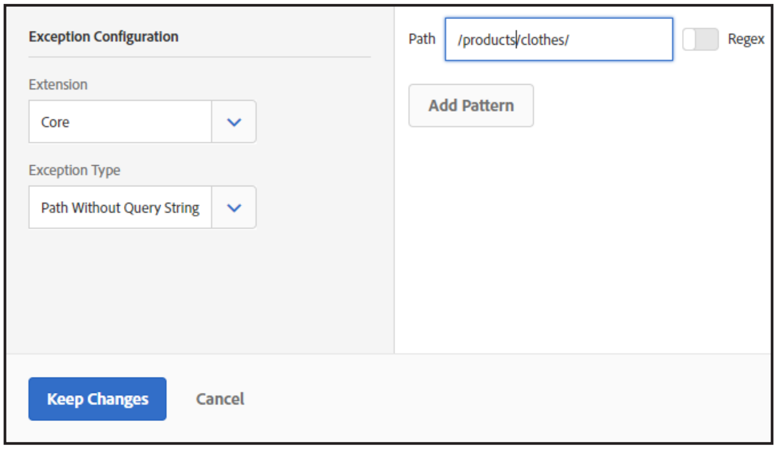
a. For **Extension**, retain the **Core** extension.

b. From the **Exception Type** drop-down. Scroll down and notice the different exception types available under the Core extension. You will use one of these types based on your requirement. All exception types available under the Core extension and their settings are described in the Appendix section of this document.

c. For this task, select **Path Without Query String**.

d. Specify a path on the right. For example, specify /products/clothes/

e. Click **Keep Changes**.



Exception Configuration

Extension: Core

Exception Type: Path Without Query String

Path: /products/clothes/

25. You have specified events, a condition, and an extension. Next, you need to specify the actions (THEN condition). The Core extension contains a Custom Code action type for executing user-provided JavaScript or HTML. Specify the code that runs after the event is triggered and conditions or exceptions are evaluated, as shown:

a. For **Extension**, retain the **Core** extension.

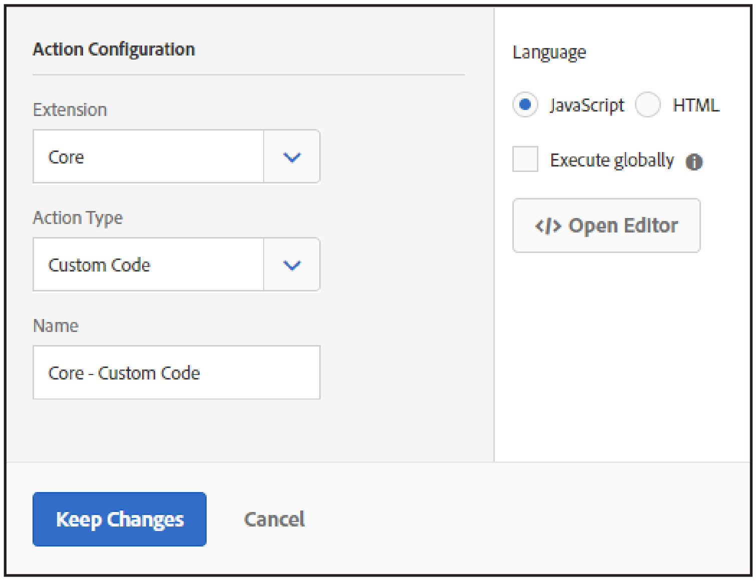
a. From the **Action Type** drop down list, select **Custom Code**.

b. Select the language used to define the action. Select JavaScript or HTML.

c. Enter the code in the Editor, and save it

d. Click **Keep Changes**.

e. Click **Save Rule**.



Action Configuration

Extension: Core

Action Type: Custom Code

Name: Core - Custom Code

Language: ● JavaScript ○ HTML

□ Execute globally

Analytics Extension

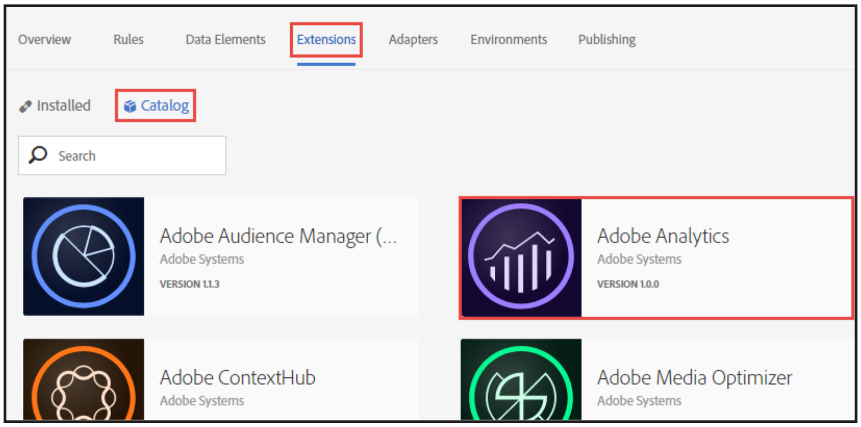
The Adobe Analytics extension supports client-side Adobe Analytics implementations using AppMeasurement.js. When building rules, the Analytics extension provides the following action types:

- Set Variables
  - Set eVars: Select an eVar from the drop down list, specify whether you want to set the eVar value (Set As) or copy (Duplicate From) another eVar, and provide a value (or select the eVar you want to duplicate). You can add multiple eVars by clicking the Add eVar button.
  - Set Props: Select a prop from the drop down list, specify whether you want to set the prop value (Set As) or copy (Duplicate From) another prop, and provide a value (or select the prop you want to duplicate). You can add multiple props by clicking the Add prop button.
  - Add Events: Select an event from the drop down list. Optionally, select or specify a data element used for event serialization. You can add multiple events.
  - Hierarchy: The hierarchy variable is used to determine the location of a page in your site's hierarchy or page structure. Hierarchies reflect the natural organization of the site. Specify each level in the hierarchy. If required, configure additional hierarchies.
  - Other Information: Specify other information used by your pages, such as, Page Name, Page URL, Server, Channel, Referrer, Campaign, State, Zip, Transaction ID.
  - Custom Page Code: Use the editor to specify your custom page code.
- Send Beacon
  - Increment a pageview - s.t(): Select this option if you want to increment a pageview.
  - Do not increment a pageview - s.t(): Select if you do not want to increment a pageview. Select a link type and set the parameter for the selected link.
- Clear Variables
  - No additional configuration is required if the Clear Variables action type is selected.

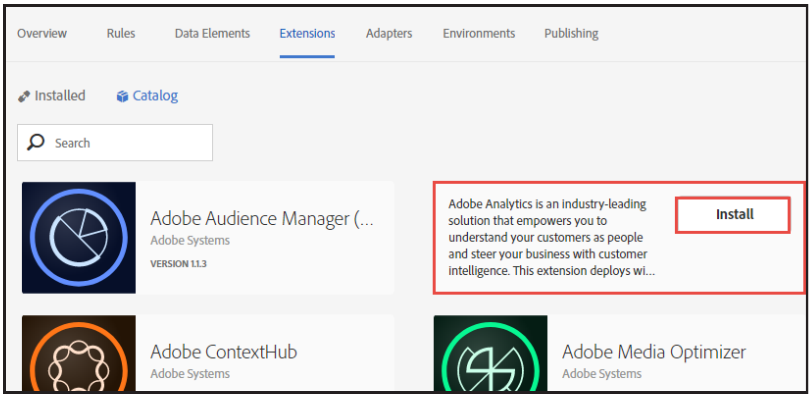
Exercise 1: Configure the Analytics Extension

Task 1: Install Analytics Extensions

1. On the **Extensions** tab, click **Catalog** to access and install additional extensions, as shown:

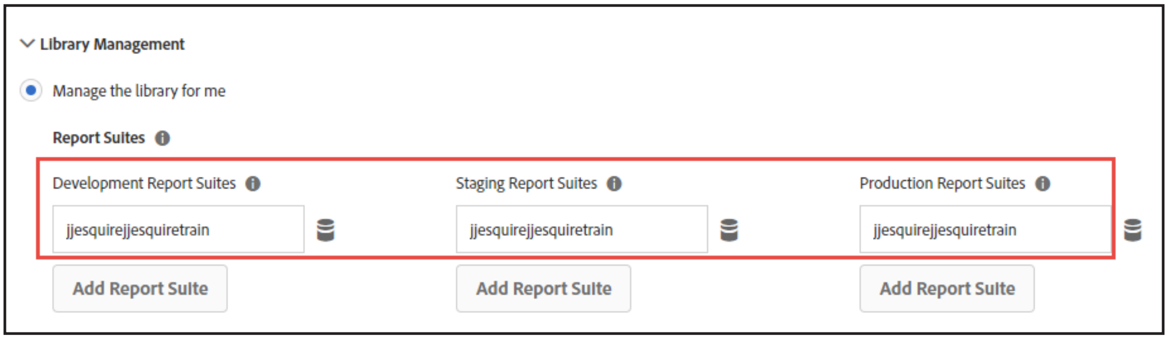


2. To install the Adobe Analytics extension, hover the cursor over the extension and click **Install**, as shown. The Analytics Extension configuration page opens.



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

3. Under **Library Management**, specify report suite to which your data should be sent. You can type the report suite ID in the field or click the database icon next to each report suite field to select a data element. You can add multiple report suites by clicking **Add Report Suite**. For this task, type **jjesquirejjesquiretrain** as the report suite under Development Report Suite, Staging Report Suite, and Production Report Suite fields, as shown.



Library Management

● Manage the library for me

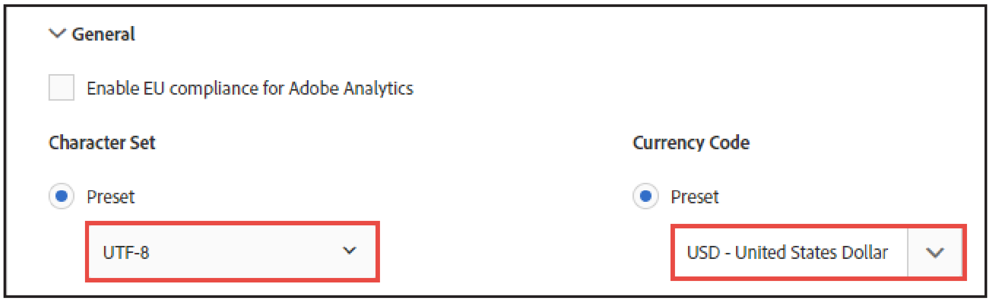
Report Suites

Development Report Suites: jjesquirejjesquiretrain

Staging Report Suites: jjesquirejjesquiretrain

Production Report Suites: jjesquirejjesquiretrain

4. From the **General** section, set the **Character Set** and **Currency Code**, as shown:
   a. Select UTF-8 from the **Character Set Preset** drop down list.
   b. Select a currency code from the **Currency Code Preset** drop-down list. For example, select USD - United States Dollar.



General

□ Enable EU compliance for Adobe Analytics

Character Set: ● Preset — UTF-8

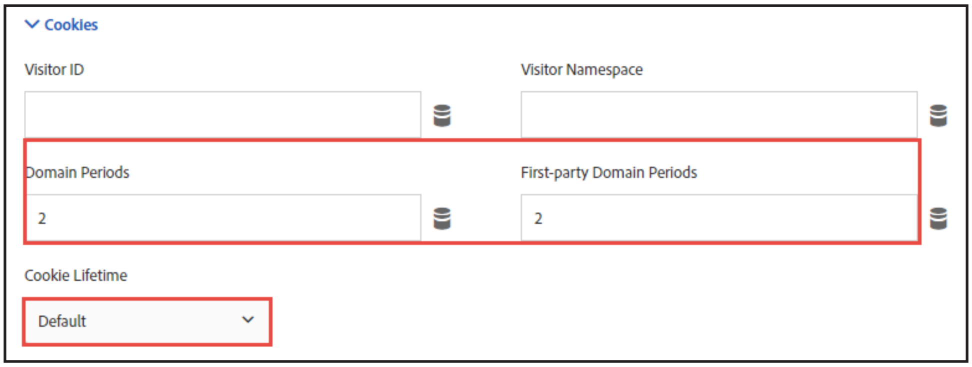
Currency Code: ● Preset — USD - United States Dollar

5. Leave the fields under Global Variables blank. As a best practice, manage variables in your rules instead. However, depending on your implementation, you can set eVars, props, and hierarchies under **Global Variables**.

6. In the **Link Tracking** section, select the **Track download links** and **Keep URL Parameters** check boxes.

7. From the **Cookies** section, type 2 as a value for both **Domain Periods** and **First-party Domain Periods**.

8. From the **Cookie Lifetime** drop down list, select Default, as shown:



Cookies

Visitor ID

Visitor Namespace

Domain Periods: 2

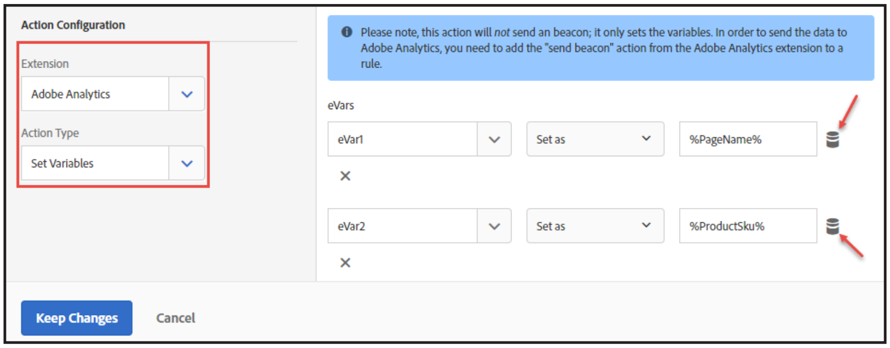
First-party Domain Periods: 2

Cookie Lifetime: Default

9. Click **Save**. The Analytics extension is installed.

Task 2: Configure rules using the Adobe Analytics extension

1. On the **Rules** tab, click the Page Load rule you created earlier. The Edit Rule page opens.
2. Click **Add** under **ACTIONS**. The Actions Configuration page opens. Specify the following details, as shown:
   a. From the **Extension** drop-down list, select **Adobe Analytics**.
   b. From the **Action Type** drop-down list, select **Set Variables**.
   c. Under **Variables** section on the right, select **eVar1** from the **eVar** drop-down list.
   d. Click the database icon on the right and select **pageName**.
   e. Click **Add eVar**.
   f. Select **eVar2** from the **eVar** drop-down list, click the database icon on the right, and select **ProductSku**.
   g. Click **Keep Changes**.



