http://blogs.adobe.com/digitalmarketing/wp-content/themes/pagelines-template-theme/images/adobe.png`



Adobe

Consulting

[CLIENT NAME]

Analytics Technical Specification

Contents

[Overview 3](#_Toc418592495)

[Introduction to Data Layers 3](#_Toc418592496)

[DTM: Getting Started 3](#_Toc418592497)

[How to use this document 4](#_Toc418592498)

[How-to: "Variables Used in this Solution" 4](#_Toc418592499)

[How-to: "Deployment Instructions" 4](#_Toc418592500)

[How-to: "Tracking Configuration" 5](#_Toc418592501)

[How-to: "Validation" 7](#_Toc418592502)

[Solution Components 9](#_Toc418592503)

[Site Content Measurement 9](#_Toc418592504)

[Appendix: Validation 15](#_Toc418592505)

[Validation: Using the Adobe Debugger 15](#_Toc418592506)

[Validation: Debugging with Charles 18](#_Toc418592507)

[Validation: Debugging DTM 22](#_Toc418592508)

[Appendix: DTM 23](#_Toc418592509)

[DTM Baseline Setup and Configuration 23](#_Toc418592510)

[Data Layer: digitalData 26](#_Toc418592511)

[Mapping Data Layer to Data Elements 27](#_Toc418592512)

[Appendix: Report Validation 29](#_Toc418592513)

[Adobe Analytics Interface 29](#_Toc418592514)

[Analytics Reports 30](#_Toc418592515)

[Validation Techniques by Report Type 33](#_Toc418592516)

# Overview

This document is intended as the standard guide for deploying Adobe Analytics on your site using Dynamic Tag Manager (DTM). The majority of this document's content is technical in nature as the intended audiences are teams that will be implementing the solution and validating the implementation.

This document is divided up into different sections, each of which describes a different component of the overall Analytics solution. Each component is designed to cover specific business need based on what was captured in the Business Requirement Document (BRD).

This document has instructions for both sides of a DTM deployment:

* setting a JavaScript *data layer* on your page (usually done by developers)
* configuring tracking to turn that data layer into meaningful reports (using DTM)

## Introduction to Data Layers

A "data layer" is a framework of JavaScript objects your developers would put on your pages that can be used by tracking tools (including tag management systems like DTM) to populate reports.

Implementing a data layer on your site will give you the ultimate control and flexibility over your implementation, and allow for the easiest maintenance going forward. The names of these JavaScript objects are theoretically arbitrary, but the best practice is to use something consistent and predictable. Your developers may already have a data layer, or a preference for the format. There are a few different standards the tracking community has created as a starting point- this technical specifications document will use the W3C standard "digitalData" object that was created by a group of experts from major internet tracking companies and will be accepted by the widest variety of tracking technology (in case you ever need to use the data layer for more than this DTM implementation).

For more information on the Data Layers, please see the appendix.

## DTM: Getting Started

By this stage in the engagement, you should have received a Getting Started Guide that walked you through getting access to DTM and a proper DTM configuration at a high level. You should have a DTM login and at least one property configured.

You should also have received from your consultant the embed code snippets that need to go on each page that will have tracking. For more information on setting up your property and your Adobe Analytics tools, please see the appendix.

# How to use this document

Each Solutions Component section that follows details one piece of your implementation. Each section has 4 key areas that provide unique information that will be helpful in ensuring a successful deployment.

## How-to: "Variables Used in this Solution"

This subsection is a listing of the Data Layer variables to be deployed during the implementation. This information is useful for the client's development/implementation team as a quick reference. It provides a high-level look at where the tracking will come from (for example, a JavaScript data layer, a query parameter, custom code…), and where that data will show up in your Analytics reports.

A complete guide to each variable can be found in your Solution Design Reference (or SDR), which includes the business requirements, variables used to track the requirements, Tech Spec solution each variable corresponds to, the tracking methodology used for each variable, Data Element/Object/Rule if applicable as well as scope and implementation status of each variable.

### Example: Variables in this Solution

|  |  |  |  |
| --- | --- | --- | --- |
| Tracking Variable | Used For | Source | Report |
| s.pageName | Pages | data layer | Site Content > Pages |

## How-to: "Deployment Instructions"

This subsection contains detailed instructions for the development team on how to deploy the data layer and on-page DTM code needed for this solution. The table provided details each JavaScript digitalData sub-object to create, followed by a description and an example. Example code blocks for that solution follow, though your developers may use a different syntax to create the same object.

This subsection may also detail how to fire beacons after a page has loaded, using either DTM *Direct Call Rules* or DTM *Event Based Rules*.

**Direct Call Rules** allow developers to choose when a DTM rule (and therefore Analytics tracking beacon) should fire, by setting a *\_satellite.track()* function in the code on the page. This gives developers and analysts complete control and predictability over when these rules fire, but requires developers to make changes to the page. See the [Experience Cloud documentation portal](https://marketing.adobe.com/resources/help/en_US/dtm/t_rules_direct_conditions.html) for more information on Direct Call Rules.

**Event Based Rules** use existing aspects of the page (called the DOM) to listen for certain behaviors and fire rules accordingly. This often relies on CSS selectors (meaning they may break if the CSS/HTML of a page changes) and may conflict with existing JavaScript event listeners already on the page. Yet this method does allow tracking to happen without changes to the existing page. See the [Experience Cloud documentation portal](https://marketing.adobe.com/resources/help/en_US/dtm/rules.html) for more information on Event Based Rules.

### Example: Deployment Instructions

Set the following objects on every page of the site:

|  |  |
| --- | --- |
| Data Object | Description |
| **digitalData.page.pageInfo.pageName** | Set a clear, descriptive page name. |

#### Code Examples

On every page of the site, set the following, after the digitalData object has been defined somewhere on the page:

digitalData.page={

pageInfo:{

pageName:"Electronics>Computers>MacbookPro:ProductDetails",

}

}

### Walk-through: Code Example

The developer would use the above documentation to know to set digitalData.page.pageInfo.pageName on all pages of the site. They may need further guidance from the owners of the reporting on what values should be set or what format to follow, as these will show directly in your reports.

|  |
| --- |
| NOTE |
| * Instead of following the exact syntax of the Code Example, the developer may choose a different format. For instance, the following does not match the code example provided above, but would also set digitalData.page.pageInfo.pageName to "electronics>computers>macbook pro:product details":   digitalData.page = {};  digitalData.page.pageInfo = {};  digitalData.page.pageInfo.pageName = "Electronics>Computers>MacbookPro:ProductDetails";   * Both formats are acceptable. |

See the appendix for more details on using digitalData JavaScript objects.

## How-to: "Tracking Configuration"

This subsection documents how the data on your page is configured and turned into Analytics tracking beacons. This may involve setting up rules and data elements in DTM, making changes to your Analytics Tool JavaScript Library (formerly called an "s\_code"), or using processing rules in the Adobe Analytics admin tool.

