**METRICS OF ADOBE ANALYTICS**

Metrics are the foundation of reports and help you view and understand data relationships and enable side-by-side comparisons of different data sets about your website. Metrics are quantitative information about visitor activity, such as views, click-throughs, reloads, average time spent, units, orders, revenue, and so on.

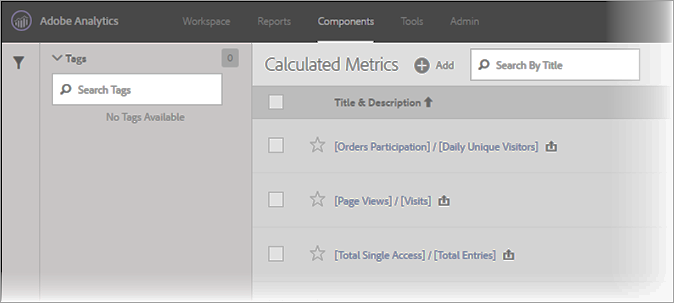
Metrics and associated data are displayed in the columns of reports. Traffic metrics show data about the volume of visitors. Conversion metrics show data about success events, such as purchases, downloads, or any other action that you want users to take on your website.

**Metric Types**

| **Metric Type** | **Definitions** |
| --- | --- |
| Traffic metrics | **Page View:** One Web page load in a user’s browser (one execution of the Reports & Analytics code).  **Visit:** Any number of page views when visitor comes to your site. A visit ends after 30 minutes of inactivity.  **Unique Visitor:** A person visiting your site for the first time during a given time frame, such as Hour, Day, Week, Month, Quarter or year. (This also includes Unique Visitors for any time frame.) |
| Conversion metrics | These show data about success events, such as purchases, downloads, or any other action that you want users to take on your website. |
| Video metrics | Analytics provides support for tracking a number of video metrics, including total views, time spent, and completion rates. |
| Social metrics | You can measure your brand's presence on the social web. Social metrics work with Analytics standard metrics. By combining these with calculated metrics, you can view a report that shows how often a product is mentioned, gauge product sentiment, and see how Social metrics correlate with Analytics key performance indicators. |

Calculated Metrics

Calculated metrics enable you to combine metrics to create mathematical operations that are used as new metrics. These metrics can be created for a report to which you add metrics. Administrators can create calculated metrics for all users of a report suite.



Tools

Here is a short overview of the Calculated Metrics tools:

| **Tool** | **Capabilities** |
| --- | --- |
| [Calculated Metric Builder](https://marketing.adobe.com/resources/help/en_US/analytics/calcmetrics/cm_build_metrics.html#concept_5EC82A91EB9C44FC870326C85F9D0B18) | * Create calculated and advanced calculated metrics. * Add segments inline to metric formulas. * Compare segments in the same report. For example, compare local visitors vs. international visitors. * Use statistical functions. * Provide detailed metric descriptions (show what it does, where to use it, where NOT to use it). * Copy definitions into new metrics. * Provide an inline metric preview. * Set metric polarity, which indicates whether it's good or bad if a given custom event (metric) goes up. * Tag metrics. |
| [Calculated Metric Manager](https://marketing.adobe.com/resources/help/en_US/analytics/calcmetrics/cm_manager.html#concept_BA6815CB06D842D5825766396B691653) | * Share metrics with others. * Approve and curate metrics. * Organize (tag) your metrics so people can find them. * Delete metrics. * Rename metrics. |
| Metric Selector rail | Replaces the **Show Metrics** popup in Reports & Analytics.  It lets you search for and add/apply metrics to the report. You can also change the[sort](https://marketing.adobe.com/resources/help/en_US/analytics/calcmetrics/cm_finding.html#concept_A09845053A934CB7B755391D76E76C08) order (options are: alphabetical, recommended, frequently used, recently used.) In addition, you can filter on Report Suites to show only metrics created in a specific report suite.  To access this Metric Selector, click the Metrics icon  to the left of a report. This is what the Metric Selector looks like: |
| [API for Calculated Metrics](https://marketing.adobe.com/developer/blog/new-calculated-metrics-and-the-apis) | Part of the Admin API set. |