Action Configuration

Extension: Adobe Analytics

Action Type: Set Variables

Please note, this action will *not* send an beacon; it only sets the variables. In order to send the data to Adobe Analytics, you need to add the "send beacon" action from the Adobe Analytics extension to a rule.

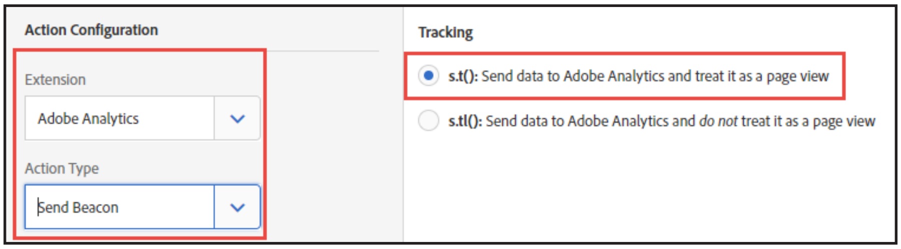
eVars

eVar1 | Set as | %PageName%

eVar2 | Set as | %ProductSku%

3. Click the plus sign next to the action you defined just now to add another action. The **Action Configuration** page opens. Specify the following details, as shown:
   a. From the **Extension** drop-down list, select **Adobe Analytics**.
   b. From the **Action Type** drop-down list, select **Send Beacon**.
   c. Under **Tracking**, select the **s.t(): Send data to Adobe Analytics and treat it as a page view** option button.

d. Click **Keep Changes**.



Action Configuration

Extension

Adobe Analytics

Action Type

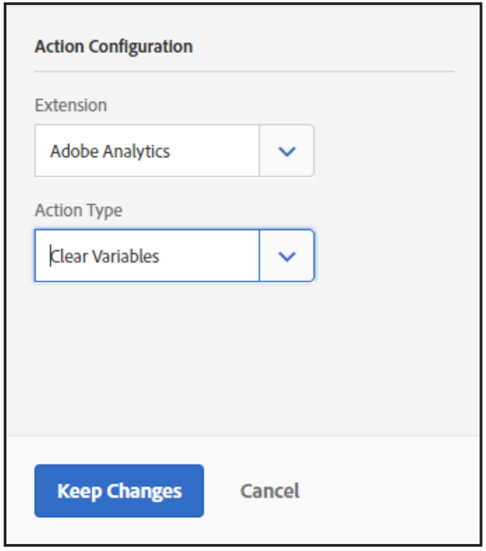
Send Beacon

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

4. Click the plus sign next to the **Send Beacon** action you defined now to add another action. On the Action Configuration field, specify the following, as shown:
   a. From the **Extension** drop-down list, select **Adobe Analytics**.
   b. From the **Action Type** drop-down list, select **Clear Variables**.
   c. Click **Keep Changes**.



Action Configuration

Extension

Adobe Analytics

Action Type

Clear Variables

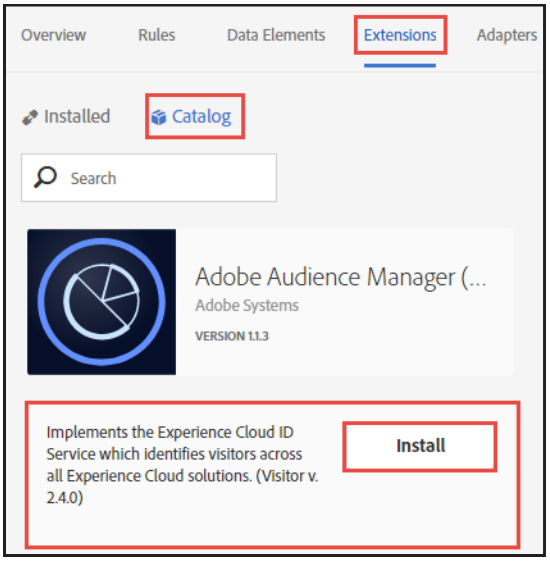
5. Click **Save Rule**. The rule is configured.

Demo: Configure rules using the Experience Cloud ID extension

By using the Experience Cloud ID extension, you can create and store unique identifiers for your site visitors. It consists of two parts:

- The extension configuration where the settings of VisitorAPI.js are managed
- A rule action-type called 'Set Customer ID', which can send multiple customer IDs to Adobe solutions for integrations with Customer Relationship Management (CRM) systems

1. To install the Experience Cloud ID extension, open your property and open the **Extensions** tab.
2. On the **Catalog** tab, hover the cursor over the Experience Cloud ID extension, and click **Install**, as shown. The Extension configuration page opens.

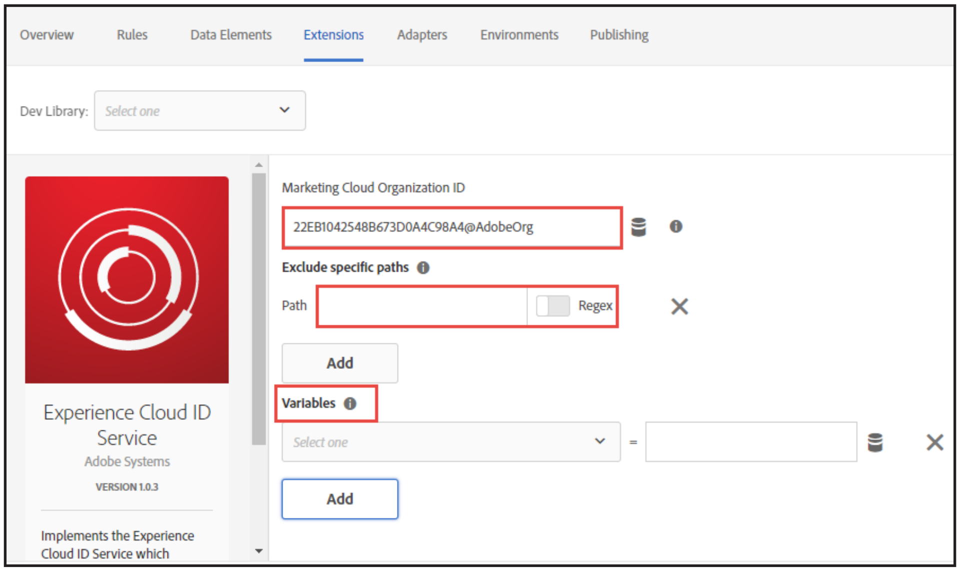


Implements the Experience Cloud ID Service which identifies visitors across all Experience Cloud solutions. (Visitor v. 2.4.0)

Notice that the Marketing Cloud Organization ID is automatically generated. The ID is a 24-character, alpha-numeric string followed by @AdobeOrg.

3. Optionally, you can exclude specific paths. The Experience Cloud ID does not load if the URL matches the path you have specified in this field. Optionally, enable the Regex option if it is a regular expression. You can add multiple paths by clicking the Add button in this field.

4. Optionally, you can configure variables. The name-value pairs are set as Experience Cloud ID instance properties. Use the Variables drop-down list to select a variable, and type or select a value in the value field. You can add multiple variables by clicking the Add button under the Variable field, as shown:



Marketing Cloud Organization ID: 22EB1042548B673D0A4C98A4@AdobeOrg

Experience Cloud ID Service
Adobe Systems
VERSION 1.0.3

Implements the Experience Cloud ID Service which

5. For this task, retain the default options and click **Save**. The Experience Cloud ID extension is created.

When building rules, the Experience Cloud ID extension provides the Set Customer ID action type, which is used to set one or more customer IDs.

6. On the **Data Elements** tab, click **Add data element**. The **Create New Data Element** page opens.
7. In the **Name** field, enter **Authentication State**.
8. In the **Extension** field, retain the **Core Extension**.
9. From the **Data Element Type** drop-down list, select **Custom Code**.
10. Click **Open Editor** to include your custom code. The editor opens
11. Enter the following code in the editor:

```
if (digitalData.user[0].profile[0].attributes.loggedIn)
     return “logged in”
else
     return “logged out”
```

12. Click **Save**. The editor closes.
13. Click **Save** to create the data element.

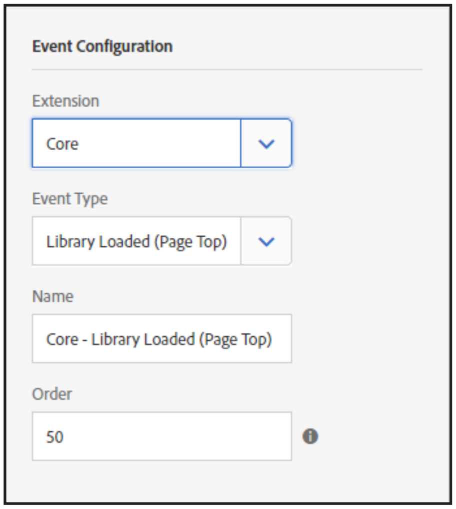
14. On the **Rules** tab, click **Add Rule**. The **Add Rule** page opens.

15. In the **Name** field, enter a suitable name. For example, enter **All Pages - Top - Auth**.

16. Click **Add** under **EVENTS**. The **Event Configuration** page opens.

17. For **Extension**, retain the **Core** extension.

18. From the **Event Type** drop down list, select **Library Loaded (Page Top)**, as shown.



Event Configuration

Extension: Core

Event Type: Library Loaded (Page Top)

Name: Core - Library Loaded (Page Top)

Order: 50

19. Click **Keep Changes**. The event is configured.

20. Click **Add** under **CONDITIONS**. The **Condition Configuration** page opens.

21. For **Extension**, retain the **Core** extension.

22. From the **Condition Type** drop-down list, select **Data Element**.

23. In the **Data Element** field on the right, enter the name of the data element you created earlier. For example, enter Authentication State. You can also select the data element by clicking the database icon next to the field.

24. In the **has the value** field, enter **logged in**. The rule will only fire if the specified data element returns this value.

25. Click **Keep Changes**. The condition is configured.

26. Click **Add** under **ACTIONS**. The **Action Configuration** page opens.

27. From the **Extension** drop down list, select **Experience Cloud ID**, as shown. The Experience Cloud ID extension provides the Set Customer ID action type, which is used to set customer IDs.

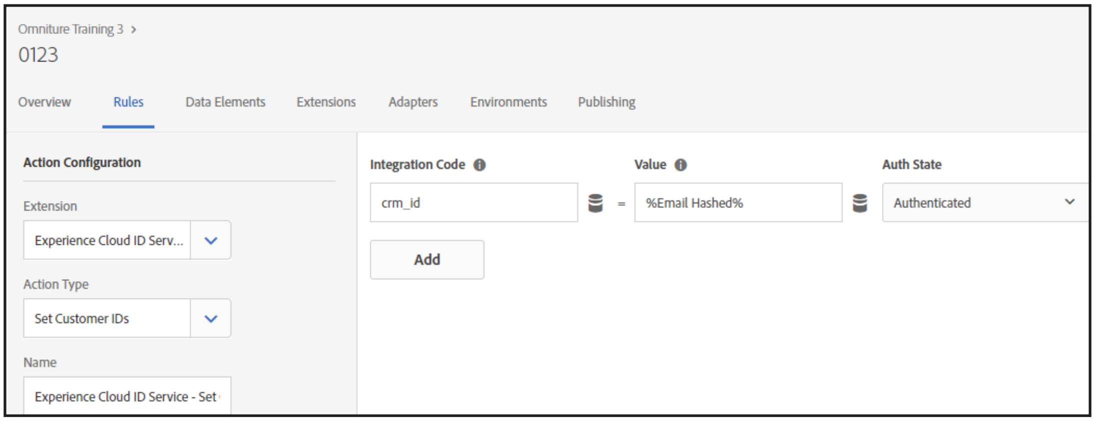
28. From the **Action Type** drop down list, select **Set Customer IDs**, as shown,

29. In the **Integration Code** field, enter crm_id, as shown.

30. In the **Value** field, enter the data element you created earlier. For example, enter **Authentication State**.

31. From the **Auth State** drop down list, select **Authenticated**, as shown.

32. Click **Keep Changes**.



Extension: Experience Cloud ID Serv...

Action Type: Set Customer IDs

Name: Experience Cloud ID Service - Set

Integration Code: crm_id = Value: %Email Hashed%

Auth State: Authenticated

33. Click **Save** to save the rule.

Appendix

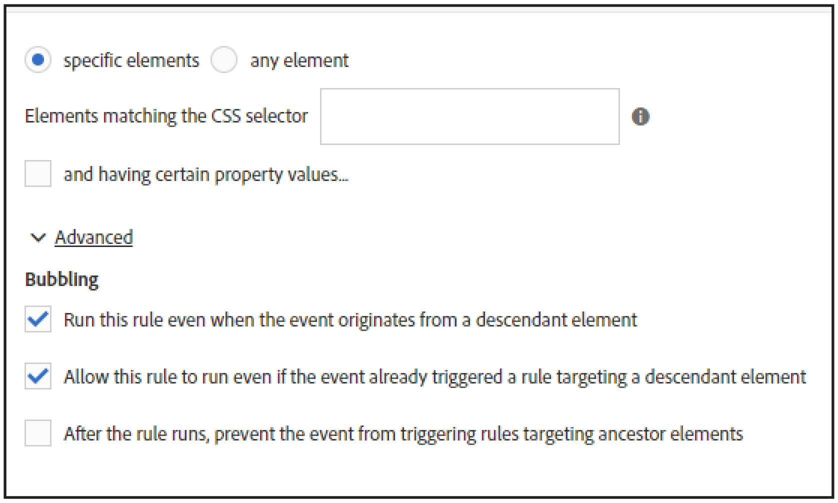
The event types available to build rules using the Core extension are grouped under categories such as browser, form, keyboard, media, mobile, mouse, other, and page load. Each category includes event types, which you will use based on your requirements.