### Example: DTM Data Elements

Create a data element for each of the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DTM Data Element Name | Element Type | JS Object Path/Parameter | For use in which DTM Rules | AA Variable Mapping | Notes |
| content: page name | JS Object | digitalData.page.pageInfo.pageName | "content: all pages" | s.pageName | Should have a default value of "no pageName found" |

### Example: DTM Page Load Rules

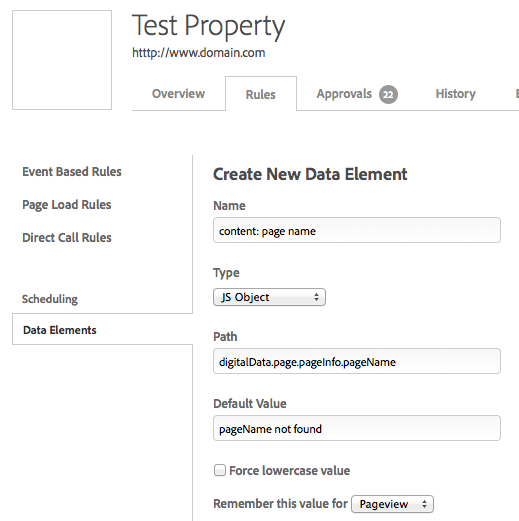
Create a Page Load Rule in DTM with the following specifications:

|  |  |  |
| --- | --- | --- |
| Rule Name | Rule Condition | Notes |
| content: all pages | No condition (fires on all pages) | This rule should contain all global logic. |

### Tracking Configuration: Walking-through

#### Data Elements

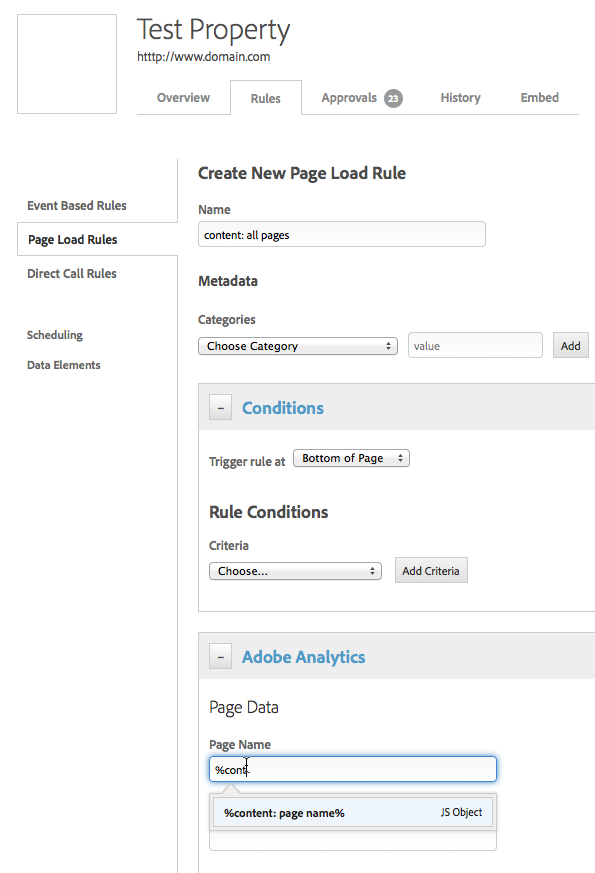
Using the above documentation, I would create a new Data Element with the name "content: page name", set the type to "JS object", and reference the full path of the digitalData object. Unless otherwise instructed, I would leave the Default Value field blank, and tell DTM to use the default option "Remember this value for *pageview*". I would save the rule, then create any others called for in this solution component.



For more information on creating Data Elements, see the [Experience Cloud documentation portal](https://marketing.adobe.com/resources/help/en_US/dtm/data_elements.html).

#### Page Load Rules

Next, I would create a new Page Load Rule with the name "content: all pages". Based on the solution component documentation, it would have no conditions. I see in the Data Elements table that the "content: page name" Data Element is meant to be mapped to the pageName variable in this "content: all pages" rule, so I would find the field for "Page Name", press the % sign, and select my newly-created "content: page name" data element from the list. After mapping all variables called for in the solution components, I would save my rule.



Please note that Page Load Rules stack- if a given page meets the conditions of multiple rules, those rules will combine to create one beacon. For instance, my search results page may fire both my "content: all pages" rule (which sets pageName) AND my "search: search results" rule (which sets a custom eVar with the search term) - DTM would combine this into one pageView beacon with both pageName and the custom eVar set.

For more information on creating Page Load Rules, see the [Experience Cloud documentation portal.](https://marketing.adobe.com/resources/help/en_US/dtm/etv_rules_page.html)

## How-to: "Validation"

The ultimate sign of a successful implementation is accurate data within the Adobe Analytics reporting tools. This subsection describes how to test and validate the reports to ensure accuracy and validity of the data.

Once data elements are configured, variables are mapped and rules are saved, you can test tracking directly on your site. Be sure you are viewing your DTM staging library and having debugging turned on so you can see your DTM rules fire in your console log. If the rule works, you can then validate that the appropriate variables are being set by using the Adobe Debugger. Lastly, you can use this guide to validate that each report is receiving data as expected.

Once any rules and data elements are validated, you can "approve" the rule in DTM and publish your library to your production DTM library and site.

### Example: Beacon Validation

Use the debugger to check the following:

* pageName is set correctly and identifies the page being viewed.

If the beacon doesn't fire, or doesn't include the variables expected, then examine the possible causes:

#### Rule Validation

Consult the table from that solution component's "DTM Page Load Rules" to see which should fire. With DTM debugging turned on, you should be able to see when the rules fire in any JavaScript console.   
If the rule is not firing, check that your conditions for the rule in DTM are set correctly. If the rule is firing, but the variables don't show the expected values, then check the data object has the values expected.

#### Data Object Validation

Consult the table from that solution component's "DTM Data Elements" to see which JavaScript objects map to which Analytics Variable you should be seeing in the debugger.

Once the beacon is validated, you can check that the data is showing in your Analytics reports.

### Example: Report Validation

Check that each report tied to this solution has valid data for the metrics and notes mentioned below:

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Location | Metrics | Notes |
| pageName | Site Content > Pages | Page Views, Visits, Unique Visitors | Check that no rows begin with "http". |

### How-to: Report Validation

Open up each of the reports in Adobe Analytics and check for any unexpected values. Pay attention to the Data Range. For different reports, it may make sense to check against different metrics, or to break one report down by another metric to validate the relationship between those data points.

Refer to the "Report Validation" section in the Appendix of this document for further details on how to navigate through the R&A interface to pull the reports.

# Solution Components

## Site Content Measurement

This solution allows the business unit to evaluate the effectiveness of the site's pages, including their influence on conversion, landing page bounce rate and exit rate as well as gives the marketers the ability to drill down on each level of your site hierarchy. To do that requires that every page of the site be tagged with an effective page name and variables to represent each level of the content hierarchy.

### Variables in This Section

The below table consists of variables used for this solution.

|  |  |  |  |
| --- | --- | --- | --- |
| Analytics Variable | Used For | Source | Report |
| s.pageName | Pages | data layer | Site Content > Pages |
| s.channel | Site Section | data layer | Site Content > Site Sections |
| s.prop1 | Site Subsection 1 | data layer | Custom Traffic > Custom Traffic 1-10 > Sub Section 1 |
| s.prop2 | Site Subsection 2 | data layer | Custom Traffic > Custom Traffic 1-10 > Sub Section 2 |
| s.prop3 | Previous Page | tool code | Custom Traffic > Custom Traffic 1-10 > Previous Page |
| s.prop4 | Page Type | data layer | Custom Traffic > Custom Traffic 1-10 > Page Type |
| s.prop5 | Tap Type | data layer | Custom Traffic > Custom Traffic 1-10 > Tab Type |
| s.prop6 | Tab Name | data layer | Custom Traffic > Custom Traffic 1-10 > Tab Name |
| s.eVar1 | Page Name (conversion) | data layer | Custom Conversion > Custom Conversion 1-10 > Page Name |
| s.eVar2 | Page URL | tool code | Custom Conversion > Custom Conversion 1-10 > Page URL |
| s.eVar3 | Previous Page | tool code | Custom Conversion > Custom Conversion 1-10 > Previous Page |
| s.eVar4 | Tab Type | copied from prop5 | Custom Conversion > Custom Conversion 1-10 > Tab Type |
| s.eVar5 | Tab Name | copied from prop6 | Custom Conversion > Custom Conversion 1-10 > Tab Name |
| event1 | Custom Page View | fires on all pages | Site Metrics > Custom Events > Custom Events 1-20 > Custom Page View |