Capabilities

You can

* Create metrics across Analysis Workspace, Reports & Analytics, Ad Hoc Analysis, Report Builder, Anomaly Detection, and Contribution Analysis.
* Create segmented metrics that are derived at report run time, [without having to change the implementation](http://youtu.be/CuQTm9RaUpY). These can be viewed historically because they are based on segments.
* Share metrics across report suites. This means that all newly created metrics apply to all reports suites in the same login company.
* (Advanced Calculated Metrics only) Segment on metrics. For example, you can create a metric for "New visitors", with a count of people for whom this is the first session.
* (Advanced Calculated Metrics only) Incorporate statistical functions to help you better describe your data. For example, you can count the number of items in a report or add in the number of standard deviations for each item.
* Utilize metrics created in Ad Hoc Analysis in the other Analytics tools and vice versa.

**Note:** You can continue to create metrics in Ad Hoc Analysis. Its calculated metric builder user interface is now similar to the new metric builder.

Limitations

Some Adobe Analytics features let you use events but not calculated metrics:

* Funnels in Reports & Analytics
* Fallout in Analysis Workspace
* Cohort Analysis in Analysis Workspace
* Data Warehouse
* Segments
* Real-Time reports
* Current Data reports
* Analytics for Target

SINGLE PAGE APPLICATION:

https://www.barclays.co.uk/

https://www.provincial.com/

https://www.commbank.com.au/banking.html?ei=mv\_banking

https://www.pnc.com/en/personal-banking.html

https://www.usbank.com/index.html

https://www.scotiabank.com/gls/en/index.html#personal-banking

## Features

Launch has gone mobile!!

Used by thousands of customers for web tag management, Adobe Launch can now be used to manage your mobile SDKs.

Creating a mobile property in Launch allows you to:

* Create data elements and build sophisticated rules that can combine actions across multiple solutions.
* Manage mobile extensions:
  + Mobile Core and Profile extensions are pre-installed with every mobile property.
  + Extensions are available for other Adobe solutions including Adobe Analytics, Adobe Target and Adobe Audience Manager.
* Mobile Core extension and all additional extensions can be downloaded and installed through a dependency managers such as Maven and CocoaPods.
* When you add an extension, you must recompile your app and complete the app store submission/approval process.

Data elements, rules, and extension configs are delivered to your application dynamically, are updated at app launch, and do not require app store updates.

For more information about getting started with mobile properties, see [Mobile](https://docs.adobelaunch.com/getting-started/mobile-1).

# Create and Deploy Configurations in Adobe Launch

1. Click **New Property**.
2. Create a new property and select **Mobile** as the platform.
3. Find the new property in the Properties list and click to open it.
4. Go to the **Extensions** tab.

Mobile Core and Profile extensions are installed by default.

1. Click **Catalog**, and install additional extensions needed.
2. Go to the **Data Elements** tab and add any data elements needed.
3. Go to the **Rules** tab and add any rules needed.
4. Go to the **Environments** tab.

Production, Staging, and Development environments have been added by default.

1. Click **Add Environment**, and add any additional development environments needed.

Additional Production and Staging environments cannot be added.

1. Go to the **Publishing** tab to publish the configuration.

This process involves creating a library of changes and then deploying the library:

a. Click on **Add New Library** under the **Development** section of the publishing workflow.

b. Specify any name for the library and select a development environment from the **Environment** dropdown.

c. Add the configuration changes to be deployed.

d. Click **Add All Changed Resources** (or to add only some changes, click **Add a Resource**).

e. Click **Save & Build for Development**.

The library will build and then show under the **Development** section of the publishing workflow.

1. Click on the down arrow for the library and select **Submit for Approval**.

The configuration contained in the library will then be deployed to the Development environment and the library will show under the Submitted section of the publishing workflow. Later, the library can be deployed to Staging and Production environments using the rest of the publishing workflow. For now, testing can be done using the configuration in the Development environment.