Browser

Tab Blur and Tab Focus are the two event types available under Browser. These event types trigger an action when a tab loses or gains the focus. You do not have to configure anything additional.

Form

The event types available under Forms trigger an action when a form loses or gains focus, or when a form is submitted. When you select Blur, Focus, or Submit under Form, you need to configure the elements matching the CSS selector. CSS selectors help target specific elements on a Web page. In addition, under Advanced settings, you can choose to run a rule even when the event originates from a descendant element, allow a rule to run even if the event already triggered a rule targeting a descendant element, and/or prevent the event from triggering rules targeting ancestor elements after the rule runs, as shown.



● specific elements ○ any element

Elements matching the CSS selector

□ and having certain property values...

Bubbling

☑ Run this rule even when the event originates from a descendant element

☑ Allow this rule to run even if the event already triggered a rule targeting a descendant element

□ After the rule runs, prevent the event from triggering rules targeting ancestor elements

Keyboard

The Key Press event type triggers an action if a key is pressed. You can either select specific elements matching the CSS selector or select the **any element** option. If you select **any element**, you do not need to configure anything further. Additionally, you can select the required Bubbling option(s) under Advanced settings to run a rule even when the event originates from a descendant element, allow a rule to run even if the event already triggered a rule targeting a descendant element, and/or prevent the event from triggering rules targeting ancestor elements after the rule runs.

Media

The event types under Media trigger an action when the media ends, loads data, stalls, is paused, is played, or when the volume is increased or decreased. Specify the elements to target specific elements on a Web page. In addition, for Media Time Played event type, indicate the time passed in seconds or percentage (%).

Mobile

The event types under Mobile trigger an action if the device's orientation changes or if the user zooms in or zooms out the media. You do not need to configure anything additional.

Mouse

The event types under Mouse trigger an action when a user clicks an element, hovers the cursor over a specific element, or scrolls down a page. Under settings, you can specify:

- The elements and 'Bubbling' options
- Whether to delay navigation until the rule runs, if element is a link
- Whether the rule is triggered immediately or after a specified number of milliseconds

Other

The following event types are available under the Other category:

- Custom Code: Triggers an action if a custom code is executed. Open the editor and include your custom code.
- Custom Event: Triggers an action if a custom event type occurs. Specify the name of the custom event type, elements, and Bubbling options.
- Data Element Changed: Triggers an action if a specified data element changes. Specify the data element name from the list by clicking the database icon.
- Direct Call: This event type is ideal when Launch cannot detect an event in the DOM, such as with Adobe Flash. Specify the string that will be passed to _satellite.track() in your direct call (without quotes).
- Element Exists: Triggers an action if a specified element exists. Specify the elements matching the CSS selector.

- Enters Viewport: Triggers an action if the user enters a specified viewport. Specify elements and Bubbling options. In addition, configure whether the rule is triggered immediately or after a specified number of milliseconds.
- Time on Page: Triggers an action if the user remains on the page for a specified number of seconds. Specify the number of seconds that must pass before the event is triggered.

Page Load

The event types under Page Load triggers an action when a page loads, when a user reaches the bottom of the page, or when a window or a library is loaded. No additional configuration is required.

References

Refer these links for:

Event, Condition, Exception, and Action types available for Analytics extension

Condition, Exception, Action types available for Core extension

Configure Launch Rules and Data Elements

Introduction

Data elements are the building blocks of data dictionary in Launch. You will use and reuse data elements extensively to achieve your varied goals. In addition, you will use rules to integrate data, so that you can measure and react to various interactions in real-time.

Objectives

After completing this chapter, you will be able to:

- Apply embed codes in your site
- Describe data elements
- Create data elements
- Describe rules
- Describe events, conditions, exceptions, and action types
- Implement a rule

Data Elements

Launch uses data elements to collect, organize, and deliver data across technologies. They are pointers to the information you want to collect and send to different places on your page. This could be:

- A defined data layer in JSON
- DOM elements
- Cookies
- Session and local storage

Data element values are mapped to URLs, cookie values, and JavaScript variables. You can reference this value with its variable name throughout Launch. This collection of data elements becomes the dictionary of defined data that you can use to build your events, conditions, and actions. This data dictionary is shared across Launch for reuse.

Data elements improve the efficiency of the tagging process. You can define data rules once and reuse them in multiple places. For example, if you use a page name schema by referencing the document.title element, you can create a 'Page Name' data element in Launch that can be used in any rule that needs to reference the page name. If you decide to change the way you refer the page name, you just need to change the reference once in the data element you created, and all rules that reference that data element are automatically updated.

Data elements: Usage

- Rules: You can use data elements in the rule editing interface by using the search box to find the name of your data element.
- Custom script: You can use data elements in custom scripts by using the _satellite object syntax: _satellite.getVar('data element name');

The following table describes, with example, some of the data elements available in the Core extension, which you will use in subsequent exercises:

Type	Description	Example

JavaScript Variable	Helps reference any available JavaScript object or variable, using the path field.
	Note: To use a data collector object as part of your data layer, use dot notation in the path to reference the object and property you want to capture into the Data Element. For example, _myData.pageName, or digitalData.pageName, etc.
		`<script>`
		`//data layer`
		`var Page_Name = “Homepage”`
		`</script>`

DOM Attribute	Helps retrieve any element value, such as a div or H1 tag	CSS Selector Chain:
		`id#dc logo img`
		Get the value of:
		`src`

Cookie	Helps reference any available domain cookie in the cookie name field.	`cookieName`
URL Querystring Parameter	Helps specify a single URL parameter in the URL Parameter field. Specify only the name section. Omit designators like “?” or “=”	`contentType`

Custom Script	A return statement is necessary to indicate what value should be set as the data element value. If a return statement is not included, the default value or an empty string will be returned as the data element value.	`var pageType =`
		`$(‘div.page-`
		`wrapper’).`
		`attr(‘class’).`
		`split(‘`
		`‘)[1];`
		
		`if`
		`(window.location.`
		`pathname ==`
		`‘/’){return`
		`‘homepage’;`
		`}`
		
		`else`
		`{`
		`return pageType;`
		`}`

Note: The element types are available based on the extensions you have installed. See Additional Information section of this document for a list of element types available under different extensions.

Note: Nothing is automatically published in Launch. After creating or editing data elements, you need to publish the changes for them to take effect.

Rules

Launch is a rule-based system. If a specified event takes place, a rule is triggered and specified actions take place. Launch looks for user interaction and associated data, and when the defined criteria are met, extensions, script, or HTML are triggered. You can build rules to integrate disparate products into a single solution.

A rule consists of two parts:

- Events, along with any conditions and exceptions, form the 'If' part of a rule. An event is what you want the rule to look for. You can specify one or more events that must take place to trigger the rule. Multiple events are joined by an OR operation. Any of the specified events will trigger the rule.
  - › Events: Specify one or more events that must take place to trigger the rule.
  - › Conditions: Narrow the event by configuring any conditions that must be true for an event to trigger the rule.
  - › Exceptions: Specify any exceptions that would keep the rule from firing, even if the events and conditions are met.
- Actions: Are the 'Then' part of the rule. An action occurs after an event takes place and all conditions and exclusions are satisfied. A rule in Launch can trigger as many discrete actions as you want, and you can control the order in which these actions occur.

Rule Ordering

It is often important to have your rules fire in a specific order. In Launch, you can control the order of execution for rules that share an event. Adobe recommends that you order your rules with positive numbers between 1 and 100 (The default order for all rule components is 50).

If you want to run a rule sooner, give it a number lower than 50. Rules that have the same order run in no particular order. Even though rules are fired in order, they do not necessarily finish in the same order. If Rule A and Rule B share an event, and you assign an order so that Rule A comes first, then if Rule A does something asynchronously, there's no guarantee that Rule A finishes before Rule B starts.

The load order for rules depend on whether the rule action is configured with JavaScript or HTML, and whether the rules use a page bottom event, page top event, or a different type of event. Additionally, you can order different custom code types amongst each other. For example, you can have a JavaScript custom code action, then an HTML custom code action, then a JavaScript custom code action. Adobe Launch ensures that they are executed in the same order.

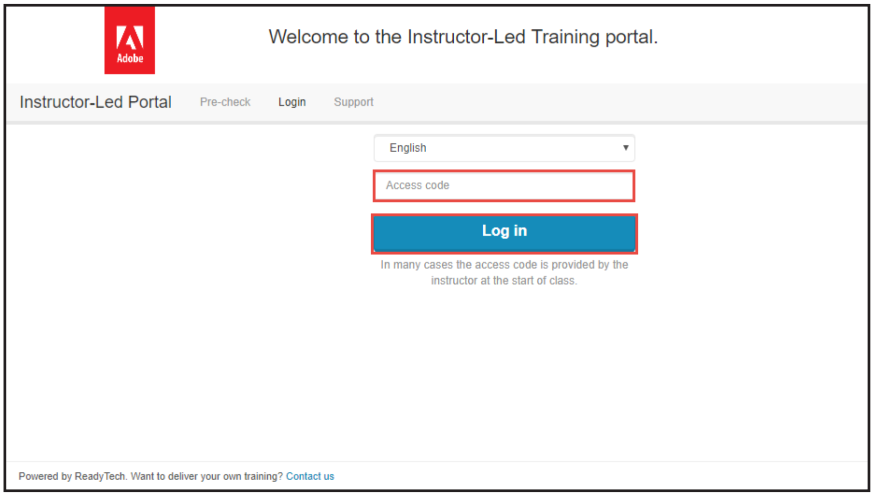
Rules with page bottom or page top event	Rules with any other event
The JavaScript is embedded in the main Launch library. The custom script is wrapped in a script tag and written to the document using document. write. If the rule has multiple custom scripts, they are written in order.	The JavaScript is loaded from the server as regular text, wrapped in a script tag, and added to the document using Postscribe. If the rule has multiple JavaScript custom scripts, they are loaded in parallel, but executed in the same order that was configured in the rule.
The HTML is embedded in the main Launch library. document.write is used to write the HTML to the document. If the rule has multiple custom scripts, they are written in order.	The HTML is loaded from the server and added to the document using Postscribe. If the rule has multiple custom HTML scripts, they will be loaded in parallel, but executed in the same order that was configured in the rule.

For a list of event, condition, exception, and action types, available for Analytics extension, refer to this link:

Task 1: Log on to Launch

In this training, you will use a ReadyTech training environment to access Launch. To log in to the ReadyTech environment:

1. Open a browser, and navigate to https://adobe.instructorled.training to access the ReadyTech training site.
2. Enter the **Access Code** provided to you and click **Log In**, as shown**.** The Instructor-Led Portal screen opens.

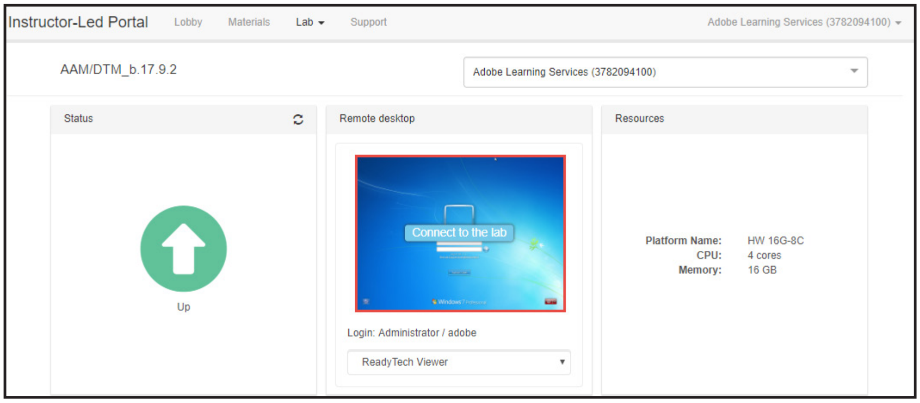


Welcome to the Instructor-Led Training portal.

In many cases the access code is provided by the instructor at the start of class.

Powered by ReadyTech. Want to deliver your own training? Contact us

3. Click **Connect to the lab**, as shown:



AAM/DTM_b.17.9.2

Adobe Learning Services (3782094100)

Login: Administrator / adobe

Platform Name: HW 16G-8C
CPU: 4 cores
Memory: 16 GB

4. Enter the login credentials provided to you by your instructor. The ReadyTech desktop is displayed.
5. Open Chrome in ReadyTech and navigate to http://launch.adobe.com
6. Enter the Launch login credentials provided to you by your instructor, and click **Sign In**.

Note: If you have created properties before, you will see a list of properties on the Properties page. If you are logging on for the first time, your first task is to create a property.

Task 2: Install the Adobe Experience Cloud Debugger Chrome Extension

The Adobe Experience Cloud Debugger is a Chrome extension that contains basic debugging tools for all SaaS-based Experience Cloud Solutions. To install the Adobe Experience Cloud Debugger extension:

1. Open the below link in your ReadyTech browser:
   https://chrome.google.com/webstore/detail/adobe-experience-cloud-de/ocdmogmohccmeicdhlhhgepeaijenapj
2. Click **ADD TO CHROME**.
3. Click **Add extension** in the confirmation window. The extension is added to your browser. You can access it from the Bookmarks bar on the right of your screen. You will use Experience Cloud Debugger in later exercises.

Task 3: Install the Launch Command Chrome Extension

The Launch Command Chrome extension is used to switch embed codes implemented on a page. You can use it to make the page load a:

- Development library instead of the Production library
- Launch library instead of a DTM library
- library from a different account

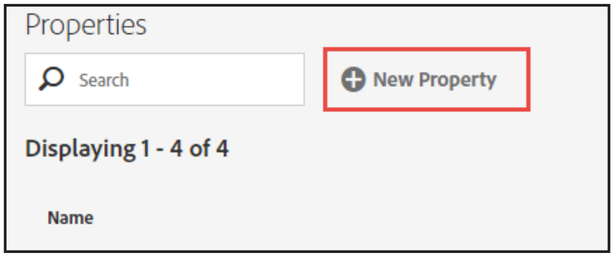
To install the Launch command Chrome extension:

1. Open the below link in your ReadyTech browser:
https://chrome.google.com/webstore/detail/launch-command/nkjhamgjeocefocmpbcjfmohkjgildki
2. Click **ADD TO CHROME**.
3. Click **Add extension** in the confirmation window. The extension is added to your browser. You can access it from the Bookmarks bar. In a later task, you will add embed codes to the Launch Command extension.