### Deployment Instructions

#### Data Layer - All Pages

Set the following objects on every page of the site:

|  |  |
| --- | --- |
| Data Object | Notes |
| **digitalData.page.pageInfo.pageName** | Set a clear, descriptive page name. Consider including the type or category of page to start the page name. Continue the page name with each level of the site hierarchy until the final descriptive portion of the page. Use colon ":" as the delimiter between each portion of the page. |
| **digitalData.page.category.pageType** | Set this variable to the pageType |
| **digitalData.page.category.primaryCategory** | Set this variable to the top level category |
| **digitalData.page.category.subCategory1** | Set this variable to the first level subsection of the page. If there is no sub section, set the value to "n/a". |
| **digitalData.page.category.subCategory2** | Set this variable to the second level subsection of the page. If there is no sub section, set the value to "n/a". |

|  |
| --- |
| Page Naming Best Practice |
| * **Context -** Include the directory structure or content hierarchy in the page name to help users understand where the page "lives" within the site. This also allows for simplified report filtering. * **Clarity -** Ensure the page name is clear and easily identifiable for infrequent users. This will promote faster adoption of Adobe Analytics by business users and also allow for easier, more efficient reporting. * **Conciseness -** Keep the page name as short as possible to maximize limited character space. This is also important for clean, simple reports but should not take precedence over context and clarity. |

Example data object for "home page":

digitalData.page={

pageInfo:{

pageName:"HomePage",

},

category:{

pageType:"Home",

primaryCategory:"Home",

subCategory1:"n/a",

subCategory2:"n/a"

}

}

Example data object for "product detail page":

digitalData.page={

pageInfo:{

pageName:"Electronics:TVandVideo:Dell:Inspiron",

},

category:{

pageType:"pdp",

primaryCategory: "electronics",

subCategory1:"TVandVideo",

subCategory2:"Dell"

}

}

Example data object for "category home page":

digitalData.page={

pageInfo:{

pageName:"Category:Technology",

},

category:{

pageType:"category",

primaryCategory:"technology",

subCategory1:"n/a",

subCategory2:"n/a"

}

}

#### Tab Clicks

Since a tab click happens after the initial page view beacon, tracking will need to be configured to fire a DTM rule based on the user's actions. This can be done using an Event-Based Rule, which listens for certain user actions based on the CSS and DOM, or using a Direct Call Rule, detailed below.

**Direct Call Rule: Data Layer**

Set the following variables when the user clicks a tab, followed by a Direct Call Rule trigger, such as \_satellite.track("tab click"):

|  |  |
| --- | --- |
| Data Object | Notes |
| **digitalData.tab.type** | When a user clicks on a tab, set the variable to the type of the tab being clicked. |
| **digitalData.tab.name** | When a user clicks on a tab, set the variable to the name of the tab being clicked. |

**Direct Call Rule: Complete Example**



Set the following on the click of a tab, and dynamically populate the name of the link clicked:

digitalData.tab={

type:"product detail",

name:"how to buy"

}

\_satellite.track('tab click');

### Configuring Tracking

#### DTM Data Elements

Create a data element for each of the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data Element | Element Type | JS Object Path/Parameter | DTM Rules | AA Variable Mapping | Notes |
| content: page name | JS Object | digitalData.page.pageInfo.pageName | "content: all pages" | s.pageName | Should have a default value of "no pageName found" |
| content: page type | JS Object | digitalData.page.category.pageType | "content: all pages" | s.prop4 |  |
| content: site section | JS Object | digitalData.page.category.primaryCategory | "content: all pages" | s.channel |  |
| content: sub section 1 | JS Object | digitalData.page.category.subCategory1 | "content: all pages" | s.prop1 |  |
| content: sub section 2 | JS Object | digitalData.page.category.subCategory2 | "content: all pages" | s.prop2 |  |
| content: tab type | JS Object | digitalData.tab.type | direct call rule: "tab click" | s.prop5 |  |
| content: tab name | JS Object | digitalData.tab.name | direct call rule: "tab click" | s.prop6 |  |

#### Create DTM Rules

Create the following DTM rules:

|  |  |  |  |
| --- | --- | --- | --- |
| Rule Type | Rule Name | Conditions | Special Instructions |
| Page Load | content: all pages | no conditions set (fires on all pages) | In addition to mapping the newly-created data elements to the appropriate variables, set "event1", and set eVar4 to duplicate the value of prop5, and eVar5 to duplicate the value of prop6. |
| Direct or Event Based | tab click | Direct Call Rule: condition of "tab click" Event-Based Rule: based on "click" of the tab HTML element | Set to "s.tl". Map the data elements to the appropriate value |

#### Add to code library

The following code has been placed in your tool library within the s\_doPlugins function.

/\* Site Content Measurement \*/

if(s.pageName){

s.eVar1="D=pageName";

s.eVar2="D=g";

s.events=s.apl(s.events,"event1",",",2)

}

/\* For Previous Page value \*/

s.prop3=s.getPreviousValue(s.pageName,'gpv','');

if(s.prop3){

s.eVar3="D=c3";

}

### Validation

#### Implementation Testing

Test the variables listed in this section by analyzing incoming image requests and looking for expected values for the variables listed in the table at the beginning of this section. Refer to the "Implementation Testing" section in the Appendix of this document for further details on the tools that can be leveraged to ensure the right variables and values are being set.

Use the debugger to check the following:

* pageName is set correctly and identifies the page being viewed.
* channel is set to the first level of hierarchy
* prop1 is set to the second level of hierarchy
* prop2 is set to the third level of hierarchy.
* Prop3 is set to the page type
* eVar1 is set to the same value as pageName
* eVar2 is set to the URL of the page
* eVar3 and prop3 are not set if the page being viewed is the first page of the visit or it's set to the previous page viewed.
* The events list includes event1

Click on a tab that fires a custom image request set by the two custom functions listed in this section. Use the debugger to check the following:

* eVar4 and prop5 are set to type of tab clicked.
* eVar5 and prop6 are set to name of tab clicked.

#### Report Review

Check that each report tied to this solution has valid data for the metrics and notes mentioned below:

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Location | Metrics | Special Instructions |
| s.pageName | Site Content > Pages | Page Views, Visits, Unique Visitors | * No row values begin with 'http'. Since Adobe Analytics automatically assigns a page's URL as the page name when one is not assigned, the presence of straight URLs (beginning with http) in the Pages report indicates that some pages do not have s.pageName set. * Each value in each row identifies a single, unique page on the site. * The values in each row are clear. Analysts looking at the Pages report can immediately recognize which page each row refers to. * The values in each row are well structured and include the page type, breadcrumb trail, or other values that would help identify the page's location within the site's content hierarchy. |
| s.channel | Site Content > Site Sections | Page Views, Visits, Unique Visitors | * None |
| s.prop1 | Custom Traffic > Custom Traffic 1-10 > Sub Section 1 | Page Views, Visits, Unique Visitors | * None |
| s.prop2 | Custom Traffic > Custom Traffic 1-10 > Sub Section 2 | Page Views, Visits, Unique Visitors | * None |
| s.prop3 | Custom Traffic > Custom Traffic 1-10 > Previous Page | Page Views, Visits, Unique Visitors | * Correlate the Pages report with Previous Page report to understand the page the user was on before navigating to the current page. The Page Views metric (column) should reflect the number of times visitors viewed the previous page to get to the current page. |
| s.prop4 | Custom Traffic > Custom Traffic 1-10 > Page Type | Page Views, Visits, Unique Visitors | * None |
| s.prop5 | Custom Traffic > Custom Traffic 1-10 > Tab Type | Page Views, Visits, Unique Visitors | * Correlate Tab Type with Tab Detail to ensure there is always a relation between the two. |
| s.prop6 | Custom Traffic > Custom Traffic 1-10 > Tab Name | Page Views, Visits, Unique Visitors | * Correlate Tab Type with Tab Detail to ensure there is always a relation between the two. |
| s.eVar1 | Custom Conversion > Custom Conversion 1-10 > Page Name | Page View | * Page Views metric should not be attributed to a row with the "none" value. |
| s.eVar2 | Custom Conversion > Custom Conversion 1-10 > Page URL | Page View | * Page Views metric should not be attributed to a row with the "none" value. * Subrelate Page Name to Page URL to ensure there is always a relation and that no two unique URLs (excluding query parameters) have the same page name. |
| s.eVar3 | Custom Conversion > Custom Conversion 1-10 > Previous Page | Page View | * Subrelate Page Name to Previous Page to ensure there is always a relation and understand how users are getting to the page of interest. * The "none" value could represents the first page of the visit, since this variable is not set on the first page of visit. |
| s.eVar4 | Custom Conversion > Custom Conversion 1-10 > Tab Type | Instances | * Subrelate Tab Type with Tab Detail to ensure there is always a relation between the two. |
| s.eVar5 | Custom Conversion > Custom Conversion 1-10 > Tab Name | Instances | * Subrelate Tab Type with Tab Detail to ensure there is always a relation between the two. |
| event1 | Site Metrics > Custom Events > Custom Events 1-20 > Custom Page View | N/A | * None |

