



# Clickvoyant

Insights so fast, you'll think we're psychics

Comprehensive Tagging and Data Layer Guide

Google Analytics Universal and Google Analytics 4

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>Introduction</b>	<b>6</b>
About Google Tag Manager	6
What is the dataLayer?	7
<b>Install Google Tag Manager</b>	<b>8</b>
Add Container Code All Pages	8
<b>Install Google Optimize</b>	<b>9</b>
Setup Google Optimize in Google Tag Manager	9
The anti-flicker snippet code	11
The anti-flicker code sequence	12
<b>Data Layer Initialization</b>	<b>13</b>
Dynamic Parameters	13
<b>Custom Events</b>	<b>14</b>
Global Components	14
Navigation Clicks	14
Event Naming Convention for GA Universal	14
Event Naming Convention for GA4	14
Trigger	15
Call-to-Action Clicks	15
Event Naming Convention for GA Universal	15
Event Naming Convention for GA4	16
Trigger	16
Social Media Actions	16
Event Naming Convention for GA Universal	17
Event Naming Convention for GA4	17
Trigger	17
Non-Tagged Links	17
Event Naming Convention for GA Universal	18
Event Naming Convention for GA4	18
Trigger	18
Chat Bots	20
Opens	20
Event Naming Convention for GA Universal	20

Event Naming Convention for GA4	20
Trigger	20
Closes	20
Event Naming Convention for GA Universal	21
Event Naming Convention for GA4	21
Trigger	21
Content Accordions	23
Event Naming Convention for GA Universal	24
Event Naming Convention for GA4	24
Trigger	25
Click-To-Calls	26
Event Naming Convention for GA Universal	26
Event Naming Convention for GA4	26
Trigger	26
Modals, PopUps, and Lightboxes	28
Event Naming Convention for GA Universal	28
Event Naming Convention for GA4	28
Trigger	28
Search	29
Search Results	30
Event Naming Convention for GA Universal	30
Event Naming Convention for GA4	30
Trigger	30
Search Filter	31
Event Naming Convention for GA Universal	31
Event Naming Convention for GA4	31
Trigger	32
Forms	33
Submissions	33
Event Naming Convention for GA Universal	33
Event Naming Convention for GA4	33
Trigger	33
Field Input	34
Event Naming Convention for GA Universal	34
Event Naming Convention for GA4	34
Trigger	35
Multipage Form Usage	35
Event Naming Convention for GA Universal	36
Event Naming Convention for GA4	36

Trigger	36
User Variables	<b>38</b>
Registration	38
Event Naming Convention for GA Universal	38
Event Naming Convention for GA4	38
Trigger	38
Logout	38
Event Naming Convention for GA Universal	39
Event Naming Convention for GA4	39
Trigger	40
Login	40
Event Naming Convention for GA Universal	40
Event Naming Convention for GA4	41
Trigger	41
Cookie Permissions	41
Event Naming Convention for GA Universal	42
Event Naming Convention for GA4	42
Trigger	42
Ecommerce	<b>43</b>
Product Tracking	43
Product Impressions	43
Event Naming Convention for GA Universal	44
Event Naming Convention for GA4	44
Trigger	44
Product Clicks	45
Event Naming Convention for GA Universal	46
Event Naming Convention for GA4	46
Trigger	46
Product Detail Views	47
Event Naming Convention for GA Universal	47
Event Naming Convention for GA4	47
Trigger	48
Cart Tracking	48
Add to cart	48
Event Naming Convention for GA Universal	49
Event Naming Convention for GA4	49
Trigger	49
Remove from the cart	50

Event Naming Convention for GA Universal	51
Event Naming Convention for GA4	51
Trigger	51
Adding a discount	51
Event Naming Convention for GA Universal	52
Event Naming Convention for GA4	52
Trigger	52
Checkout Tracking	53
Event Naming Convention for GA Universal	54
Event Naming Convention for GA4	54
Trigger	54
Checkout Options	55
Event Naming Convention for GA Universal	55
Event Naming Convention for GA4	55
Trigger	55
Purchase Tracking	56
Event Naming Convention for GA Universal	57
Event Naming Convention for GA4	57
Trigger	57
Purchasing with a discount	58
Event Naming Convention for GA Universal	58
Event Naming Convention for GA4	59
Trigger	59
Refund Tracking	60
Event Naming Convention for GA Universal	60
Event Naming Convention for GA4	61
Trigger	61

# Introduction

The amount of data you can collect has gotten so crazy, that businesses can spend more than \$50K implementing it. With all the budget gone, what budget is left to analyze it?