Note: When the extension is enabled, it will replace all DTM and Launch embed codes on every page of every tab you load. This is different from DTM Switch, which was localized to the domain you were on.

Task 4: Create a property

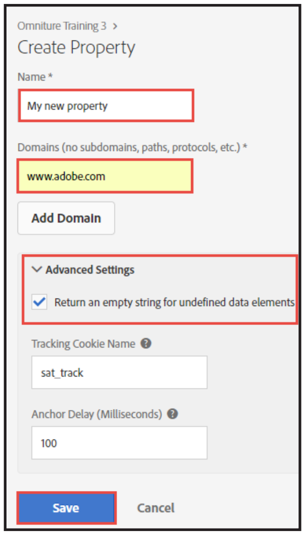
1. Go back to the Launch UI.
2. On the **Properties** page, click **New Property** to create a new property. The **Create Property** dialog box opens, as shown:



Displaying 1 - 4 of 4

3. Update the property fields:
   a. In the **Name** field, enter **My new property**.
   b. In the **Domains** field, enter **www.adobe.com**.
   c. Under **Advanced Settings**, select the **Return an empty string for undefined data elements** if you want undefined values to be empty, rather than to assign default values.

d. Optionally, you can specify a custom cookie name in the **Tracking Cookie Name** field, and delay time in the **Anchor Delay (milliseconds)** field. For this exercise, retain the default values.



Omniture Training 3 >

Create Property

Name *

My new property

Domains (no subdomains, paths, protocols, etc.) *

www.adobe.com

Advanced Settings

☑ Return an empty string for undefined data elements

Tracking Cookie Name

sat_track

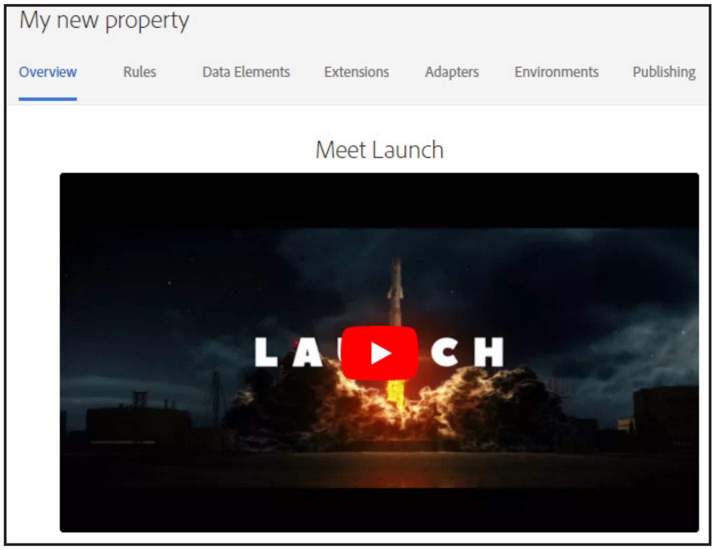
Anchor Delay (Milliseconds)

100

4. Click **Save**. Notice that the new property you created is listed on the **Properties** page.

NOTE: Launch uses the Tracking Cookie Name to track your opt-out status for receiving other cookies. Anchor Delay is the delay time for tracking the link. Longer delays improve tracking accuracy. Any value under 500ms is recommended.

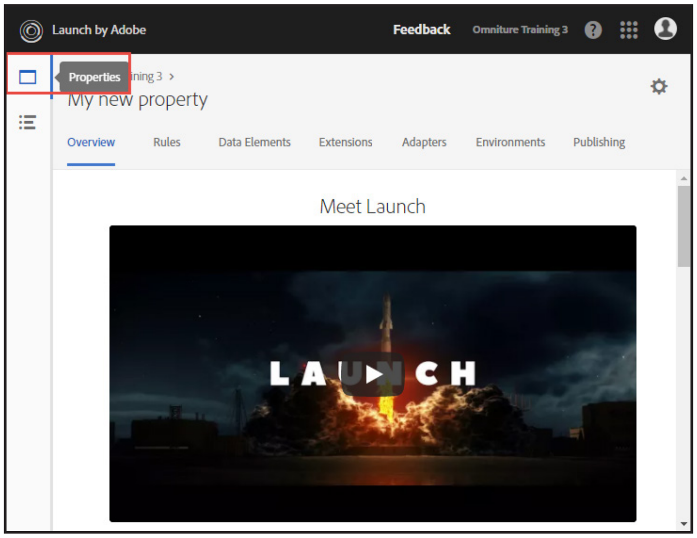
5. Click the property you created. The **My new property** page opens. The **Overview** tab lists all your recent activities. If you have not created anything so far, your **Overview** tab looks similar to the screen shot shown:



My new property

Meet Launch

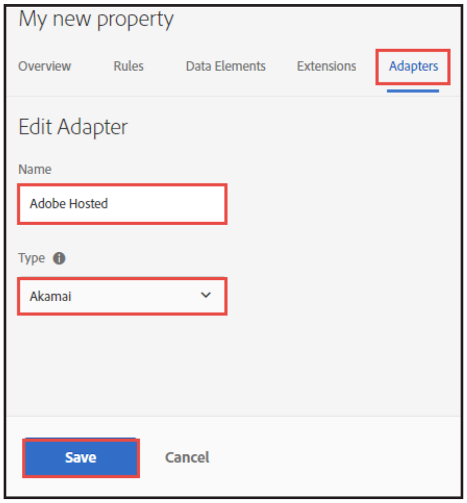
6. Click the **Properties** icon, as shown, to go back to the **Properties** page. Notice the newly created property in the **Properties** page.



Task 5: Create adapters

After creating a property, the next step is to set up adapters for hosting environments. An adapter is a connection between Launch and the hosting location. When you create a build, Launch delivers that build to a location determined by the adapter assigned to the environment. You can choose to have Launch manage that location for you (using Akamai) or manage it yourself (Using SFTP). In this training, you will use Akamai.

1. From the **Properties** page, click the property you created in the previous task to open it. The **My new property** page opens.
2. On the **Adapters** tab, click **Create a New Adapter**. The **Edit Adapter** page opens.
3. Enter a name for the adapter in the **Name** field. For example, enter Adobe Hosted.
4. From the **Type** drop-down list, select **Akamai**.



Name: Adobe Hosted

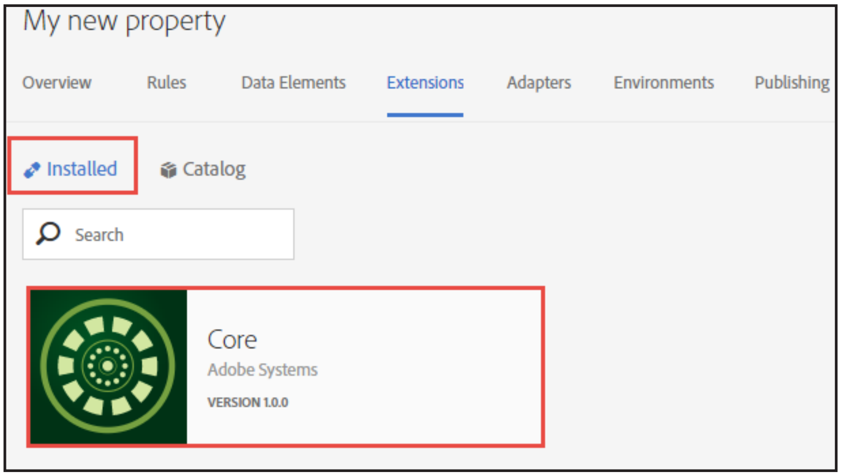
Type: Akamai

5. Click **Save** > **Close**.

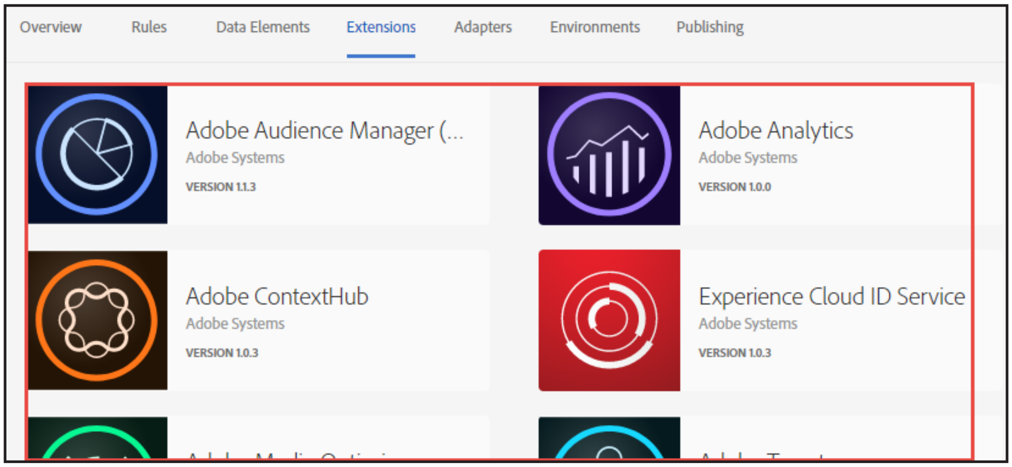
NOTE: You can add multiple adapters by clicking the **Add Adapter** button. Additionally, you can create multiple FTP adapters for staging and production environments. In this training, you will be using Akamai.

Task 6: Add Analytics extension

1. Click the **Extensions** tab. Notice that in the **Installed** tab, only the default Core extension is available.

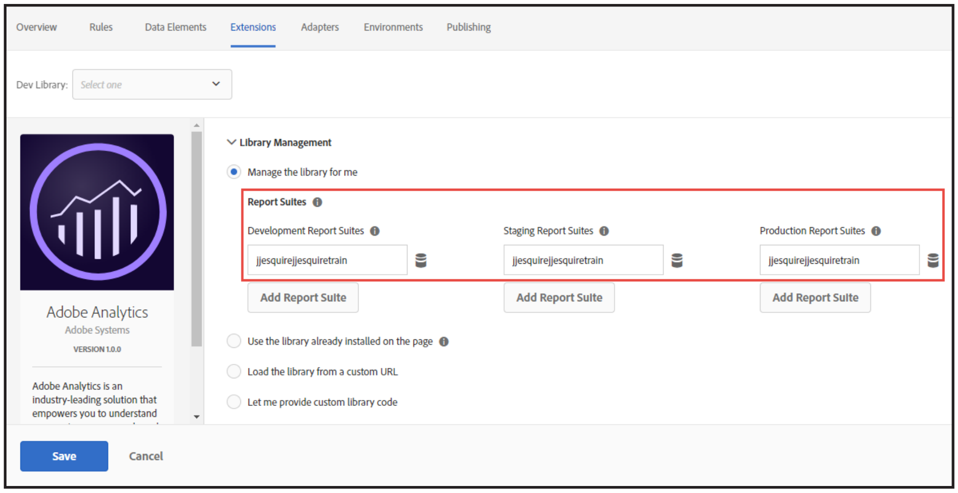


2. Click the **Catalog** tab to access the available extensions. Notice that all available extensions are listed here including the Analytics extension.



3. To install the Adobe Analytics extension, hover the cursor over the extension and click **Install**. The Adobe Analytics extension configuration page opens.

4. Under **Library Management**, specify the report suite to which your data should be sent. You can type the report suite ID in the field or click the database icon next to each report suite field to select a data element. You can add multiple report suites by clicking **Add Report Suite**. For this task, type **jjesquirejjesquiretrain** as report suite under Development, Staging, and Production Report suites, as shown.



Dev Library: Select one

Library Management

● Manage the library for me

Report Suites

Development Report Suites: jjesquirejjesquiretrain

Staging Report Suites: jjesquirejjesquiretrain

Production Report Suites: jjesquirejjesquiretrain

○ Use the library already installed on the page

○ Load the library from a custom URL

○ Let me provide custom library code

Adobe Analytics is an industry-leading solution that empowers you to understand

5. Click **Save**.

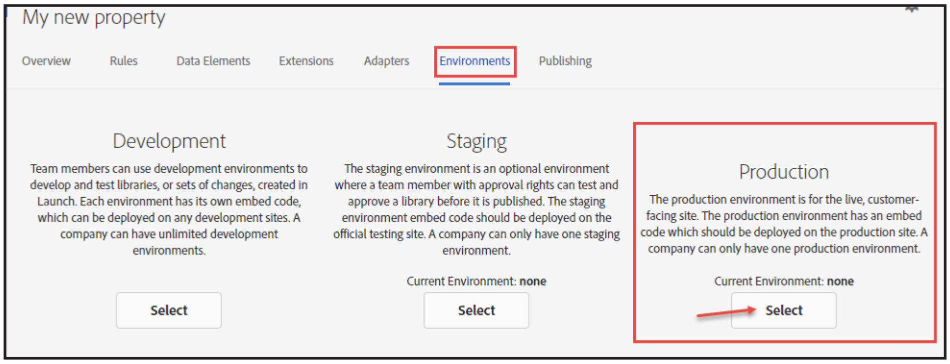
Exercise 1: Retrieve embed code and apply to the site

Task 1: Create environment

An environment is a set of deployment instructions for build format and location. An environment has a one-to-one relationship with an embed code. The embed code:

- Consists of two <script> tags that you need to include within the HTML of your website
- Is automatically generated for based on the environment configuration. You need to copy and paste the codes in web pages where you want to run Launch.
- Is used to call the JavaScript library code

1. On the **Environments** tab, click **Create New Environment**. You will see the Development, Staging, and Production environments section.
2. To set up a production environment, click **Select** under **Production**, as shown:



Development

Team members can use development environments to develop and test libraries, or sets of changes, created in Launch. Each environment has its own embed code, which can be deployed on any development sites. A company can have unlimited development environments.

Staging

The staging environment is an optional environment where a team member with approval rights can test and approve a library before it is published. The staging environment embed code should be deployed on the official testing site. A company can only have one staging environment.