# Appendix: Validation

## Validation: Using the Adobe Debugger

The Adobe Experience Cloud Debugger is a small JavaScript-based utility that you can add to any browser and will be your primary "quick source" to go to whenever you need to test and validate the implementation of any Adobe Experience Cloud product. Once activated, it will show you the contents of all server calls sent from the browser to the Adobe Experience Cloud servers

### Deployment Instructions

#### Adobe Experience Cloud Debugger Installation

Install the Adobe Experience Cloud Debugger by going through the following steps:

* Open a browser window.
* Go to any URL, for example, www.adobe.com.
* In the browser window, click Favorites/Bookmarks
* Add the page to your Favorites/Bookmarks
* Change the name of the favorite to Adobe Debugger or some other unique name that will help you identify it.
* Edit the URL of the favorite to be equal to the following:

javascript:void(window.open("","stats\_debugger","width=600,height=600,location=0,menubar=0,status=1,toolbar=0,resizable=1,scrollbars=1").document.write("<script language=\"JavaScript\" id=dbg src=\"https://www.adobetag.com/d1/digitalpulsedebugger/live/DPD.js?tnt=0&recs=0&am=0&survey=0&\"></"+"script>" + "<script language=\"JavaScript\">window.focus();</script>"));

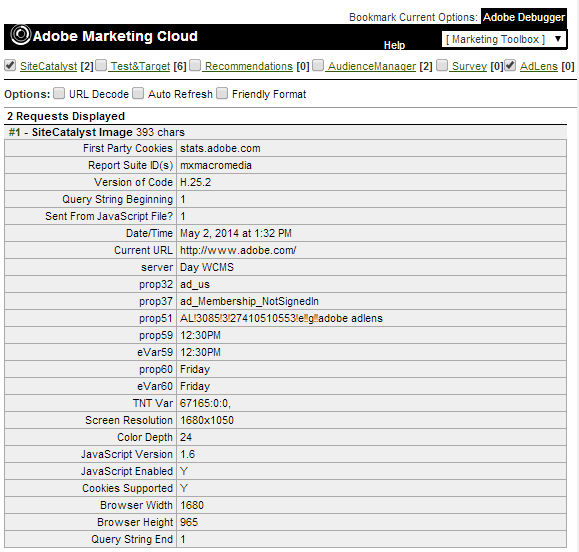
#### Adobe Experience Cloud Debugger Initial Test

Via your browser, go to any page of your site that contains Adobe Analytics code on it. Activate the Debugger by selecting the bookmark/favorite that you just created via the Deployment Instructions above. When the debugger is activated, a popup window should appear with the details of any Analytics server calls sent out from the page.

Any server call, by its very nature, resembles a URL that you could type into your address bar (e.g. <http://www.adobe.com>). However, Adobe Analytics server calls are much more complex than a typical URL; they contain a very large amount of information that the Adobe Servers need to collect and process in order to fill up the correct reports within the Analytics interface. The Debugger helps facilitate the testing and debugging process by parsing these server calls and presenting them to you in an easy-to-read format. For example, this rather complex-looking server call…

http://stats.adobe.com/b/ss/mxmacromedia/1/H.25.2/s23849677203688?AQB=1&ndh=1&t=2%2F4%2F2014%2013%3A32%3A46%205%20360&g=http%3A%2F%2Fwww.adobe.com%2F&server=Day%20WCMS&c32=ad\_us&c37=ad\_Membership\_NotSignedIn&c51=AL%213085%213%2127410510553%21e%21%21g%21%21adobe%20adlens&c59=12%3A30PM&v59=12%3A30PM&c60=Friday&v60=Friday&tnt=67165%3A0%3A0%2C&s=1680x1050&c=24&j=1.6&v=Y&k=Y&bw=1680&bh=965&AQE=1

…will be translated into the following by the Debugger:



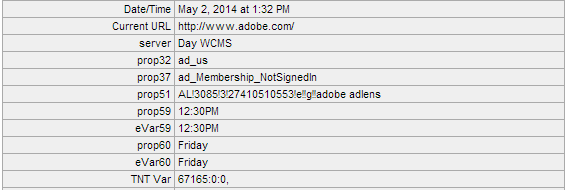
This format will make testing and debugging much easier to accomplish, but even this easy-to-read format has an intimidating appearance upon first glance. So, while going through the testing process, keep in mind that some of the contents across all server calls will maintain the exact same value

* The First Party Cookies value lets you know whether your implementation is using first party cookies to store the Analytics Visitor ID. Adobe Consulting usually recommends using First Party Cookies but does not require them in order to collect data from your site. If you are not using first party cookies, this portion of the debugger window would show something like the following instead:



* The Report Suite ID(s) value shows which report suite in the Adobe Analytics system will contain the data sent from the server call. While testing in a development or QA environment, you need to ensure that this value is not equal to the report suite ID associated with your site's production environment. Likewise, while testing in your production environment, the value should not be equal to the report suite ID associated with your dev or QA environment.
* Version of Code/Query String Beginning/Sent from JavaScript file? – these values won't need to be examined as they are simply used to designate when the actual server call contents to be processed begin to appear in the server call
* Screen Resolution/Color Depth/JavaScript Version/JavaScript Enabled/Cookies Supported/Browser Width/Browser Height – These values provide technical information about both your computer and browser and, like the portions above, do not necessarily need to be examined during the testing process.

In sum, the most important parts we need to examine while testing and debugging are everything in between:



These portions of the server call will differ across each page of the site. Later sections of this document will cover what to look for specifically while testing and debugging under certain situations.

## Validation: Debugging with Charles

The Adobe Experience Cloud Debugger is helpful for quickly reading the contents of server calls that are sent from the browser. However, the Debugger does not contain sufficient functionality to examine all server calls and also does not keep a log of all server calls produced during a visit over time. Because of this, Adobe Consulting strongly recommends that on top of using the Debugger, you use another outside web monitoring tool/packet sniffer during the testing and validation process. Adobe Consulting's preferred tool to use is the web debugging/proxy application known as Charles (available at [http://www.charlesproxy.com](http://www.yoursite.com/news/economic-news-article.jsp)).

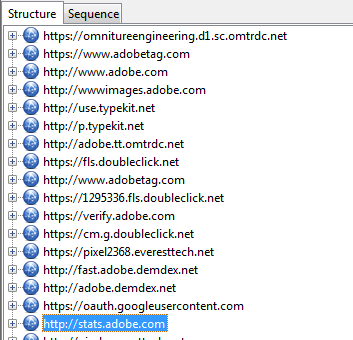
Charles' most visible feature is its HTTP monitoring functionality. Per its website's own explanation, "[a]n HTTP Monitor is an application that enables the developer to view all of the HTTP traffic between your computer and the Internet. This includes the request data… and the response data.

An HTTP Monitor is useful as it enables the developer to see the hidden communications between the web browser (or other application using HTTP) and the Internet."

In terms of debugging an Adobe Analytics implementation, Charles will not only keep a log of all server calls that are created, but also group them together under a single domain entry throughout the duration of the visit. This is especially helpful in cases where the browser sends server calls as a page unloads or sends multiple server calls at a single time.