# Install the Adobe Cloud Platform SDKs in an Android or iOS App

1. Open the mobile property in Launch and go to the **Environments** tab to get the install instructions for adding the SDK to an app.
2. Find the environment needed in the table and click on the box icon under the **Install** column.
3. On the **Mobile Install Instructions** pop-up, choose **Android** or **iOS**.
4. Follow the instructions for using Grade with Android or CocoaPods with iOS. They necessary dependecy and initialization code can be copied from the pop-up to the app project.

# Use the Adobe Cloud Platform SDKs in an Android App

**Important:** This version of the Adobe Experience Cloud Platform SDKs supports **Android 4.0 (API 14) or later.**

The SDK configuration should be retrieved remotely from Launch:

1. Get the App ID from Adobe Launch.
2. Create MainActivity.java in the app.
3. Add the following line:

MobileCore.configureWithAppID("YOUR\_APP\_ID");

1. Launch the app and it will send a remote config request to the Adobe Launch servers and configure the app using the remote config.

# Use the Adobe Cloud Platform SDKs in an iOS App

**Important:** This version of the Adobe Cloud Platform SDKs supports **iOS 10 or later.**

The SDK configuration should be retrieved remotely from Launch:

1. Get the App ID from Adobe Launch.
2. Within your app, open AppDelegate.swift (or AppDelegate.m if developing in Objective-C).
3. Add this line in your didFinishLaunchingWithOptions method:

ACPCore.configure(withAppId: "YOUR\_APP\_ID") // swift

​

[ACPCore configureWithAppId:@"YOUR\_APP\_ID"]; // obj-c

1. Launch the app and it will send a remote config request to the Adobe Launch servers and configure the app using the remote config.

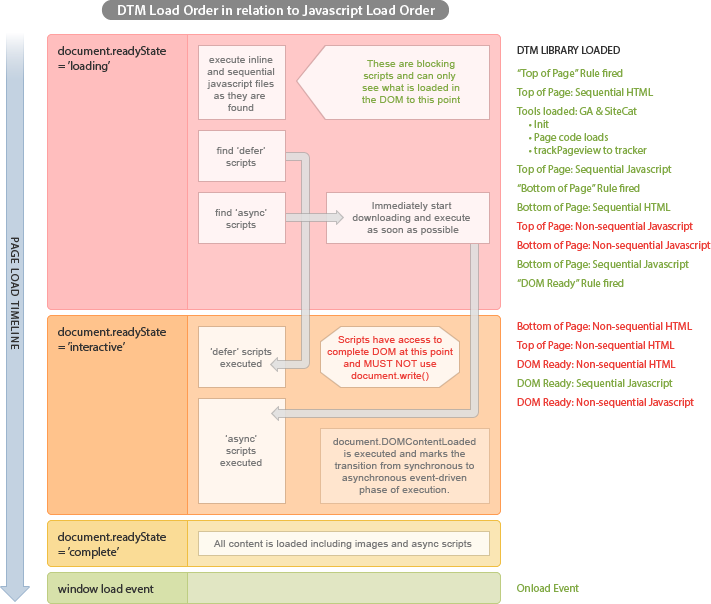
For more information about the mobile extensions, see [Mobile](https://docs.adobelaunch.com/extension-reference/mobile).

**09-10-2018**

# **Load Order for Rules**

The dynamic tag management load order in relation to the JavaScript load order.

The following diagram shows the general page load order.



**Note:** Red text on the above diagram represents non-sequential activity. Green text represents sequential activity.

The following section discusses how the different JavaScript/third-party tags script options are utilized within the four page-load phases:

* **Top of Page**
  + **Sequential HTML:**Injected into <HEAD/> below DTM library include script if <SCRIPT/> tags are used, otherwise is injected at top of <BODY/>
  + **Sequential JavaScript global:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Sequential JavaScript local:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Non-sequential JavaScript:**Injected as asynchronous <SCRIPT/> in <HEAD> below DTM library include script
  + **Non-sequential HTML:**Injected as hidden IFRAME and does not affect page HTML
* **Bottom of Page**
  + **Sequential HTML:**Injected after \_satellite.pageBottom() callback script with document.write() prior to DOMREADY so that there is no destruction of the visible page
  + **Sequential JavaScript global:**Injected after \_satellite.pageBottom() as JavaScript include <SCRIPT/>
  + **Sequential JavaScript local:**Injected after \_satellite.pageBottom() as JavaScript include <SCRIPT/>
  + **Non-sequential JavaScript:**Injected as asynchronous <SCRIPT/> in <HEAD> below DTM library include script
  + **Non-sequential HTML:**Injected as hidden IFRAME and does not affect page HTML
* **DOM Ready**
  + **Sequential HTML:**Will not work because DOMREADY is active and document.write() will overwrite the page
  + **Sequential JavaScript global:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Sequential JavaScript local:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Non-sequential JavaScript:**Injected as asynchronous <SCRIPT/> in <HEAD> below DTM library include script
  + **Non-sequential HTML:**Injected as hidden IFRAME and does not affect page HTML
* **Onload (window load)**
  + **Sequential HTML:**Will not work because DOMREADY is active and document.write() will overwrite the page
  + **Sequential JavaScript global:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Sequential JavaScript local:**Injected into <HEAD/> below DTM include script as JavaScript include <SCRIPT/>
  + **Non-sequential JavaScript:**Injected as asynchronous <SCRIPT/> in <HEAD> below DTM library include script
  + **Non-sequential HTML:**Injected as hidden IFRAME and does not affect page HTML

Non-sequential HTML third-party rules are always placed into iFrames. There is no such thing as asynchronous HTML so dynamic tag management refers to the iFrame functionality with that name.

**Note:** Even though scripts might be injected into the <HEAD/> later in the page processing like DOMREADY and ONLOAD, scripts will only appear in the proper timing sequence and be either global, local, synchronous, or asynchronous. Also, dynamic tag management checks to make sure that the timing is being handled properly and if DOMREADY has already occurred, it will not attempt a page-destructivedocument.write(). Dynamic tag management is diligent about not breaking the visual page if at all possible. For this reason, you might see an "asyc installation" error when the installation might not actually be "async." It is simply an alert that helps you recognize that the timing on the page is somewhat questionable and that DOMREADY has fired prior to all of the synchronous scripts loading as they should. If that happens, check to make sure that the pageBottom() callback is actually in the correct place immediately prior to the </body> tag. If not then these types of errors can occur.

The dynamic tag management queue processes them in the order listed above but does not distinguish between a JavaScript and an HTML script within the page load phase.

This means that the order of the page load rules in the queue determines the order in which they are de-queued.

A simple way to determine the queue order is to look at \_satellite.configurationSettings.pageLoadRules in the JavaScript console and examine the sequence.

For example, a complete view of the page load rule queue order can be seen by entering the following code into the JavaScript console:

\_satellite.each(\_satellite.configurationSettings.pageLoadRules,function(i){(i.event=='pagetop')?\_satellite.notify(i.event+': '+i.name,1):false})

\_satellite.each(\_satellite.configurationSettings.pageLoadRules,function(i){(i.event=='pagebottom')?\_satellite.notify(i.event+': '+i.name,1):false})

\_satellite.each(\_satellite.configurationSettings.pageLoadRules,function(i){(i.event=='domready')?\_satellite.notify(i.event+': '+i.name,1):false})

\_satellite.each(\_satellite.configurationSettings.pageLoadRules,function(i){(i.event=='windowload')?\_satellite.notify(i.event+': '+i.name,1):false})

**15-10-2018**

Dynamic Tag Manager provides you a great level of control in terms of how third party scripts can be added to your website.  Being able to control this can be very important as there are times where your vendor may require code to be placed strategically on your site in order for it to work correctly.

The problem with this flexibility is that it can be very confusing for the average user trying to choose which configurations to use.  The other problem is that your change could impact performance as you may end up adding to the load time or even worse; rendering the website unusable.

This article will help to explain the various settings that you will encounter, and provide examples of when to use each setting. I will also outline the common ‘gotchas’ associated with adding scripts.