Current Environment: **none**

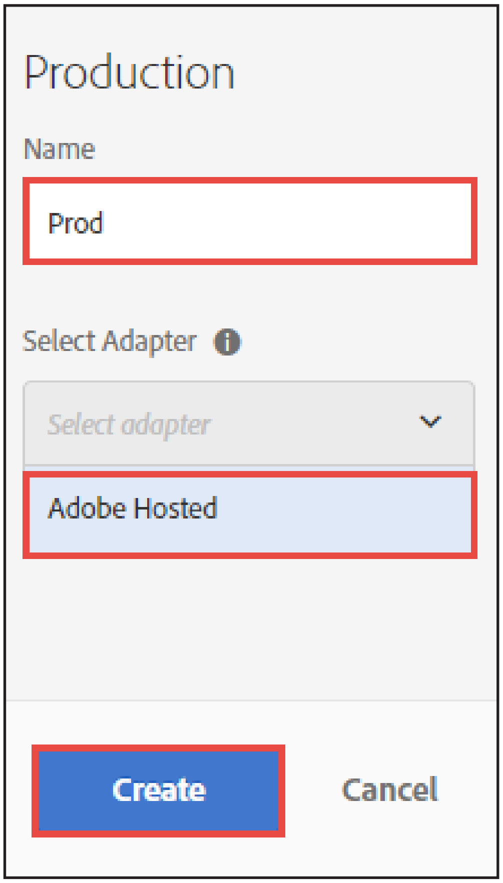
Production

The production environment is for the live, customer-facing site. The production environment has an embed code which should be deployed on the production site. A company can only have one production environment.

Current Environment: **none**

3. In the **Name** field, enter an appropriate name. For example, enter Prod.
4. From the **Select Adapter** drop-down list, select the adapter you created in the previous task. For example, select Adobe Hosted, as shown.

5. Click **Create,** as shown:

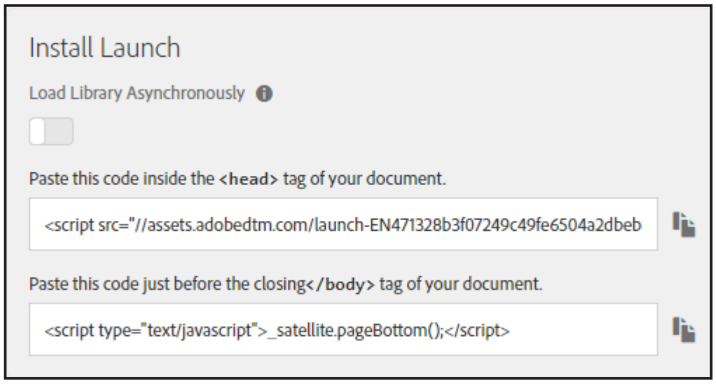


Production

Name: Prod

Select Adapter: Adobe Hosted

Notice the header and footer embed codes at the bottom of the page, as shown in the screenshot. These codes determine the loading of JavaScript and page content on your site.



Install Launch

Load Library Asynchronously

Paste this code inside the <head> tag of your document.

```
<script src="//assets.adobedtm.com/launch-EN471328b3f07249c49fe6504a2dbeb
```

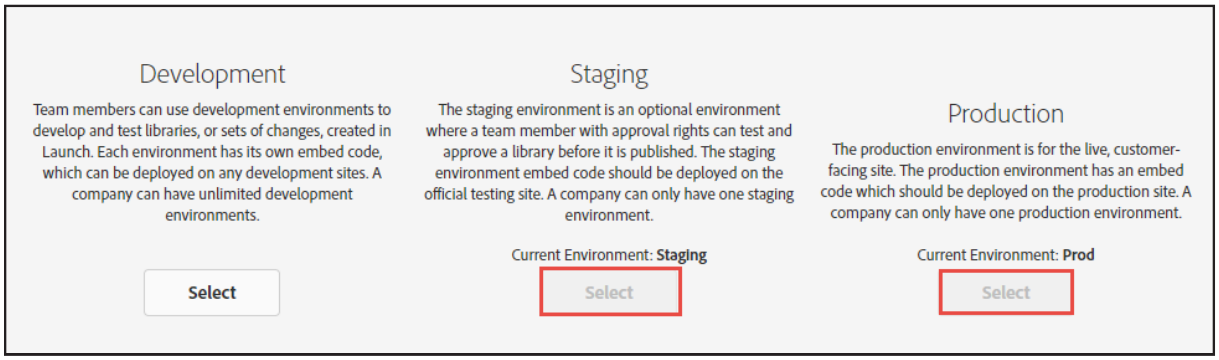
Paste this code just before the closing</body> tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

6. Click **Close**.
7. To setup a staging environment, click **Add Environment** and click **Select** under **Staging.**
8. In the **Name** field, enter an appropriate name. For example, enter **Staging**.
9. From the **Select adapter** drop-down list, select the adapter you created in the previous task. For example, enter Adobe Hosted.
10. Click **Create.** The embed codes are displayed at the bottom of the page.

11. Click **Close**.
12. Click **Add Environment** to set up development environments.
13. Click **Select** under **Development.**
14. In the **Name** field, enter an appropriate name. For example, enter Dev.
15. From the **Select adapter** drop-down, select the adapter you created in the previous task. For example, enter Adobe Hosted.
16. Click **Create**.
17. Click **Close**.

Notice that the **Select** buttons under **Staging** and **Production** are grayed out, as shown. This is because, you can create only one staging environment and one production environment. However, you can create multiple development environments.



Development

Team members can use development environments to develop and test libraries, or sets of changes, created in Launch. Each environment has its own embed code, which can be deployed on any development sites. A company can have unlimited development environments.

Staging

The staging environment is an optional environment where a team member with approval rights can test and approve a library before it is published. The staging environment embed code should be deployed on the official testing site. A company can only have one staging environment.

Current Environment: **Staging**

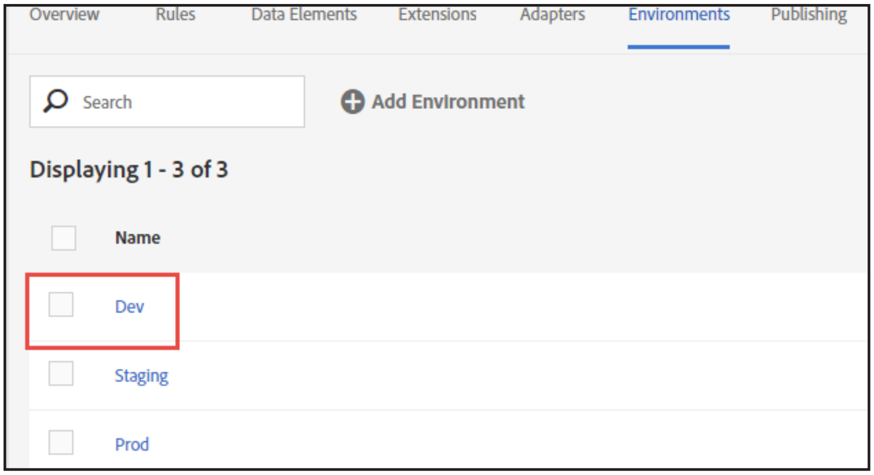
Production

The production environment is for the live, customer-facing site. The production environment has an embed code which should be deployed on the production site. A company can only have one production environment.

Current Environment: **Prod**

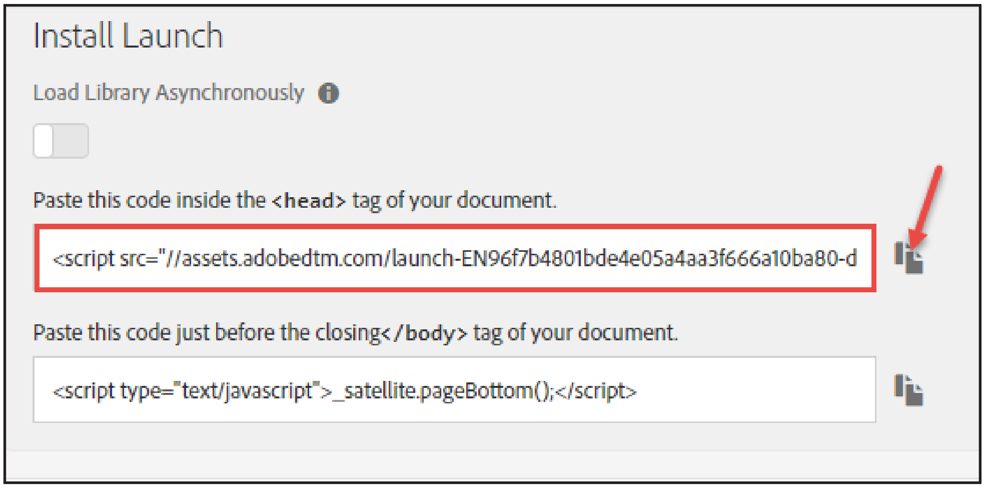
Task 2: Retrieve embed code from the development environment and apply to the practice site

1. From the **Environments** tab, click the development environment you created earlier, as shown. The Development environment section opens.



2. Scroll down to the end of the page to access embed codes.

3. Click the **Copy** icon next to the <head> tag, as shown, to copy the code to the clipboard. The code is copied.



Paste this code inside the **<head>** tag of your document.

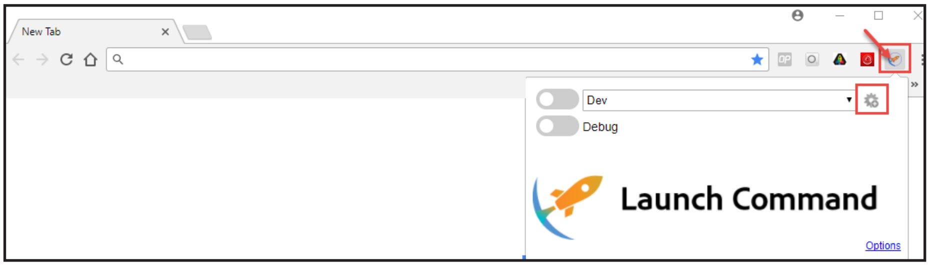
```
<script src="//assets.adobedtm.com/launch-EN96f7b4801bde4e05a4aa3f666a10ba80-d
```

Paste this code just before the closing**</body>** tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

4. In a new window, in your ReadyTech Google Chrome browser, open the **Launch Command** extension, as shown in the screenshot.

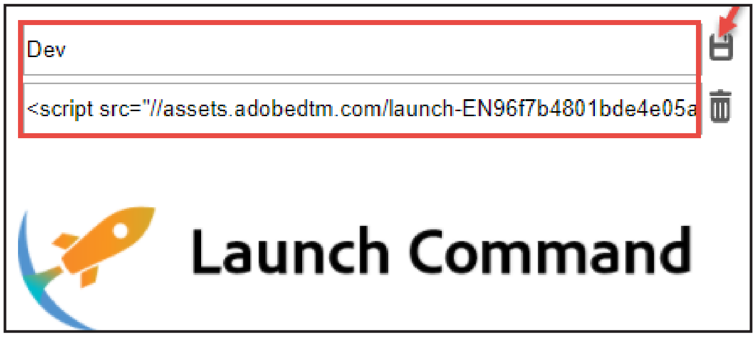
5. Click the gear icon on the right, as shown. You will see fields to update name and embed code.



6. In the **Name** field, enter a suitable name.

7. In the **Embed Code** field, paste the embed code you copied from Launch earlier.

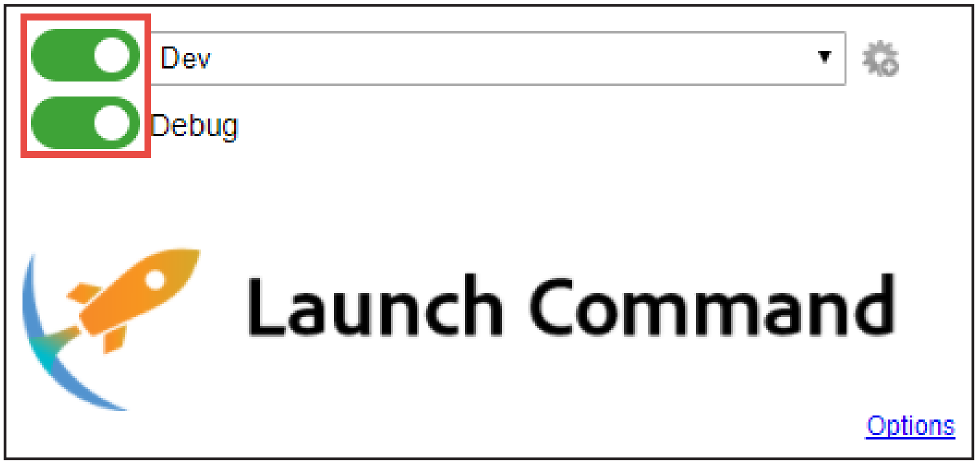
8. Click the **Save** icon on the right to save the embed code.