### Testing Instructions

After downloading, installing, and running Charles, go ahead and visit any page of your website. When you do this, you will notice that any URL that your browser accesses during the visit will appears under Charles' Structure tab:

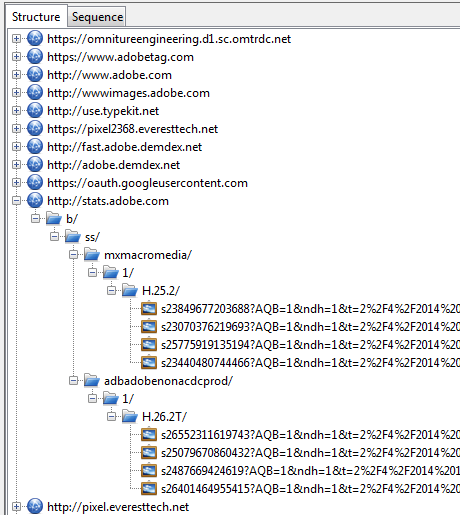


This list of URLs might appear intimidating at first, but using the Experience Cloud Debugger (in conjunction) will give you a hint as to where to begin your actual debugging within Charles. For instance, the Debugger's First Party Cookies (or Third Party Cookies) parameter shows the domain that the server calls are being sent to. For example:



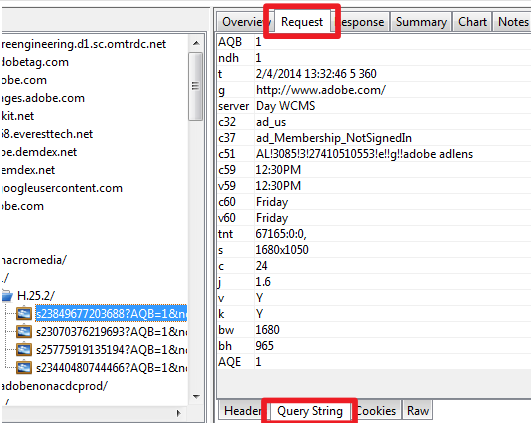


Once you know the domain where the Analytics server calls are being sent to, you can look through the list of domains recorded in Charles and should be able to find the same domain as shown in the Debugger. Once you find the domain, you may click on the plus button icon next to the domain entry to see the list of all server calls sent to that domain so far. If you compare what appears from clicking on the plus icon, you will notice how the complete domain entries in Charles match up with what you may see in the Debugger.



Once this list appears, you may click on any of the entries that have picture-based icons right next to them. Feel free to ignore any other entries under that domain, especially ones that resemble "reloading" icons; they are used mostly for identification/setup purposes when a visitor comes to your site for the first time.

As you click on each picture-based icon, you'll notice the contents of the right-hand side window in Charles will change. When this happens, be sure that the right-hand side window has both the request tab (up top) and the query string tab (at the bottom) highlighted – see the screenshot below. You may ignore the other tabs on the right-hand side window as they provide information that will not be relevant for server call debugging purposes.



You'll notice the contents of the server call in Charles don't completely matchup with what you would see if you use the Experience Cloud Debugger. Charles does not present each individual parameter of the server call in an as-easy-to-read format as the Debugger's format. All Analytics "prop" variables, for example, will show up as such in the Debugger (e.g. prop69, prop60) but will show up in Charles with the letter c instead (e.g. c59, c60).

Here is a list of the possible parameters that you will encounter during the testing and debugging process and what each variable the parameter maps to:

|  |  |
| --- | --- |
| **JavaScript Variable** | **Query String Parameter** |
| Current URL | g |
| Referring URL | r |
| pageName | pageName |
| server | server |
| channel | ch |
| contextData (begin) | c. |
| contextData (end) | .c |
| prop1 – prop75 | c1 – c75 |
| campaign | v0 |
| eVar1 – eVar75 | v1 – v75 |
| state | state |
| zip | zip |
| events | events |
| products | products |
| purchaseID | purchaseID |
| currencyCode | cc |

As hinted at above, other parameters will appear in all Analytics server calls but most likely can be ignored as they will contain data that are mostly technical in nature and not specific to the implementation.

## Validation: Debugging DTM

Adobe provides a plugin for Chrome and Firefox that allows client-side validation of DTM rules and data elements.  
See the [Experience Cloud documentation portal](https://marketing.adobe.com/resources/help/en_US/dtm/c_dtm_switch_plugins.html) for more information on getting this plugin and using it for validation.

# Appendix: DTM

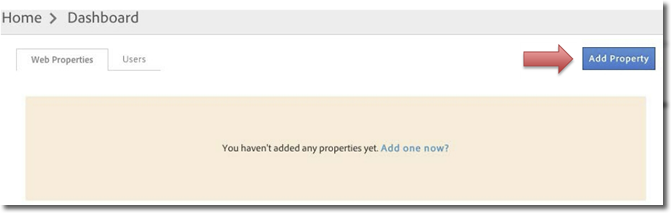
## DTM Baseline Setup and Configuration

### Deployment Instructions

A DTM implementation is divided up into properties. Before starting the actual implementation, a DTM property should be created for each user experience to be tracked. Each property has its own staging and production code library, with its own set of rules and data elements.

#### Create a Web Property

* Login into dtm.adobe.com
* Under the **Web Properties** tab on the Dashboard, click **Add Property**:



* Name the property and add the main domain.
* Specify how you would like to track traffic moving between your associated subdomains or domains.

|  |
| --- |
| NOTE |
| * You may also wish to Configure Advanced Settings relating to rule approvals, timeouts, etc. Please refer to the DTM documentation for additional details (https://marketing.adobe.com/resources/help/en\_US/dtm/web\_property.html) |

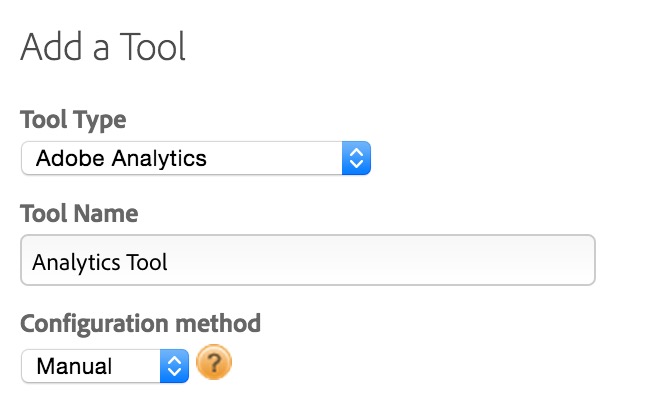
* Click Create Property
* The new property will be created in a "pending" state within your web properties tab until an admin approves and activates it. Upon activation, the property status changes from Pending Approval to Active.



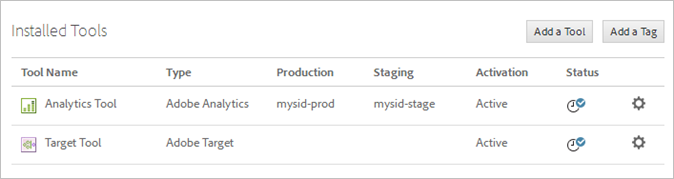
#### Add a Tool: Analytics

DTM allows you to add a variety of tools to your web property. DTM is simply the delivery mechanism. The scope of this document is limited to installing the Adobe Analytics tool

* On the **Web Properties** tab from the Dashboard, click your property
* On the **Overview** tab, click Add a Tool
* From the Tool Type menu, select Adobe Analytics



* Configure the following fields:
  + **Tool Name**: It is recommended to use "**[SITE NAME]** Analytics", but it can be anything. It is used for identification and debugging purposes.
  + **Configuration Method:** Choose between the following:
    - **Automatic:** This allows DTM to sync up with your Analytics Company settings and automatically retrieve name of configured report suites. To use this, you must provide either your Experience Cloud single sign-on ID or your Web Services Username and Shared secret.
    - **Manual:** This bypasses the configuration with your Analytics Backend and requires you to manually enter the following:
      * **Staging Account ID**: **[Dev or Staging RSID]** – This is the dev report suite ID. Always use a valid dev report suite ID. This keeps your testing data separate from your production environment
      * **Production Account ID**: **[Prod RSID]** – The account that will ultimately collect data.
      * **Tracking Server**: This should be in the form of <yoursite>.<data-center-value>.sc.omtrdc.net. For example, if the site is called "mysite.com" and the data center is San Jose – the tracking server value would be "mysite.d1.sc.omtrdc.net". For a full list of data center identifiers, review the section on RDC domains on this page: <https://helpx.adobe.com/analytics/kb/determining-data-center.html>
      * **SSL Tracking Server**: Enter your secure tracking server value. See "Tracking Server" bullet point for more information.
* Click Create Tool. The installed tool will be displayed on the Overview tab:



Visit the [online documentation on Tool Creation](https://marketing.adobe.com/resources/help/en_US/dtm/analytics_dtm.html) for more information.