# **Step 1: Define Trigger Rule**

Before we dive in, let’s take some time to understand the various trigger rules in which DTM executes the load rule. The following outlines the four conditions in firing order:

## Top of Page

“Top of Page” conditions will fire as soon as DTM is loaded. This will typically be fired as the <head> is rendering.  This is assuming that you have the “header code” installed as per the recommendation. At this point of the page lifecycle, there is no content rendered in the page such as links or images.

A very common need to load scripts here is when you need to load a script that loads beforecontent, as an example, you would use this rule to install “Optimizely” to avoid the page flicker resulting from the dynamic switching of content.

Exercise caution when installing HTML code such as AdWords conversion pixels: you should not render HTML elements such as <no script> or <div>, only tags that are allowed between <head> tags such as <meta>.

## Bottom of Page

Bottom of Page fires next when “\_satellite.pageBottom()” is called, so, it’s very important that the developer installs this right before the </body> end tag after all of the code.  Typically, when this trigger fires, you would be confident that all of the HTML has been loaded into the document.  Things like external files including images, JavaScript, and CSS is likely still loading in the background.

This is the common rule where users add their tags due to the content being ready for manipulation.

## Dom Ready

Dom ready is triggered when the HTML has finished loading, this guarantee that the loaded content is ready to be interacted with.  A practical use is when you are wanting to write jQuery logic to listen for interactions with your content.  You may notice that “Dom Ready” fires instantly after “Bottom of Page” simply due to the placement of the pageBottom() code.

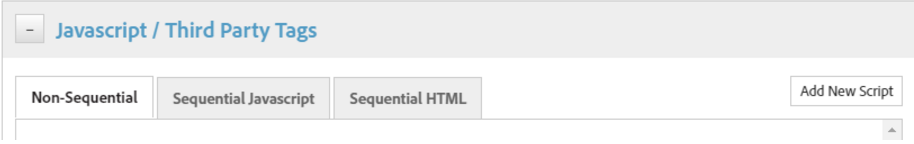
This is a safer balance to strike between speed and usability when it comes to adding your advertising pixels such as Adwords or Floodlight tags. This means that the scripts fire as soon as possible, while making sure the functionality of the website is not disrupted.

## Onload

Onload is triggered when both the HTML and dependent files include images and JavaScript.  This is the least favorite as it is the very last trigger to fire and normally you would like to see your script fire sooner.

# **Step 2: Define Script Type**

Once you define when the code is triggered, you will then need to define the type of Script to install.



## Sequential JavaScript

Sequential JavaScript is similar to inserting your custom code as an external <script> reference.

Setting this within a “Top of Page” condition will fire it sooner than if set in the “Non – Sequential” type.

Use Sequential only if you need the script to fire at a very specific time or sequence. Avoid any scripts that use the document.write function if using within “DOM Ready” or “Onload” conditions, this is due to the document already being rendered.

## Non – Sequential Javascript

Similar to Sequential JavaScript, with exception of being added asynchronously.  When setting this within a “Top of Page” condition, it will fire after any “Sequential Scripts”. From a speed perspective differences between “Sequential” and “Non sequential” should be rather limited.

Use non sequential when you’re not concerned exactly when it will execute. Best practices from a performance perspective are to default on sequential so that you don’t compete with the functionality of the website.

Avoid any scripts that use the document.write function if using within “DOM Ready” or “Onload” conditions, this is due to the document already being  rendered.

## Sequential HTML

DTM uses the document.write() method to add content to the document for sequential HTML scripts. A very common use for this type is the use of Adwords conversion tags when you have to mix HTML and JavaScript together.

Sequential HTML should be limited to only the “Bottom of Page” rule due to it being injected into the dom via document.write() the method.

This is commonly used for HTML remarketing HTML scripts such as Adwords or Floodlight tags but should always fire after your website scripts.  At times one might go back to a vendor and request a “JavaScript only” solution to avoid this limitation.

## Non – Sequential HTML

This comes across as my least favorite configuration option as DTM inserts this into your website via an externally referenced iFrame.