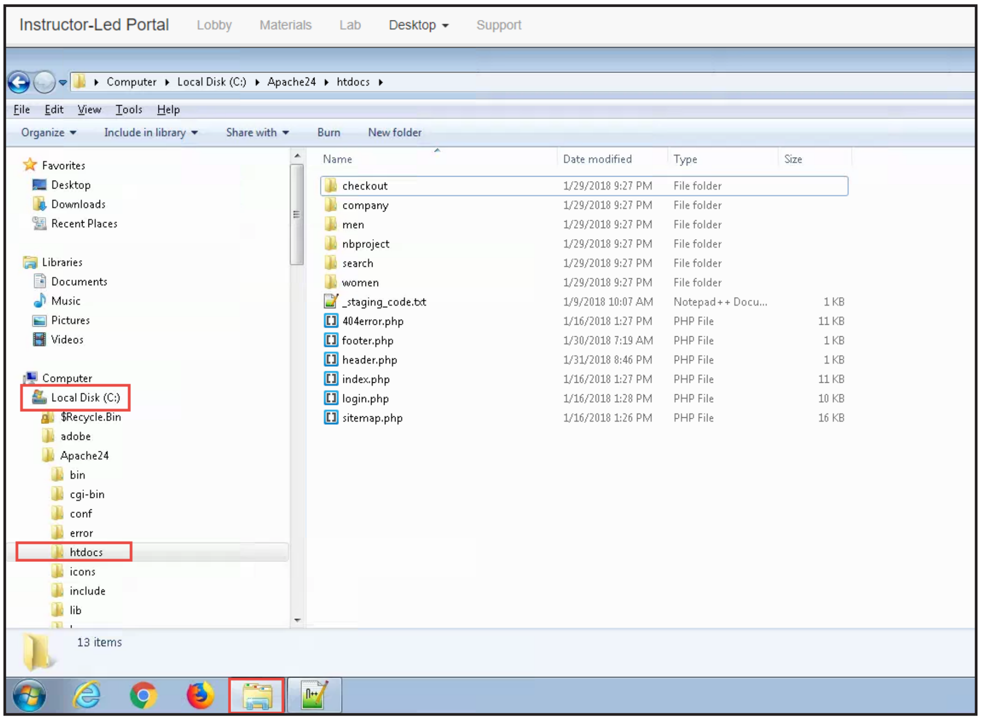
Dev

<script src="//assets.adobedtm.com/launch-EN96f7b4801bde4e05a

9. Toggle on the extension. You will see how the Launch Command extension works later in the exercise.

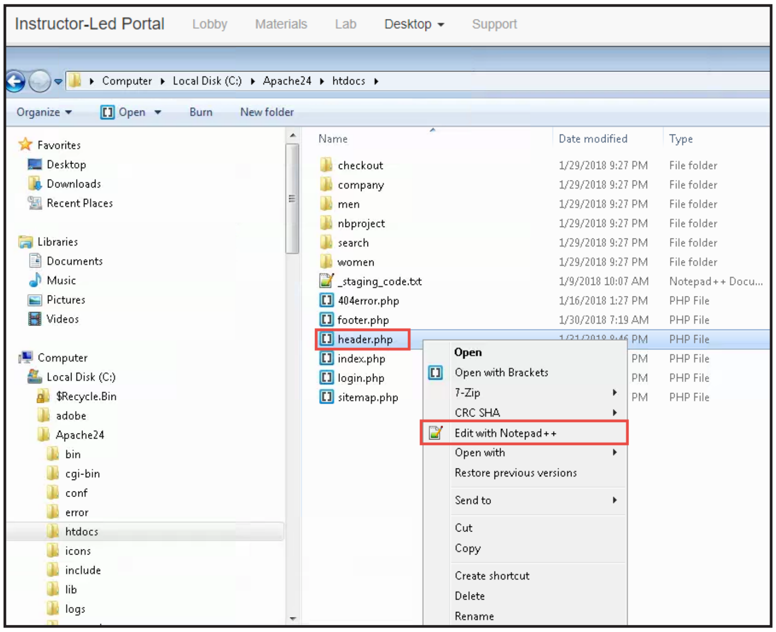


10. From your ReadyTech desktop, open the Apache24 > htdocs folder from the local C drive, as shown:



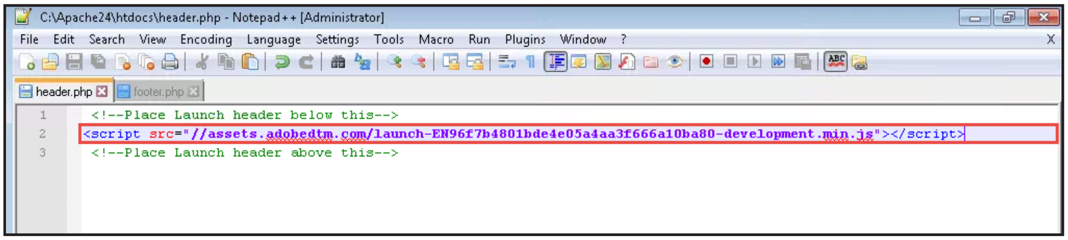
Name	Date modified	Type	Size
checkout	1/29/2018 9:27 PM	File folder	
company	1/29/2018 9:27 PM	File folder	
men	1/29/2018 9:27 PM	File folder	
nbproject	1/29/2018 9:27 PM	File folder	
search	1/29/2018 9:27 PM	File folder	
women	1/29/2018 9:27 PM	File folder	
_staging_code.txt	1/9/2018 10:07 AM	Notepad++ Docu...	1 KB
404error.php	1/16/2018 1:27 PM	PHP File	11 KB
footer.php	1/30/2018 7:19 AM	PHP File	1 KB
header.php	1/31/2018 8:46 PM	PHP File	1 KB
index.php	1/16/2018 1:27 PM	PHP File	11 KB
login.php	1/16/2018 1:28 PM	PHP File	10 KB
sitemap.php	1/16/2018 1:26 PM	PHP File	16 KB

11. Right-click header.php and click **Edit with Notepad ++,** as shown, to open it in Notepad.



Name	Date modified	Type
checkout	1/29/2018 9:27 PM	File folder
company	1/29/2018 9:27 PM	File folder
men	1/29/2018 9:27 PM	File folder
nbproject	1/29/2018 9:27 PM	File folder
search	1/29/2018 9:27 PM	File folder
women	1/29/2018 9:27 PM	File folder
_staging_code.txt	1/9/2018 10:07 AM	Notepad++ Docu...
404error.php	1/16/2018 1:27 PM	PHP File
footer.php	1/30/2018 7:19 AM	PHP File
header.php	1/31/2018 8:46 PM	PHP File
index.php	PM	PHP File
login.php	PM	PHP File
sitemap.php	PM	PHP File

12. Paste the head code you copied in step 3, as shown:

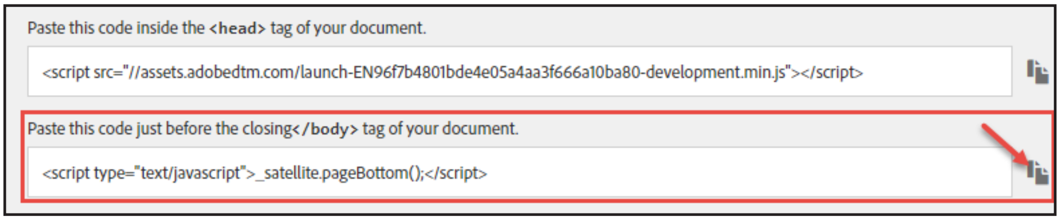

C:\Apache24\htdocs\header.php - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
header.php
footer.php

```
<!--Place Launch header below this-->
<script src="//assets.adobedtm.com/launch-EN96f7b4801bde4e05a4aa3f666a10ba80-development.min.js"></script>
<!--Place Launch header above this-->
```

13. Click **File** > **Save**.

14. Repeat steps 1 and 2.

15. Click the **Copy** icon next to the footer tag, as shown, to copy the code to the clipboard.



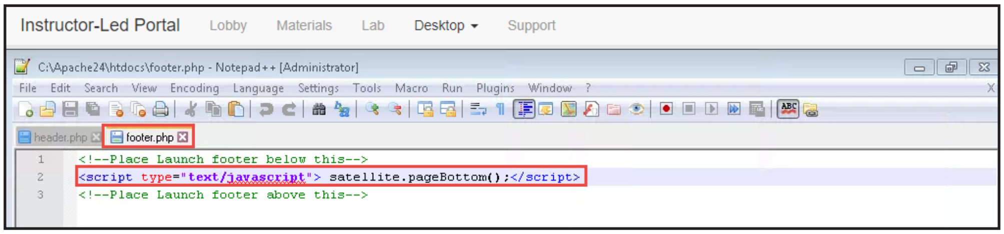
Paste this code inside the **<head>** tag of your document.

```
<script src="//assets.adobedtm.com/launch-EN96f7b4801bde4e05a4aa3f666a10ba80-development.min.js"></script>
```

Paste this code just before the closing**</body>** tag of your document.

```
<script type="text/javascript">_satellite.pageBottom();</script>
```

16. From your ReadyTech local C drive, open Apache24 > htdocs folder, right-click footer.php, open it using Notepadd++, and paste the code you copied in the previous step:

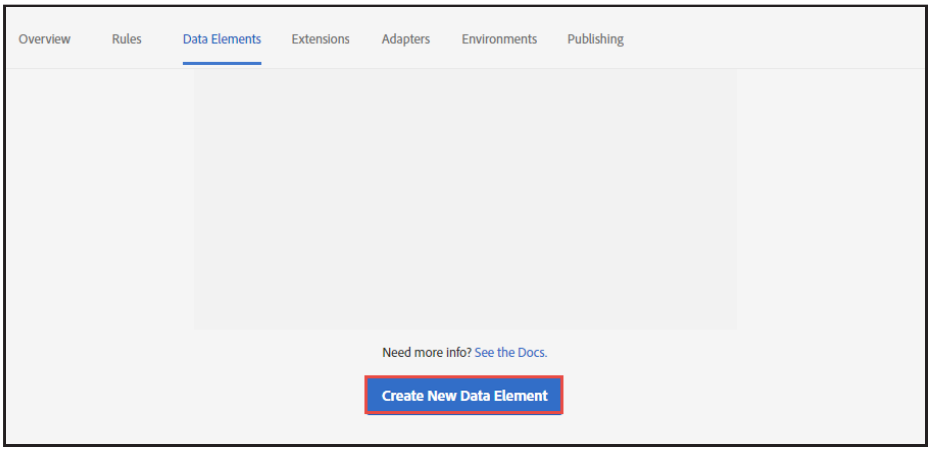

Instructor-Led Portal
Lobby
Materials
Lab
Desktop
Support
C:\Apache24\htdocs\footer.php - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
header.php
footer.php

```
<!--Place Launch footer below this-->
<script type="text/javascript"> satellite.pageBottom();</script>
<!--Place Launch footer above this-->
```

17. Click **File** > **Save**.

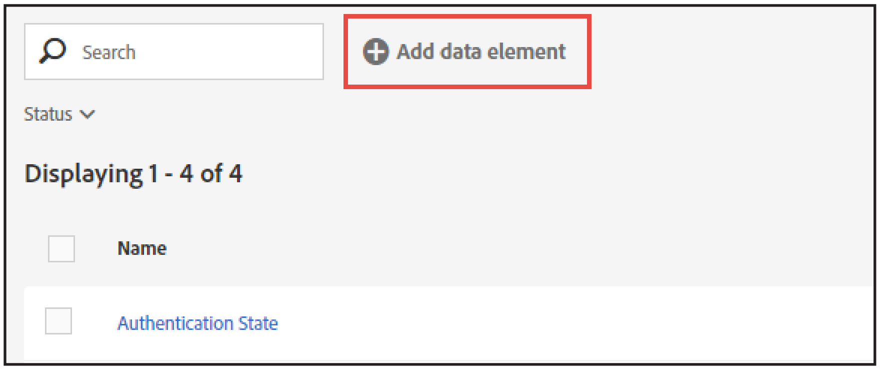
Task 1: Create a page name data element

1. On the **Data Elements** tab, click **Create New Data Element**, as shown, if you are creating your first data element. The Create New Data Element page opens.



Need more info? See the Docs.

2. If you have created data elements before, click **Add data element** at the top to add another data element. The **Create New Data Element** section opens.



Displaying 1 - 4 of 4

3. Update the fields in the **Create New Data Element** section, as shown:
   a. In the **Name** field, enter **Page Name**.
   b. Retain **Core** as the extension.

c. From the **Data Element Type** drop-down list, select **JavaScript Variable**.

d. In the **Path to variable** field on the right, enter **digitalData.page.pageName**

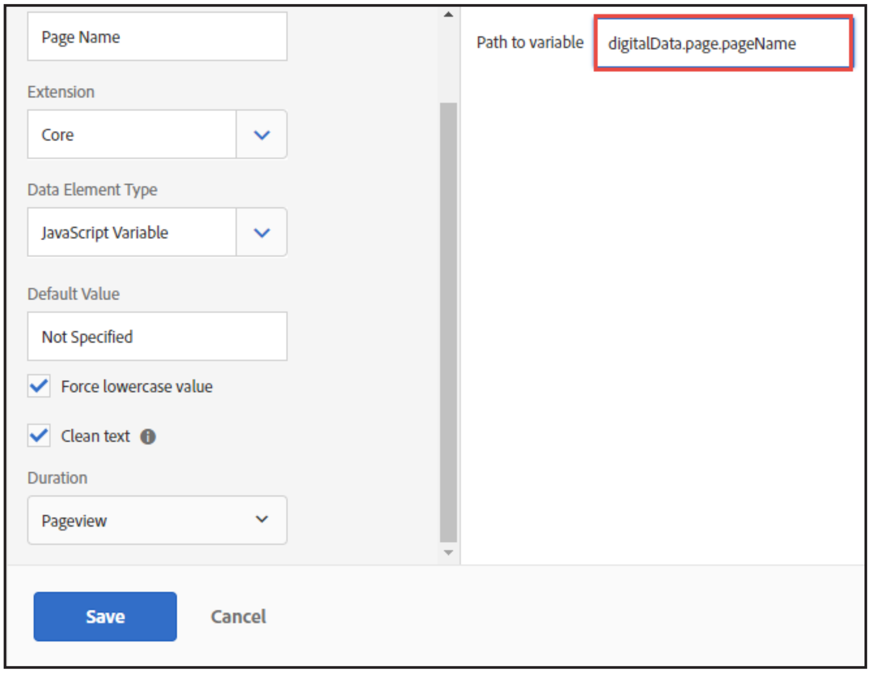
e. (Optional) Enter a value in the **Default Value** field.

f. Select the **Force lowercase value** checkbox.

g. Select the **Clean text** checkbox.

h. Retain **Pageview** in the **Duration** field.

i. Click **Save**.



Page Name

Extension: Core

Data Element Type: JavaScript Variable

Default Value: Not Specified

☑ Force lowercase value

☑ Clean text

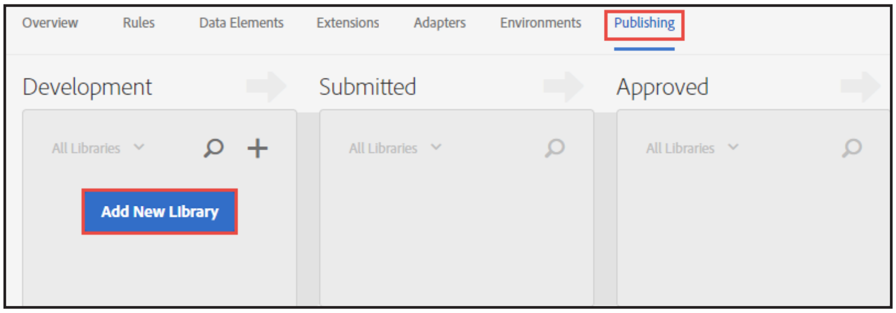
Duration: Pageview

Path to variable: digitalData.page.pageName

The Page Name data element is created. Now, verify if the data element is correctly set.