We're providing this free Google Analytics implementation guide to help you get a leg up and save budget for the actual insights. Follow it step by step to get all the data you need to analyze like a Fortune 500 enterprise.

But first, some critical concepts...

## About Google Tag Manager

Google Tag Manager (GTM) is a free tag management system that we use exclusively to implement data collection tags. The concept of a GTM is to separate the tags from the core website source. This separation frees the developer from tagging, and instead, the analytics person can manage the tags she needs for insights.

GTM works like this:

1. The website pushes key data into a data layer (for example, a user ID, or product attribute)
2. Any tags implemented within GTM can then utilize the data within the data layer – to populate analytics reports.
3. Interaction events (like element clicks, form submits, etc) can fire tags

GTM uses an event-driven model that we can utilize to track interactions as well. Below are examples for tracking AJAX requests and element clicks using jQuery

### Example Chat Button Click

```
<script>
var dataLayer = dataLayer || [];
$('.chatBtn').click(function (e) {
dataLayer.push({
'event': 'chat-click' }); }); </script>
```

### Example Login Tracking

```
<script>
var dataLayer = dataLayer || [];
```

```
$.post( "login", function (data) {
  dataLayer.push({
    'event': 'login-success', 'userID': data.userID }); }); </script>
```

## What is the dataLayer?

A data layer is a layer of your website code which contains all of the data that gets generated by visitors engaging with your website. Not all websites have a data layer, but having one enables more reliable and flexible data collection.

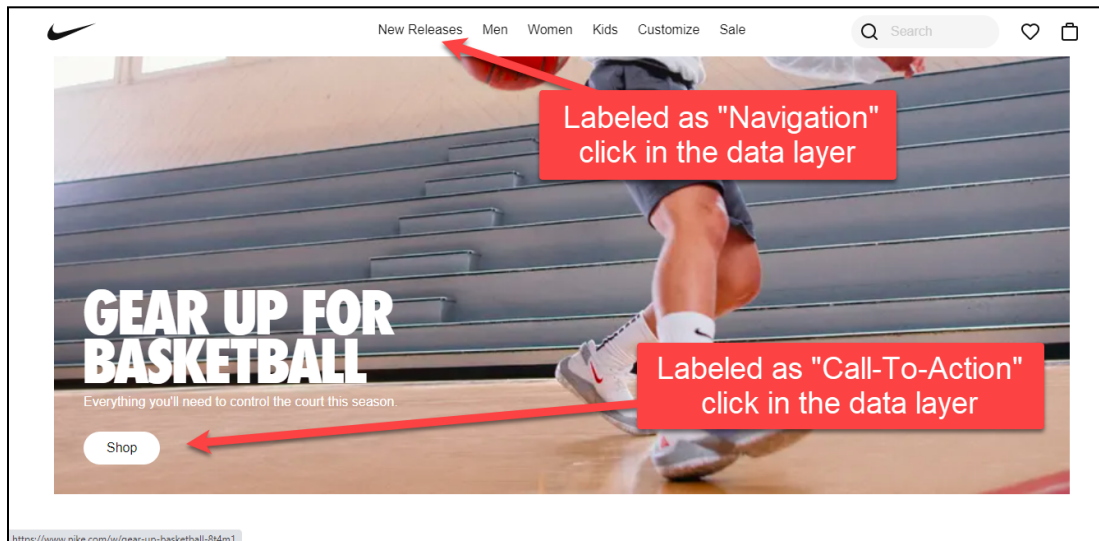
### Benefits of the Data Layer:

- When you update the site, the data is automatically updated too.
- Reduces tracking errors because it requires less management of GTM.
- Can be used by any marketing technology, facebook, linkedin, google analytics, etc.

### How the Data Layer Works

- Instead of tracking clicks on a link that says "New Releases" or a button that says "Shop" (see example below), the data layer lets you track the type of link or button that is clicked.
- Elements are labeled in the data layer based on their tracking category--not on things like button text or URL, which will likely change with site updates.

### EXAMPLE: Tracking Navigation & Call-To-Action Clicks



In the example above, changing the button from "Shop" to "Buy Now" would not disrupt tracking. The event is triggered when a user clicks a button labeled as a "Call-To-Action", not when the user clicks a button that says "Shop".