While this gets around the limitation of using the document.write() function (for marketing pixels), it limits your access to the global variables.  As an example, if you would not be able to populate fields in your vendor code such as Order ID or Product information.

Another trade off is when your vendor attempts to drop cookies, it’s actually dropping cookies on the wrong domain , and not the domain your visitor is viewing.

You’re better off using sequential HTML or advising your vendor to provide a JavaScript only version.

# **Considerations / Tips and Tricks**

* Each tag you add increases the network requests and therefore increase the time it takes for the page to fully load.
* Always balance the need to fire a script immediately versus loading it later when the page has completed loading. Consider how adding will impact the performance of the website.
* Never trust externally hosted scripts – if the server hosting the external script goes down – it can potentially make your website unusable depending on your configuration .
* Take the time to understand rendering blocking scripts and how to avoid using them.
* Use asynchronous JavaScript over synchronous methods to minimize usability impacts.
* When your vendor provides you code: ask if the code is asynchronous syntax or if one can be provided.

| **Metric Name** | **Description** |
| --- | --- |
| Average Page Depth | Displays on average how far within a visit each value was fired. This metric is valuable in determining how far within a visit your audience reaches a given page or prop value. Average Page Depth is available on any variable with pathing enabled. |
| Average Time Spent on Page | Represents the average time spent on a page within a visit. |
| Average Time Spent on Site | Represents the average time spent on a site within a visit. |
| Bounce rate | Shows the percentage of visits that contain a single hit. Bounce rate uses the Bounces metric and is calculated as: Bounces divided by Entries. |
| Bounces | A visit that consists of a single server call. For example, a single page visit is a bounce if a visitor does not interact with the page in a way that sends data to Adobe, such as clicking a link or a video start. If more than a single hit is received in a visit, a Bounce is not counted. |
| Campaign Click-throughs | Click-throughs represent the number of times that a tracking code for a given campaign was passed into reporting. When a visitor clicks on an affiliate link that has been tagged with one of these tracking codes, the visitor is taken to your landing page and the tracking code is captured in s.campaign. That data is sent into reporting and a click-through is recorded. |
| Cart Additions | The number of times an item was added to a shopping cart. This value comes from the scAdd event. |
| Cart Open | The number of times a customer opened a shopping cart by adding the first item. Occurs the first time an item is added to the shopping cart. This value comes from the scOpen event. |
| Cart Removals | The number of times an item was removed from a shopping cart. This value comes from the scRemove event. |
| Carts | The number of times a new shopping cart was opened or initialized. |
| Cart Views | The number of times the contents of the shopping cart are viewed by the customer. |
| Checkouts | An event that occurs when customers arrive at the checkout stage of a purchase. The checkout stage usually occurs just before a purchase is finalized, and usually involves the customer entering personal information (such as their shipping and billing information). You have control over the events on your site that qualify as checkouts. This value comes from the scCheckout event. |
| Click-throughs | Click-throughs represent the number of times that a tracking code for a given campaign was passed into reporting. When a visitor clicks on an affiliate link that has been tagged with one of these tracking codes, the visitor is taken to your landing page and the tracking code is captured in *s.campaign*. That data is sent into reporting and a click-through is recorded. |
| Customer (New, Return, Loyal) | Categories of the Customer Loyalty report:  New Customer: Customer with 0 purchases.  Return Customer: Customer with 1 purchase.  Loyal Customer: Customer with more than 1 purchase. |
| Daily Return Visits | Displays the number of visitors to your website more than once on a given day. A day is defined as the last 24-hour period. |
| Entries | Entries represents the number of times a given value is captured as the first value in a visit. Entries can occur only once per visit. However, it is not necessarily the first hit if the variable is not defined. |
| Exits | The number of times a given value is captured as the last value in a visit. Exits can occur only once per visit. |
| Instances | The number of times that a value was set for a variable. Instances are counted for all hit types, but are not counted when a value is recorded for a variable on a subsequent hit due to persistence. |
| Lifetime | The total amount of a given success metric for a single user. For example, the total number of lifetime visits for a user. |