4. Open the **Publishing** tab.

5. Click **Add New Library** to create a library, as shown. The **Create New Library** section opens.

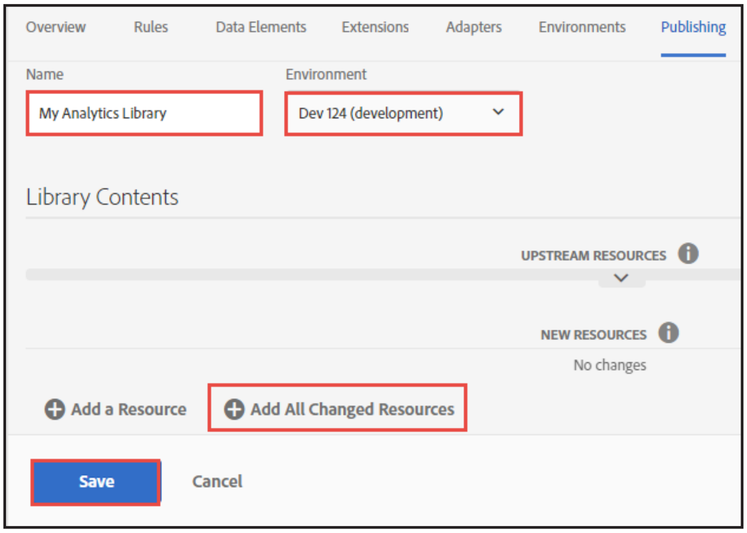


6. In the **Name** field, enter a suitable name, as shown. For example, enter My Analytics Library.

7. From the **Environment** drop-down list, select the the development environment you created earlier, as shown.

8. Click **Add All Changed Resources** to add content to your library, as shown.

9. Click **Save**.



Name: My Analytics Library

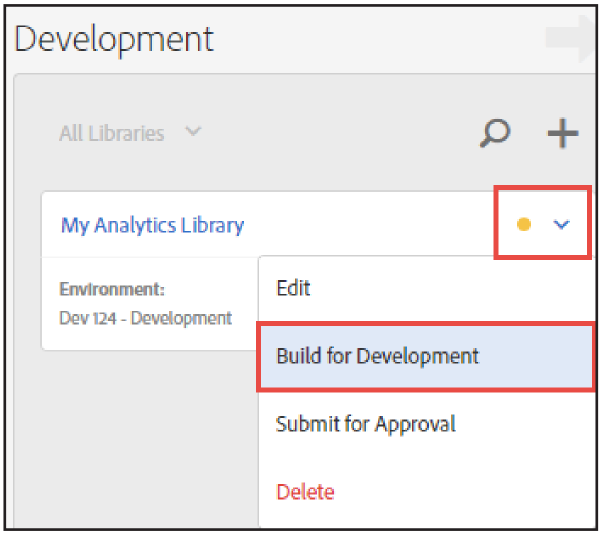
Environment: Dev 124 (development)

Library Contents

No changes

10. Your library is created. Notice the yellow dot next to your library name in the image below. It indicates that there is no build. Your changes will reflect only after you create a build.

11. From the drop-down list, click **Build for Development**, as shown:

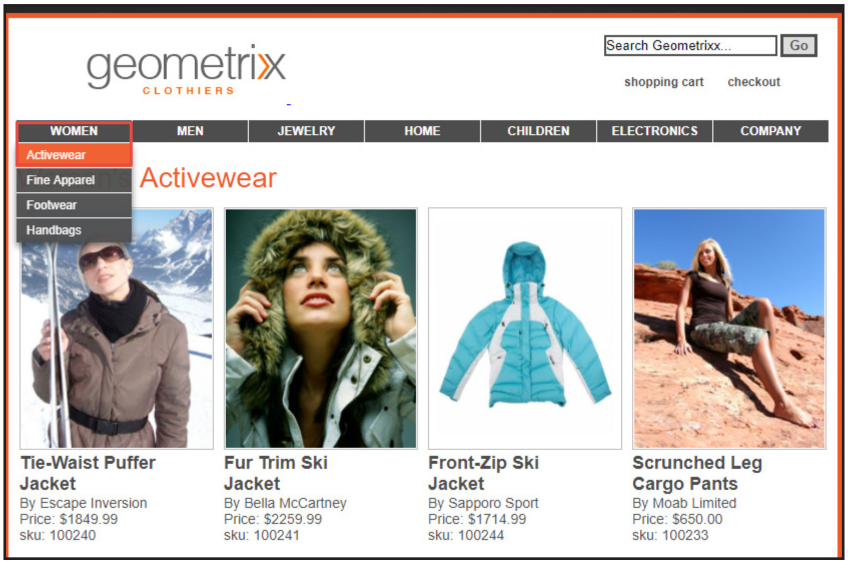


Environment: Dev 124 - Development

Notice that the yellow dot has now turned green. This indicates that the library now has a build.

12. From your ReadyTech browser, navigate to localhost.adobevlab.com. You can also access it from the Bookmark bar on the browser. The demo site opens.

13. Go to **WOMEN** > **Activewear** page, as shown:



Activewear

Tie-Waist Puffer Jacket
By Escape Inversion
Price: $1849.99
sku: 100240

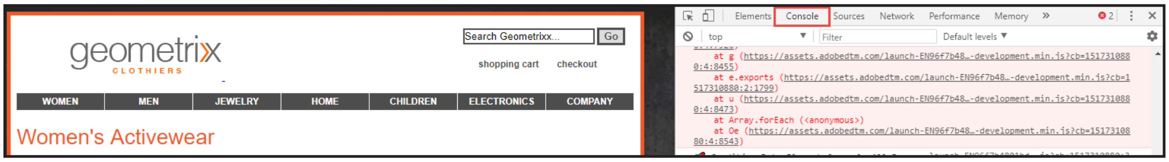
Fur Trim Ski Jacket
By Bella McCartney
Price: $2259.99
sku: 100241

Front-Zip Ski Jacket
By Sapporo Sport
Price: $1714.99
sku: 100244

Scrunched Leg Cargo Pants
By Moab Limited
Price: $650.00
sku: 100233

14. Right-click the page, and click **Inspect**. Alternatively, you can press Ctrl+Shift+I on your keyboard. The **Developer Tools** window opens on the right.

15. Open the **Console** tab, as shown:

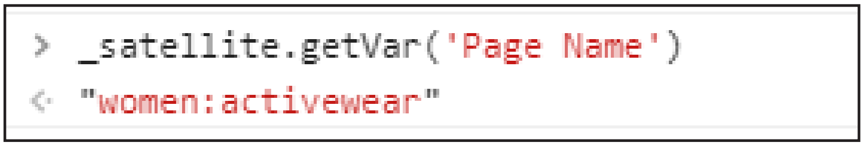


Women's Activewear

```
    at g (https://assets.adobedtm.com/launch-EN96f7b48…-development.min.js?cb=151731088
0:4:8455)
    at e.exports (https://assets.adobedtm.com/launch-EN96f7b48…-development.min.js?cb=1
517310880:2:1799)
    at u (https://assets.adobedtm.com/launch-EN96f7b48…-development.min.js?cb=151731088
0:4:8473)
    at Array.forEach (<anonymous>)
    at Oe (https://assets.adobedtm.com/launch-EN96f7b48…-development.min.js?cb=15173108
80:4:8543)
```

16. On the **Console** tab, type **_satellite.getVar(‘<*your data element*> name’)**. For example, type _satellite.getVar(‘Page Name’), where Page Name is your data element.

17. Press **Enter** on your keyboard. Notice that **Women:activewear** is displayed, as shown, which is the page you are in right now.



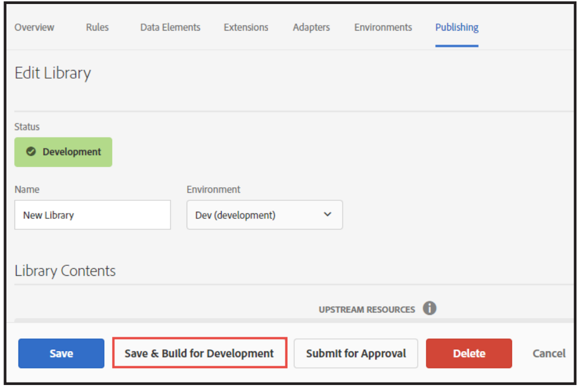
```
> _satellite.getVar('Page Name')
< "women:activewear"
```

18. Go to **MEN** > **Activewear** page.

19. On the Console tab, type **_satellite.getVar(‘<*your data element*> name’)**. For example, type _satellite.getVar(‘Page Name’), where Page Name is your data element. Press **Enter**. You will see “Men:activewear”.

Task 2: Create a product name data element

1. Go back to Launch.
2. Click the **Data Elements** tab.
3. Click **Add data element**. The **Create a New Data Element** section opens.
4. Update the fields in the **Create New Data Element** section:
   a. In the **Name** field, enter **Product Name**.
   b. In the **Extension** field, retain **Core** as the extension.
   c. From the **Data Element Type** drop-down list, select **JavaScript Variable**.
   d. In the **Path to variable** field on the right, enter **digitalData.product.productName**
   e. (Optional) Enter a value in the **Default Value** field.
   f. Select the **Force lowercase value** checkbox.
   g. Select the **Clean text** checkbox.
   h. Retain **pageview** in the **Duration** field.
   i. Click **Save**.
5. Click the **Publishing** tab.
6. Click the library you created in the previous task. The **Edit Library** section opens.
7. Scroll down to the bottom of the page, and click **Add All Changed Resources** to add all changes to your library.
8. Click **Save & Build for Development,** as shown:



Edit Library

Status

Development

Name

New Library

Environment

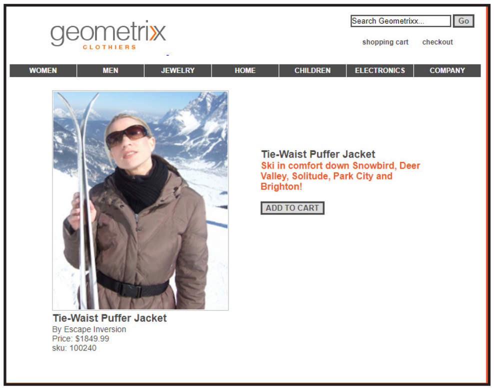
Dev (development)

Library Contents

Notice the green dot. This indicates that the library now has a build.

9. Open your demo site.

10. Go to the **WOMEN** > **Activewear** page, and click one of the products. For example, click Tie-Waist Puffer Jacket.



Tie-Waist Puffer Jacket
Ski in comfort down Snowbird, Deer Valley, Solitude, Park City and Brighton!

Tie-Waist Puffer Jacket
By Escape Inversion
Price: $1849.99
sku: 100240

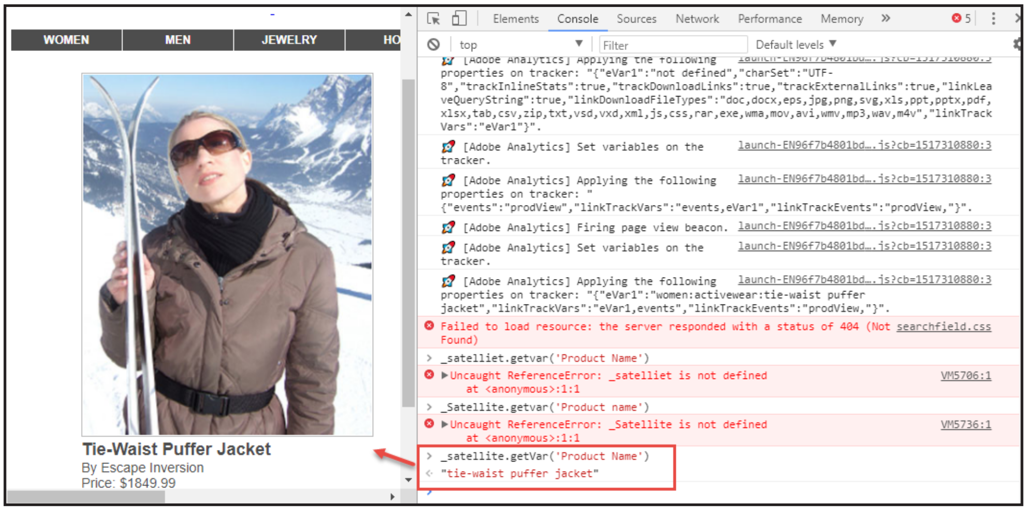
11. Right-click the page, and click **Inspect**. Alternatively, you can press Ctrl+Shift+I on your keyboard. The **Developer Tools** window opens on the right.

12. Open the **Console** tab.

13. On the **Console** tab, type **_satellite.getVar('<*your data element*> name')**. For example, type _satellite.getVar('Product Name'), where Product Name is your data element.