# Install Google Tag Manager

## Add Container Code All Pages

To install the GTM container, code must be added to every page. This container code has two snippets of code: **<script> tag** and **<iframe> tag** . The **<script> tag** must be rendered as high up as possible in the **<head>** section of the page and the **<iframe> tag** must be placed as close to the opening **<body>** tag as possible. Neither snippet should not be nested within any other element tags. The GTM container ID must be inserted where the example snippet contains the text **GTM-XXXXX** .

**IMPORTANT NOTE:** Many third party integrations are available that automatically add Google Tag Manager container code to all site pages. Do NOT duplicate container code on pages if you are already using a third party installation resource.

### Deployment Example **<script> tag:**

```
<head>

<!-- Google Tag Manager --> <script>(function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gtm.start': new
Date().getTime(),event:'gtm.js'});var f=d.getElementsByTagName(s)[0],
j=d.createElement(s),dl=l!='dataLayer'?'&l='+l:'';j.async=true;j.src=
'https://www.googletagmanager.com/gtm.js?id='+i+dl;f.parentNode.insertBefore(j,f);
})(window,document,'script','dataLayer','GTM-XXXXX ');</script> <!-- End Google Tag Manager -->

</head>
```

### **<iframe> tag:**

```
<body>

<!-- Google Tag Manager (noscript) --> <noscript><iframe
src="https://www.googletagmanager.com/ns.html?id=GTM-XXXXX " height="0" width="0"
style="display:none;visibility:hidden"></iframe></noscript> <!-- End Google Tag Manager (noscript) -->

</body>
```



# Install Google Optimize

Google Optimize allows users to conduct ‘experiments’ that test out different variants of web pages by seeing how they perform against a goal that you specify. Before we can install Optimize through Google Tag Manager, the following items must be present and/or completed:



- Accounts created for: Google Analytics, Google Tag Manager, and Google Optimize
- Google Tag Manager container installed on your website
- Google Analytics property installed on your website (through GTM)
- Google Optimize account and container linked to your Google Analytics property

## Setup Google Optimize in Google Tag Manager


1. Sign in to Google Tag Manager and open your container.
2. Click Tags > New.
3. Click Tag Configuration > Google Optimize.
4. Enter your Optimize container ID, which you can find under Container details in the container settings in Optimize.
5. Save the tag without triggers. Note: The pageview trigger must be configured in your Google Analytics: Universal Analytics configuration tag, which will fire the Optimize tag (see instructions in next step).