#### Configure Centralized Tracking File

There are several options for deploying your centralized tracking file (either AppMeasurement.js or s\_code.js) through DTM.

* Generate or obtain AppMeasurement.js or s\_code.js in one of the two following methods
  + Obtain the configured version of AppMeasuremnt.js or s\_code.js from your Adobe Consultant
  + Generate the file in the Adobe Experience Cloud by clicking **Reports and Analytics > Admin Tools > Code Manager > JavaScript (new)**
* Within your DTM Account, on the Overview tab, click Settings https://microsite.omniture.com/t2/help/en_US/dtm/graphics/settings_gear.png next to your installed Analytics tool
* Move to the **Library Management** section, select the *Custom* radio button under the "Code Configuration" section, then click **Open Editor**
* Paste either the AppMeasurement.js code or s\_code.js code generated or obtained in the first step
* From the Overview, go to the **Approvals** tab and approve the tool that was created
* Go back to the **Overview** tab and Click **Publish Property**

|  |
| --- |
| NOTE |
| * The "Managed By Adobe" option (rather than the "Custom" option listed above) does not require an AppMeasurement or s\_code file. This option may be preferable for the most basic implementations or if timelines are critical. However, there are long-term benefits to having access to your centralized tracking file accessible in DTM. For any phase beyond a base deployment, it is recommended to use the "Custom" option in the Library Management section. * There are other options listed in the Library Management section that control the timing of Analytics calls and scoping of Adobe Analytics' "s" object. Coordinate with your Adobe Consultant for a full understanding of these settings and options. |

#### Deploy Embed Codes to Site

Every unique property has unique embed codes. It's very important to understand that the embed codes are the ultimate link between what is configured in the DTM interface and what ultimately is deployed to the site. Be sure to follow these directions carefully.

The embed codes control the DTM library files. By default these files will be hosted on Adobe's global CDN through Akamai with a domain name of assets.adobe.com. You may explore options around FTP Delivery and Library Download depending on your timelines, level of comfort with the configuration of each hosting option, and other factors.

* From you Web Property page click the **Embed** tab.
* Follow the instructions in the Akamai tab instructing you to insert the corresponding code snippets onto your site. The placement of these snippets is critical to the success of your DTM implementation. **Follow the directions exactly.**
* Make sure the header code for your environment (either staging or production) is in the <head> of every page of your site, ideally before other scripts in the <head> section
* Make sure the footer code for your environment (either staging or production) is just before the closing </body> tag of every page of your site
* Example code snippets might look like this:

Example code snippets might look like this:

Header Code

<script src="//assets.adobedtm.com/c7571f2b15d4bf5ede1676e58c56e42d6143d01e/satelliteLib-112d9e4c5c159a01ee3144b9db8ce73efa39ce4e.js"></script>

Footer Code

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

## Data Layer: digitalData

The digitalData document standard is documented on the [W3C site](http://www.w3.org/2013/12/ceddl-201312.pdf) and allows flexibility for creating new sub-objects not included in the standard. We will provide specific examples of how to use this standard for your implementation in this document.

Data attributes can be added or modified in the digitalData object over the course of the page load and after any subsequent dynamic events that occur on the page (i.e. AJAX calls).

The JavaScript content in the example below shows the standard syntax for defining elements as part of the digitalData object. Various sub-objects can be created and customized according to individualized tracking requirements.

Whenever possible, it's strongly preferred to define digitalData elements within the <head> tag and prior to the DTM code snippets referenced in this document. However the digitalData object is initialized, all digitalData elements must be defined prior to the closing DTM footer code snippet and/or DOMready.

This documentation provides example code that keeps different digitalData sub-object slightly separate- instead of defining all of digitalData in one block, you can set digitalData.page in one piece on all pages, then in your Product Details page, set digitalData.product in a separate piece. For instance, after completing all instructions in this document (including Site Content Measurement, and Product Details Page), my final page code might look like this:

|  |  |
| --- | --- |
|  | I would first include the DTM embed code in the head for all pages on the site:  <script src="//assets.adobedtm.com/c7571e2b15d4bf5ede1676e58c46e42d6143d01e/satelliteLib-306387c14ccdca6ddff5a25be8622b5b53bc953a.js"></script> |
| I'd need to make sure I define digitalData as a JavaScript object before referencing it for sub-objects:  digitalData={} |
| I might set global variables, like page details, in a global header:  digitalData.page={  pageInfo:{  pageName:"Category:Technology",  },  category:{  pageType: "product details page",  primaryCategory:"category",  subCategory1:"Jackets",  subCategory2:"Biker Jacket"  }  } |
| I may choose to set solution-component-specific digitalData sub-objects separately on the page:  digitalData.product:[{  productInfo:{  productName: "Nikon SLR Camera",  productID: "sku12345",  manufacturer: "Nikon"  }  }] |
| If unable to track the Add to Cart through an Event-Based rule, or for extra control over when that extra beacon fires, we may choose to set a Direct Call Rule on the click of the Add to Cart button: <a class="btn btn-primary" role="button" href="#" onClick="**\_satellite.track('cart add')**">ADD TO CART</a> |
| Lastly, after all digitalData sub-objects have been set, I'd include the DTM embed page bottom code just above the closing </body> tag:  <script type="text/javascript">\_satellite.pageBottom();</script> |