14. Press **Enter**.

15. Notice that **tie-waste puffer jacket** is returned, as shown, which is the product you clicked.



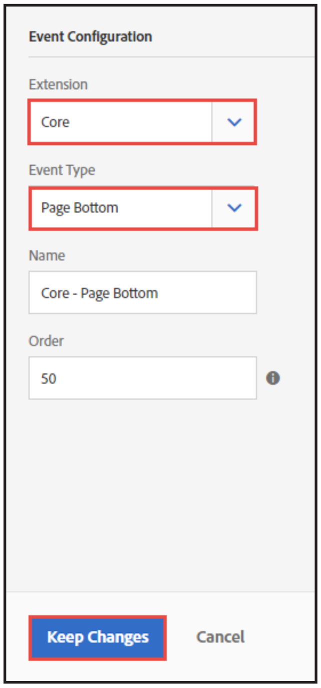
Tie-Waist Puffer Jacket
By Escape Inversion
Price: $1849.99

```
[Adobe Analytics] Applying the following       launch-EN96f7b4801bd….js?cb=1517310880:3
properties on tracker: "{"eVar1":"not defined","charSet":"UTF-
8","trackInlineStats":true,"trackDownloadLinks":true,"trackExternalLinks":true,"linkLea
veQueryString":true,"linkDownloadFileTypes":"doc,docx,eps,jpg,png,svg,xls,ppt,pptx,pdf,
xlsx,tab,csv,zip,txt,vsd,vxd,xml,js,css,rar,exe,wma,mov,avi,wmv,mp3,wav,m4v","linkTrack
Vars":"eVar1"}".
[Adobe Analytics] Set variables on the         launch-EN96f7b4801bd….js?cb=1517310880:3
tracker.
[Adobe Analytics] Applying the following       launch-EN96f7b4801bd….js?cb=1517310880:3
properties on tracker: "
{"events":"prodView","linkTrackVars":"events,eVar1","linkTrackEvents":"prodView,"}".
[Adobe Analytics] Firing page view beacon.     launch-EN96f7b4801bd….js?cb=1517310880:3
[Adobe Analytics] Set variables on the         launch-EN96f7b4801bd….js?cb=1517310880:3
tracker.
[Adobe Analytics] Applying the following       launch-EN96f7b4801bd….js?cb=1517310880:3
properties on tracker: "{"eVar1":"women:activewear:tie-waist puffer
jacket","linkTrackVars":"eVar1,events","linkTrackEvents":"prodView,"}".
Failed to load resource: the server responded with a status of 404 (Not searchfield.css
Found)
> _satelliet.getvar('Product Name')
▶Uncaught ReferenceError: _satelliet is not defined                          VM5706:1
    at <anonymous>:1:1
> _Satellite.getvar('Product name')
▶Uncaught ReferenceError: _Satellite is not defined                          VM5736:1
    at <anonymous>:1:1
> _satellite.getVar('Product Name')
<· "tie-waist puffer jacket"
```

Similarly, go to other product pages and verify the results.

Task 1: Create rules and verify

1. Open the **Rules** tab in Launch.
2. Click **Create New Rule**. The **Create New Rule** page opens. If you have created a rule before, click **Add Rule**. The **Create New Rule** section opens.
3. In the **Name** field, enter **Page Load - Bottom**.
4. To add an event that will trigger the rule, click **Add** under **EVENTS**. The **Event Configuration** section opens.
5. In the **Extension** field, retain **Core** as the extension.
6. From the **Event Type** drop-down list, select **Page Bottom**.
7. Click **Keep Changes,** as shown:



Event Configuration

Extension: Core

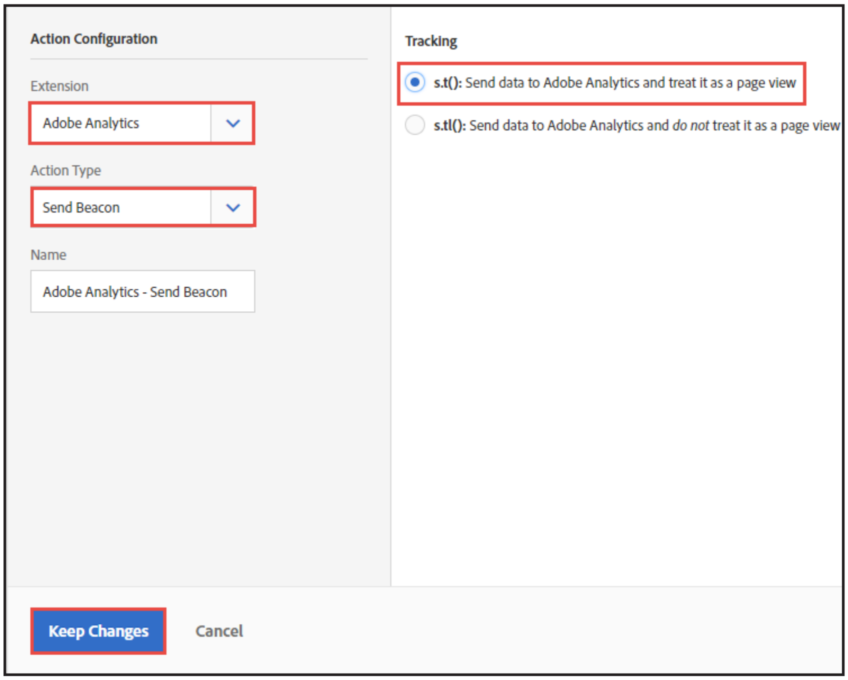
Event Type: Page Bottom

Name: Core - Page Bottom

Order: 50

Do not specify a **CONDITION** so the rule will trigger on all pages.

8. Click **Add** under **ACTIONS**. The **Action Configuration** page opens.
9. From the **Extension** drop-down list, select **Adobe Analytics**.
10. From the **Action Type** drop-down list, select **Send Beacon**.
11. Ensure that **Tracking is** set to s.t(): Send data to Adobe Analytics and treat it as a new page.
12. Click **Keep Changes,** as shown:



Action Configuration

Extension: Adobe Analytics

Action Type: Send Beacon

Name: Adobe Analytics - Send Beacon

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

13. Click **Save** to save the rule.

Add another rule called Product details - top

14. On the **Rules** tab, click **Add New Rule**. The **Create New Rule** section opens.
15. In the **Name** field, enter **Product Details - Top**.
16. To add an event that will trigger the rule, click **Add** under **EVENTS**. The **Event Configuration** page opens.
17. In the **Extension** field, retain **Core** as the extension.
18. From the **Event Type** drop-down list, select **Library Loaded (Page top)**.
19. Click **Keep Changes**.
20. Add a condition by clicking **Add** under **CONDITIONS**. The Condition Configuration section opens.

21. In the **Extension** field, retain **Core** as the extension.

22. From the **Condition Type** drop-down list, select **Data Element**.

23. From the database icon on the right, select **Product Name** from the **Data Element** field.

24. In the **has the value** field, enter ([^\s])

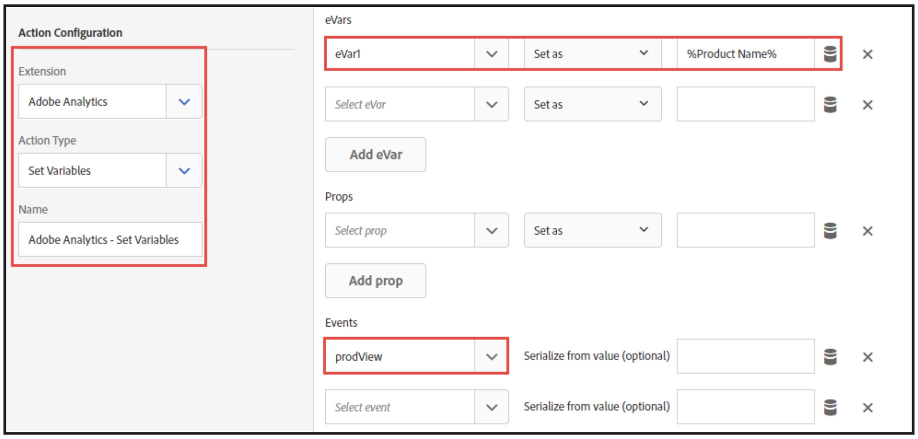
25. Click **Keep Changes**.

26. To add an action, click **Add** under **ACTIONS**. The **Action Configuration** section opens.

27. From the **Extension** drop-down list, select Adobe Analytics.

28. From the **Action Type** drop-down list, select Set Variables.

29. From the **Events** drop-down list on the right, select prodView, as shown:



Action Configuration

Extension: Adobe Analytics

Action Type: Set Variables

Name: Adobe Analytics - Set Variables

eVars

eVar1	Set as	%Product Name%
Select eVar	Set as	

Add eVar

Props

Select prop	Set as	

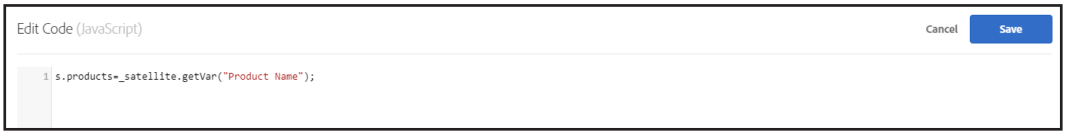
Add prop

Events

prodView	Serialize from value (optional)	
Select event	Serialize from value (optional)	

30. Scroll down to the end of the page. Under C**ustom Code**, click **</> Open Editor.** The editor opens.

31. Type s.products=_satellite.getVar("Product Name"); in the editor.



Edit Code (JavaScript)

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

32. Click **Save**.

33. Click **Keep Changes**.

34. Click **Save** to save the rule.

35. Open the **Publishing** tab.

36. Click the library you created in the previous task. The **Edit Library** section opens.

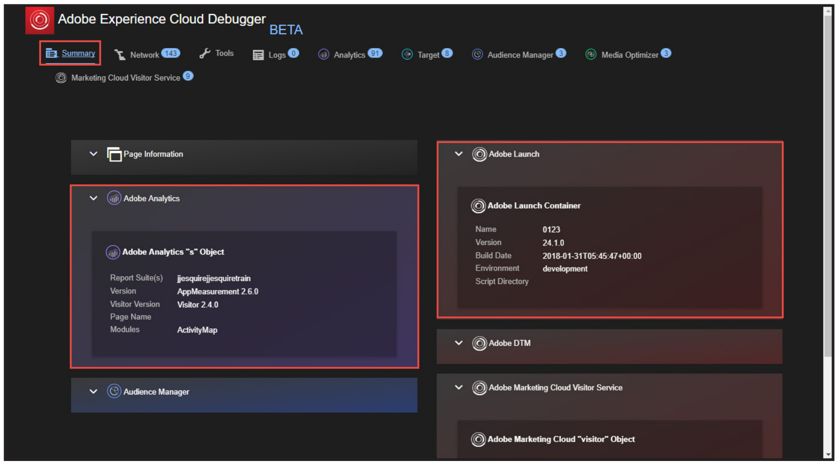
37. Scroll down to the bottom of the page, and click **Add All Changed Resources** to add all changes to your library.

38. Click **Save & Build for Development**. Notice the green dot. This indicates that the library now has a build.

39. Open your practice site.

40. Open the Experience Cloud Debugger from your ReadyTech browser Bookmark bar.

41. Under the **Summary** tab, notice the Analytics and Launch tabs. It includes basic information such as Name, Version, Environment, and Report Suite, as shown.



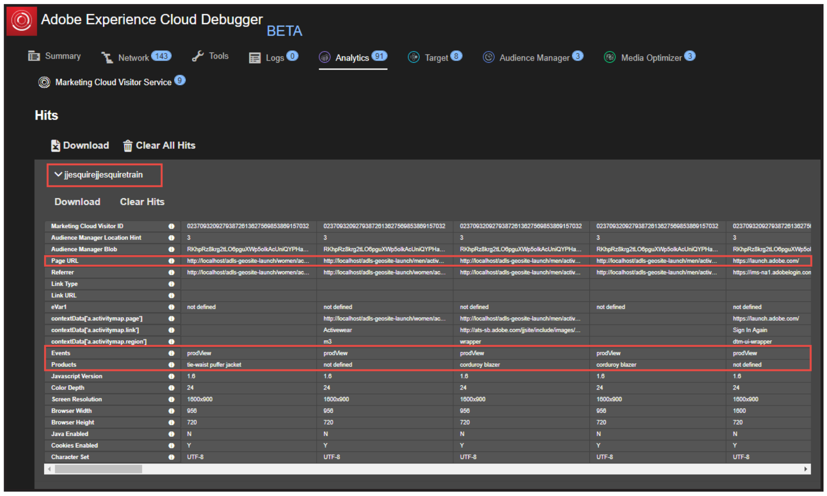
Adobe Analytics "s" Object

Report Suite(s)	jjesquirejjesquiretrain
Version	AppMeasurement 2.6.0
Visitor Version	Visitor 2.4.0
Page Name	
Modules	ActivityMap

Adobe Launch Container

Name	0123
Version	24.1.0
Build Date	2018-01-31T05:45:47+00:00
Environment	development
Script Directory	

42. Open the **Analytics** tab. Under your report suite, notice the URL, event, products, and other details, as shown:



jjesquirejjesquiretrain

Marketing Cloud Visitor ID	02370932092793872613627569853869157032	02370932092793872613627569853869157032	02370932092793872613627569853869157032	02370932092793872613627569853869157032	023709320927938726136275
Audience Manager Location Hint	3	3	3	3	3
Audience Manager Blob	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp5olkAcUniQYPHa...	RKhpRz8krg2tLO6pguXWp5ol
Page URL	http://localhost/adls-geosite-launch/women/ac...	http://localhost/adls-geosite-launch/men/activ...	http://localhost/adls-geosite-launch/men/activ...	http://localhost/adls-geosite-launch/men/activ...	https://launch.adobe.com/
Referrer	http://localhost/adls-geosite-launch/women/ac...	http://localhost/adls-geosite-launch/women/ac...	http://localhost/adls-geosite-launch/men/activ...	http://localhost/adls-geosite-launch/men/activ...	https://ims-na1.adobelogin.con
Link Type					
Link URL					
eVar1	not defined	not defined	not defined	not defined	not defined
contextData['a.activitymap.page']		http://localhost/adls-geosite-launch/women/ac...	http://localhost/adls-geosite-launch/men/activ...		https://launch.adobe.com/
contextData['a.activitymap.link']		Activewear	http://ats-sb.adobe.com/jjsite/include/images/...		Sign In Again
contextData['a.activitymap.region']		m3	wrapper		dtm-ui-wrapper
Events	prodView	prodView	prodView	prodView	prodView
Products	tie-waist puffer jacket	not defined	corduroy blazer	corduroy blazer	not defined
Javascript Version	1.6	1.6	1.6	1.6	1.6
Color Depth	24	24	24	24	24
Screen Resolution	1600x900	1600x900	1600x900	1600x900	1600x900
Browser Width	956	956	956	956	1600
Browser Height	720	720	720	720	720
Java Enabled	N	N	N	N	N
Cookies Enabled	Y	Y	Y	Y	Y
Character Set	UTF-8	UTF-8	UTF-8	UTF-8	UTF-8

43. In the practice site, click on few pages and products and return to the Experience Cloud Debugger. Observe the result.

Additional Information

Refer the links below for:

Event, Condition, Exception, and Action types available under Core extension

Action types available under Analytics extension

Action types available under Experience Cloud ID extension