### Tag Configuration

Tag Type

 **Google Optimize**  
Google Marketing Platform 


Optimize Container ID



> More settings

> Advanced Settings

### Triggering

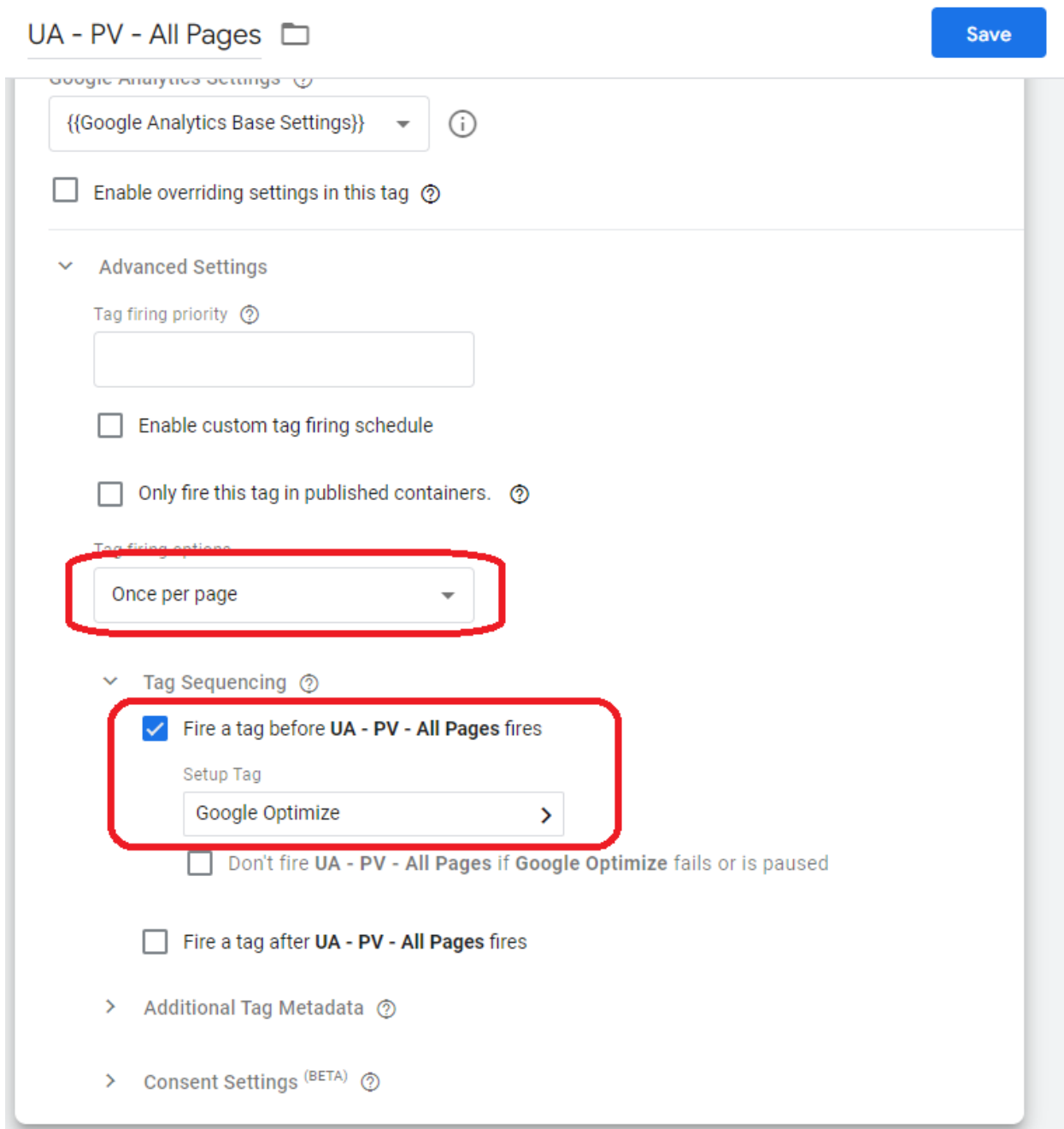


Choose a trigger to make this tag fire...

[Learn More](#)

6. Enable tag sequencing:

- Open your Google Analytics: Universal Analytics configuration tag for the Google Analytics property that is linked to your Optimize container (our example tag in the screenshot below is called “UA - PV - All Pages”)
- Click Tag Configuration > Advanced Settings > Tag Sequencing
- Check the box to fire a tag before this tag fires. Under the Setup Tag heading, click the menu and select the desired Optimize tag.
- Configure the Optimize tag to fire once per page, then save it.



7. Publish your Google Tag Manager container to complete the Optimize installation.

### The anti-flicker snippet code

The **anti-flicker snippet** includes a settable timeout functionality that stops trying to apply the Optimize experiment changes if too much time has passed.

To add the anti-flicker snippet to your site, insert the following code just after the <head> section of every \*page prior to any other code on the page. \*

```
<style>.async-hide { opacity: 0 !important} </style>
<script>(function(a,s,y,n,c,h,i,d,e){s.className+=' '+y;h.start=1*new Date;
h.end=i=function(){s.className=s.className.replace(RegExp(' ?'+y),'')};
(a[n]=a[n]||[]).hide=h;setTimeout(function(){i();h.end=null},c);h.timeout=c;
})(window,document.documentElement,'async-hide','dataLayer',3000,
{ 'CONTAINER_ID':true});</script>
```

**NOTE: If you initialize Optimize from the GTM Container, the CONTAINER\_ID should be the GTM ContainerID**

### **The anti-flicker code sequence**

When the anti-flicker snippet is on the site, the script sequence should be:

1. datalayer
2. Anti-flicker snippet
3. GTM container code

# Data Layer Initialization

As the page loads, push the following to the dataLayer to set the pageType, which will make it available to Google Tag Manager. Any of these fields that are not applicable to the page can be omitted. Place this code above the first GTM container snippet in the <head> of the page.

```
<script>  
window.dataLayer = window.dataLayer || []; window.dataLayer.push({  
  'pageType': '[PAGE TYPE]'  
}); </script>
```

## Dynamic Parameters

1. **[PAGE TYPE]** : This identifies the content on the page. e.g. “Home”, “Category Page”, “Search Results”, “Product Page”, “Blog”...

# Custom Events

## Global Components

### Navigation Clicks

Whenever a user clicks a menu item to navigate the site, push the **ce\_navigationClick** event into the dataLayer, along with the **menu type** and the **menu item** name.

```
example: <script>
dataLayer.push({
'event': 'ce_navigationClick', 'menuType': '[MENU TYPE]', 'menuItemName': '[MENU ITEM NAME]'});
</script>
```

Dynamic Parameters:

[MENU TYPE]: header, footer, left nav menu, right nav menu ...