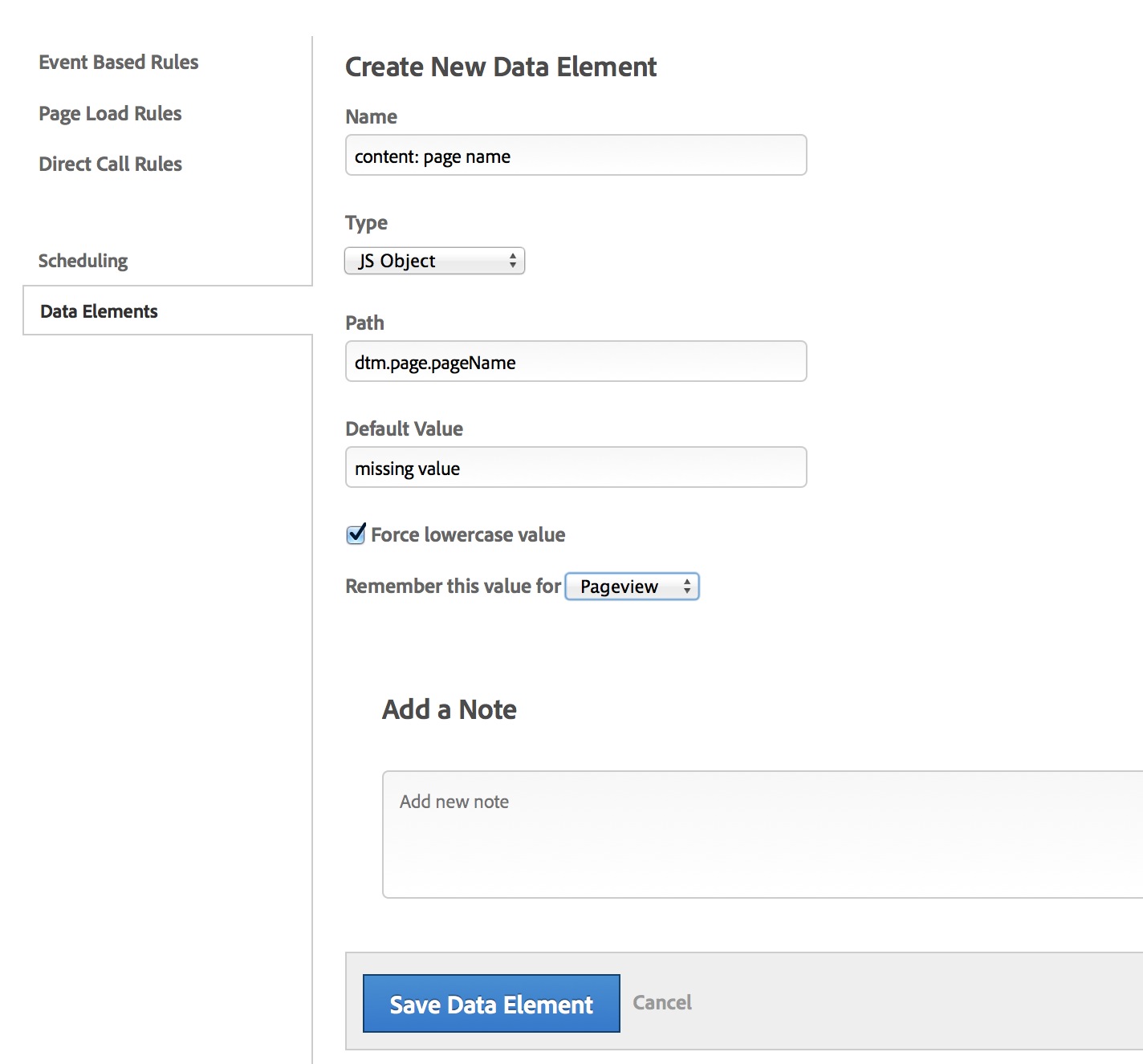
## Mapping Data Layer to Data Elements

Mapping data layer variables to data elements is a crucial part of any DTM implementation. There are a variety of ways to derive data to populate DTM data elements. One common and repeatable way to do this is by leveraging the data stored in the dtm data layer object.

Within your DTM account create data elements and map the variables in the "Data Layer Variable" to the "DTM Data Element" column values as listed in the beginning of each Solution Section.

* Within your property’s home screen, create a Data Element by clicking on **Rules** then **Data Elements** then **Create New Data Element**.
* The Name field will correspond to the "DTM Data Element" value listed in the table at the beginning of this section.
* Always select "JS Object" for the **Type** field.
* The **Name** field will correspond to the "DTM Data element" value listed in the table at the beginning of each Solution Section.
* The **Path** field will correspond to the "Data Layer Variable" value listed in the table at the beginning of each Solution Section.
* Several options are also listed for each Data Element you create (setting a default value, forcing lowercase, and giving the Data Element a custom expiration).
* Click **Save Data Element.**
* Repeat the process for each variable listed in the table at the beginning of this section.

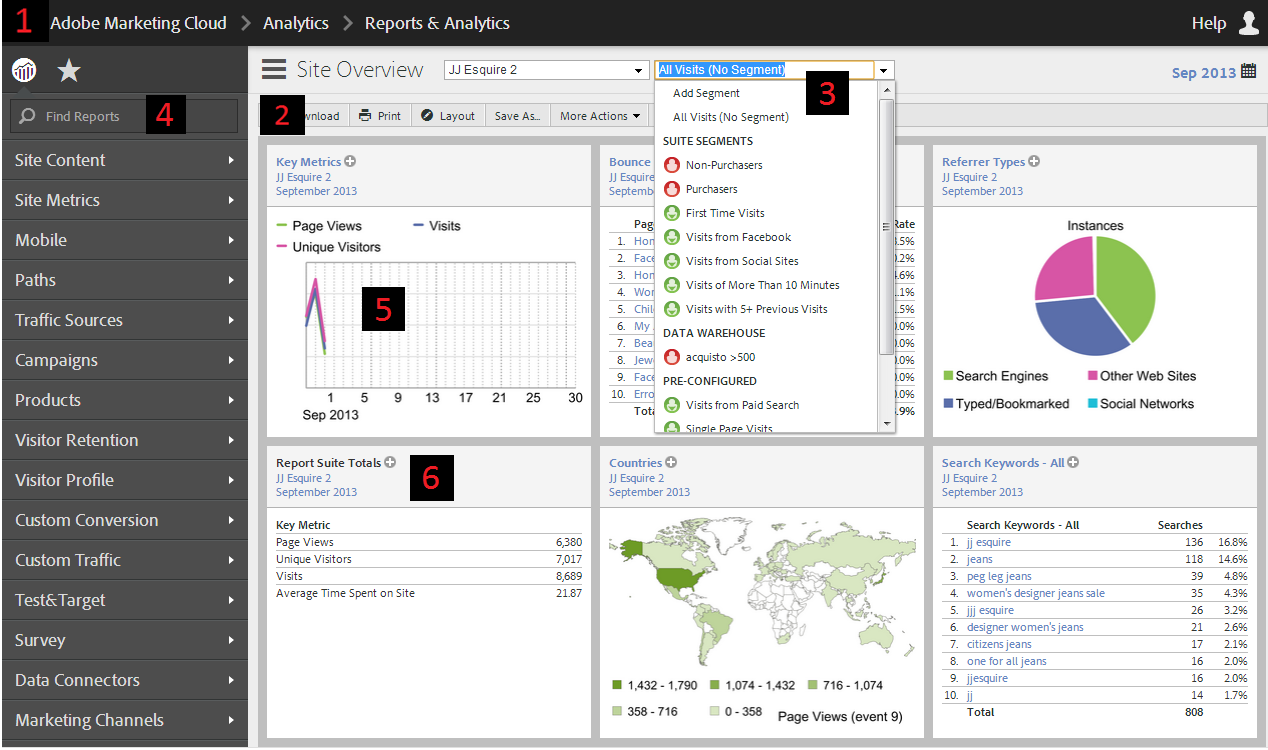
A completed data element mapped from the DTM data layer object might look like this:



# Appendix: Report Validation

## Adobe Analytics Interface

The Adobe Analytics interface, by default, provides quick access to the reports, metrics, and segments that you use most.



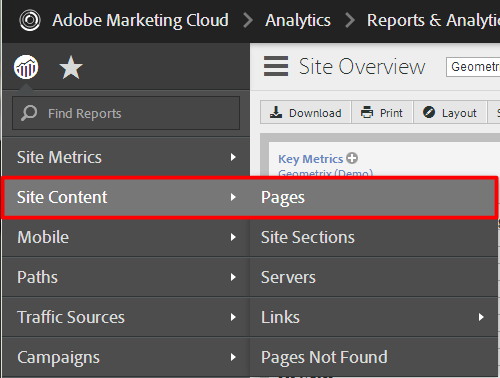
Via the Adobe Analytics home page, you can do the following:

1. Access other Adobe products (e.g. Target, Social, etc.) besides Analytics from the Adobe Experience Cloud menu.
2. Customize your landing page to either show a dashboard that you create from scratch or show the Site Overview dashboard, which Adobe Analytics generates automatically by default.
3. Apply segments to any report or dashboard reportlet. Available segments include those created by you, those created by your company colleagues, or the out-of-the-box segments included with Adobe Analytics.
4. Quickly locate reports, metrics, dashboards, and bookmarks using the left-hand menu's search field.
5. Turn full reports into thumbnail reportlets and add them to dashboards. You can apply report suite and date settings to individual reportlets or to all reportlets in a dashboard.
6. Share and distribute reports and dashboards, and use them to see quick and useful overviews of your site's performance.