| Mobile Views | The number of times a page is viewed or a dimension is set when accessed via a mobile device. Ad hoc analysis only. Instead of using the mobile views metric, we recommend applying the "Visits from Mobile Devices" segment. |
| New Engagements | New Engagements is a Marketing Channel reporting metric that counts new visitors that come as a result of a channel. This metric also counts visitors who have not been to your site in the last 30 days. A New Engagement is an eVar set at the beginning of each visit (original allocation). First-touch channels can also be New Engagements, depending on visitor engagement expiration setting. |
| Occurrences | The number of times a specific value is captured, plus the number of page views for which the given value persisted. In other words, Occurrences are the sum of page views and page events. Occurrences are available only in ad hoc analysis. |
| Orders | The number of orders made on your website during the selected time period. You can break down individual time periods by other metrics to show the items (such as products or campaigns) that contributed to the most orders during that time frame. |
| Page Depth | The average number of clicks it takes users to get to a certain page in the website. |
| Page Events | Page events consist of image request data from non-standard image requests. Sources of non-standard image requests are download links, exit links, and custom link tracking. |
| Page Views | A Page View is counted for each server call that is sent. This metric represents total instances of Page View. TrackLink calls are not counted as page views and do not increment the Page Views metric. |
| Path Views | The Path Views metric is based on pathing data, which is tracked for all users who accept persistent cookies.  The term Path View is used to indicate the number of times a page was viewed, given the constraints of the displayed path(s). This metric reports the number of page views for the given page that occurred within the selected path. This metric is available on the Paths report. Path Views shows you how many times a particular sequence of pages were viewed. |
| Product Views | Instance of the Product View being set. Occurs when the product detail page is viewed. This value comes from the prodView event. |
| Reloads | Counted when the same page name is loaded twice in a row. This typically indicates that the page was refreshed. Note that visiting the same page twice in the same visit does not count as a reload unless both visits occurred in-a-row. |
| Return Visits | Shows the number of visits where visit number is greater than 1. Return Visits includes non-cookied visitors. |
| Revenue | Revenue is captured on the purchase event, and is defined as the total dollar amount for the sum of the order for each product. This value comes from the purchase event. |
| Searches | Searches is not a default metric - it is always a custom metric.  It's the recommended default metric for search engines and keywords. This metric represents instances of a click-through, and shows the page that is associated with a specific engine or keyword. Searches metric data can be reported retroactively to the beginning of the data set. |
| Single Access | Single Access is defined by the number of visits to your site that contained a single unique Page Name value. If a user comes to your site and clicks a tracked link, triggers an event (such as a video view), or reloads the page, the visit is still considered a Single Access visit. As long as value for the pageName variable does not change, any number of requests can be sent and the visit is still considered a Single Access. |
| Time Spent | Metrics that report on the amount of time visitors spend on a page, site, or per visit. |
| Total | Total metrics report the value of all report line items for a reported period. If a filter is currently selected, the total might represent the filtered total instead of the report suite total. If no filter is selected total represents the report suite total. |
| Unique Customer | (Hourly, Daily, Weekly, Monthly, Quarterly, Yearly)  A Unique Customer is counted once for that time frame but cannot be counted again, no matter how many times the visitor returns to make a purchase. A Unique Visitor is counted once for the first visit in a specified period and not counted again until the period expires. After the period expires, the Unique Visitor is counted again. Unique Customers are always counted as Unique Visitors because they must visit the site in order to make the purchase. |
| Unique Visitors | Shows the total number of unique visitors for the reporting period (can be configured to daily, weekly, monthly, quarterly, yearly). |
| Units | The total units that were ordered for the selected time period. Because you have many units purchased per order, Units is a vital metric that reveals general inventory movement. |
| Visitors | The number of unique visitors to your site for a selected hour, day, week, month, quarter, or year. |
| Visits | A sequence of page views in a sitting. The visits metric is commonly used in reports that display the number of user sessions within the selected time period.  The visit metric is always associated with a time period, so you know whether to count a new visit if the same visitor returns to your site. |