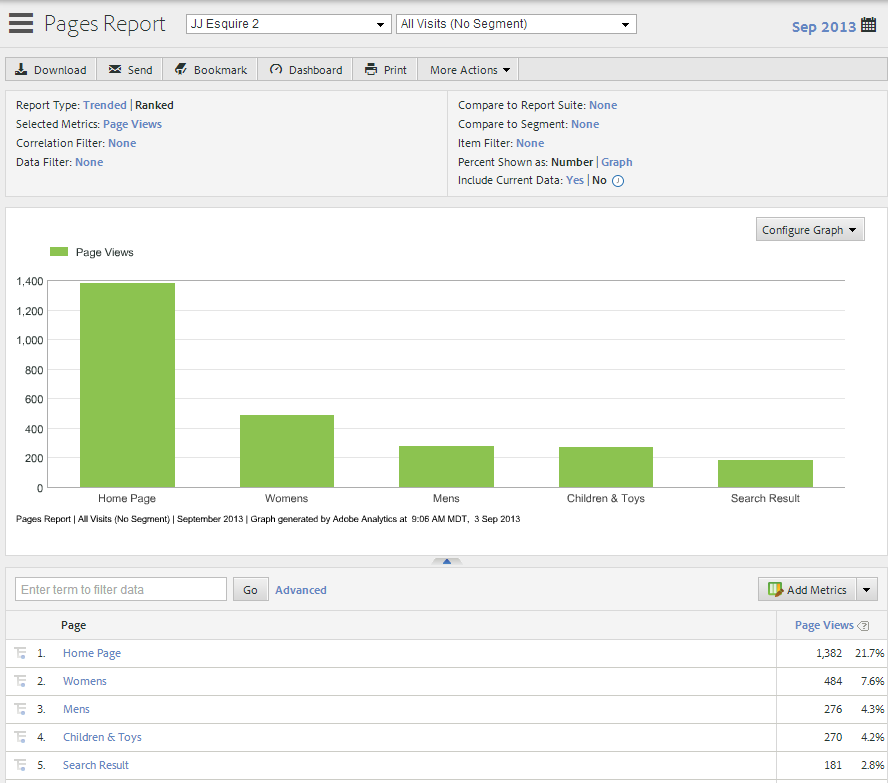
## Analytics Reports

In order to understand how to run and use Adobe Analytics reports, please go through this introductory tutorial by performing the following steps:

1. Log into Adobe Analytics
2. From the left-hand side menu, click on Site Content and then Pages



1. The interface will show the Pages report as seen in the following screenshot:

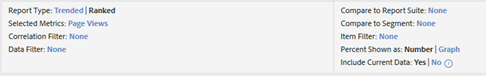


Although the configuration options may vary slightly across each report, with nearly every report you will be able to do the following:

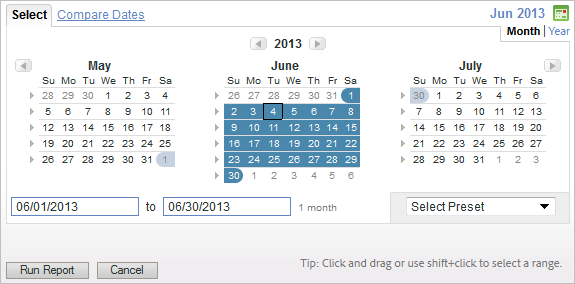
* Via the Toolbar, you may download, bookmark, and share with your colleagues the reports that you run.

http://microsite.omniture.com/t2/help/en_US/sc/user/graphics/toolbar.png

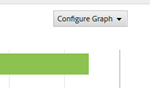
* The Settings allow you to configure the report by specifying the type of report to show, the metrics to add to the report details section, and other options. The available options in this group may change depending on the report you run.



* Clicking on the Calendar button will allow you to select the date range for your report. You can also select two time ranges to compare performance across different periods of time.



* The Configure Graph drop down box will allow you to select the type of graph or chart you want to see in conjunction with the report.

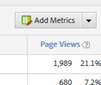


* If you wish to see no graph with your report, you may hide it by clicking on the up arrow placed just below the graph

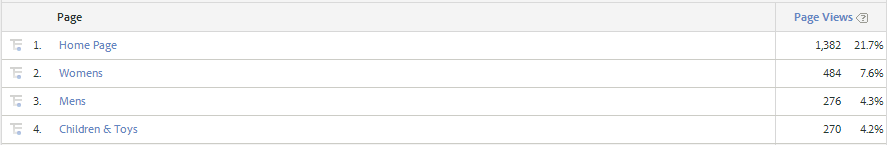


The Graph Display will change depending on the type of graph that you have selected. The default view typically shows the top items from the details list. If your graph is a trended graph showing different time periods, then data from each time period will appear instead.

* Via the Add Metrics button, you may add metrics to or remove metrics from your report. Users with administrative access can also specify the default metrics that all users will see for each individual report.



* The Details list contains the bulk of the report data. In this example, the top pages of the site are shown alongside the number of times each page was viewed.



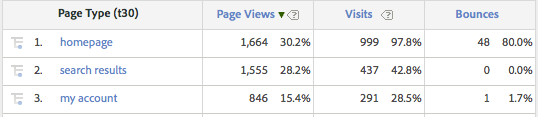
* If you have multiple metrics in a single report, you may sort the values by that metric by clicking on the metric's column header. For example, if you add Visits and Daily Unique Visitors to your pages report and then click on the Daily Visitors column header, you will see the site's top pages sorted by the largest number of visitors to each page.

## Validation Techniques by Report Type

#### Validating and Understanding Traffic Reports

Pages (s.pageName), Site Section (s.channel), and all Custom Traffic Variables (s.props) are considered "traffic variables". Their ideal default metric is "Page Views", but you can also bring in relevant out-of-the-box traffic metrics like Visits, Bounce Rate, Entry and Exit Rate, and Unique Visitors.

All props can be found in the reports under "Custom Traffic", which is divided into groups of 10 (prop 11 would be found under "Custom Traffic>Custom Traffic 1-10").



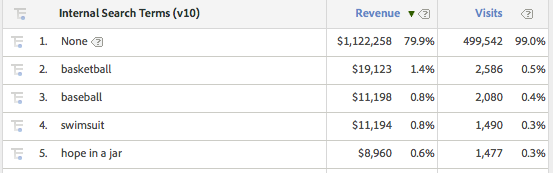
The Page Views metric (column) reflects the number of a page was viewed where this variable had the value seen in each row. For instance, in the example, 1664 page views happened where page type (prop30, in this case) was set to "homepage". The percentage tells me that the "homepage" value represents 30.2% of all pageviews on my site where prop30 was set.

The Visits metric (column) reflects the number of times the values in each row were viewed at least once during a visit to the site.

The "totals" at the bottom of the report reflect the number of times a page view happened where that variable is set. Some variables, like Site Section (s.channel)

#### Validating and Understanding Conversion Reports

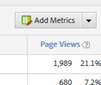
Campaign, Products, and all Custom Conversion Variables (s.eVars) are considered "Conversion Variables". They are not tied just to one page, but can persist between page views and even visits depending on their settings. All eVars can be found under "Custom Conversion".



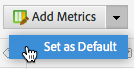
The above report tells me that 1.4% of my Revenue happened where eVar10 had been set to "basketball" at least once prior to the Order Confirmation page. 99% of visits are happening where eVar10 isn't set at all (which may just mean many users aren't using search and is not indicative of an implementation problem unless the variable is supposed to be set on every page/visit).

For eVars, the default base metric is based on your "Default Metrics" setting (In Adobe Analytics, see *Admin Console>Edit Report Suites>Individual Report Suite Settings>Default Metrics*).

* To add specific metrics (such as custom events), click the "Add Metrics" button.

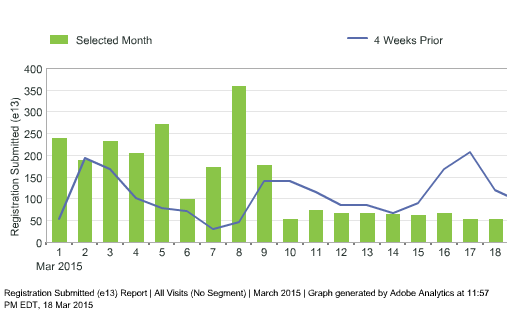


* + Common relevant out-of-the-box metrics to apply are Visits, Page Views, and Daily Unique Visitors.
* You can define Report-specific Default metrics by click the downward-pointing arrow to the right of "Add Metrics". Note that this affects all users of this report:



#### Validating and Understanding Event Reports

Purchase, cart additions, and custom s.events are all considered "Metrics". All custom events can be found under "Site Metrics > Custom Metrics". Event reports show a simple timeline of the number of times an event was set during that timeframe. If you use a standard date range (like a Calendar Month), additional lines in the report may represent previous date ranges, to give current data historical context:



|  |
| --- |
| NOTE |
| * Reports may produce differing numbers based on the metrics and date ranges applied. Be sure to verify the metrics and date ranges when viewing any report. |

For additional help with report validation, coordinate with your Adobe Consultant.