[MENU ITEM NAME]: The text of the menu item that was clicked.

### Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
navigation	{{DL - menuType}}	{{DL - menuItemName}}

### Event Naming Convention for GA4

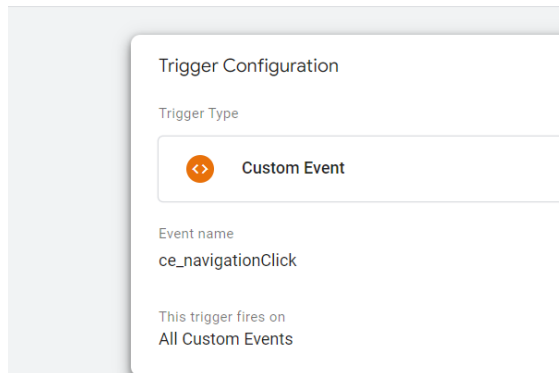
Event name: navigation\_click

<i>Parameter name</i>	<i>Value</i>
event_cat	navigation
menu_type	{{DL - menuType}}
menu_item_name	{{DL - menuItemName}}

## Trigger

CE - ce\_navigationClick Custom Event: Event name = ce\_navigationClick | All Custom Events

CE - ce\_navigationClick 



### Call-to-Action Clicks

Any buttons on the page that is meant to drive users to an action should be tracked as Calls To Action, please add the following data attributes to the `<a>` or `<button>` tags:

- `data-click-category`: 'call to action'
- `data-click-action`: '[CLICK TEXT]'
- `data-click-label`: '[CLICK URL]'

example:

```
<a href="/about-us.html" class="btn btn-secondary text-uppercase margin-right-small "aria-label="Watch Company" data-click-category="call to action" data-click-action="Read More" data-click-label="/about-us.html"> Read more </a>
```

In cases where there are multiple CTA links with the same text and/or URL on a single page, you may add more information to the **data-click-action** attribute to differentiate them. See the following example for the screen-captured “Read More” CTA:

example:

```
<a href="/about-us.html" class="btn btn-secondary text-uppercase margin-right-small "aria-label="Adtalem Global Education" data-click-category="call to action" data-click-action="Read More - Button 3" data-click-label="/about-us.html"> Read more </a>
```

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
{{AEV - Attribute - Click Category}}	{{AEV - Attribute - Click Action}}	{{AEV - Attribute - Click Label}}

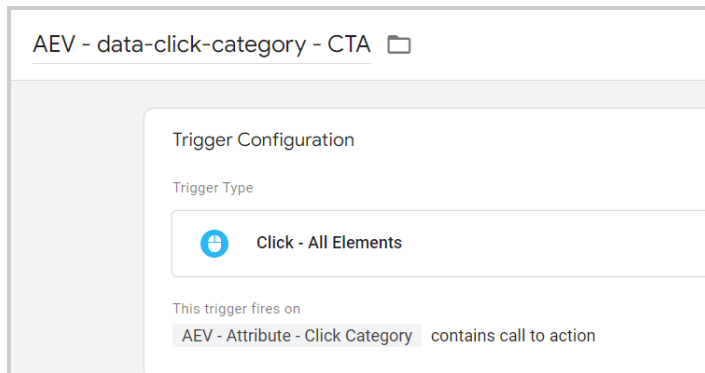
## Event Naming Convention for GA4

Event name: call\_to\_action

<i>Parameter name</i>	<i>Value</i>
event_cat	call to action
click_text	{{AEV - Attribute - Click Action}}
click_URL	{{AEV - Attribute - Click Label}}

## Trigger

AEV - data-click-category - CTA Click - All Elements: AEV - Attribute - Click Category | contains | call to action



## Social Media Actions

For any components that contain a social media action (share or link) please add the following data attribute to the <a> or <button> tags

- **data-click-category:** 'social'
- **data-click-action:** ['share' or 'link']
- **data-click-label:**[SOCIAL MEDIA PROPERTY]

Example:

```
<a href="https://www.facebook.com/mybiz" class="icon-link" aria-label="Facebook">
```



```

data-click-category="social" data-click-action="link" data-click-label="facebook" rel="nofollow"
target="_blank" > <span class="visually-hidden">Facebook</span> <span class="icon
icon--facebook"></span> </a>

```

## Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
{{AEV - Attribute - Click Category}}	{{AEV - Attribute - Click Action}}	{{AEV - Attribute - Click Label}}

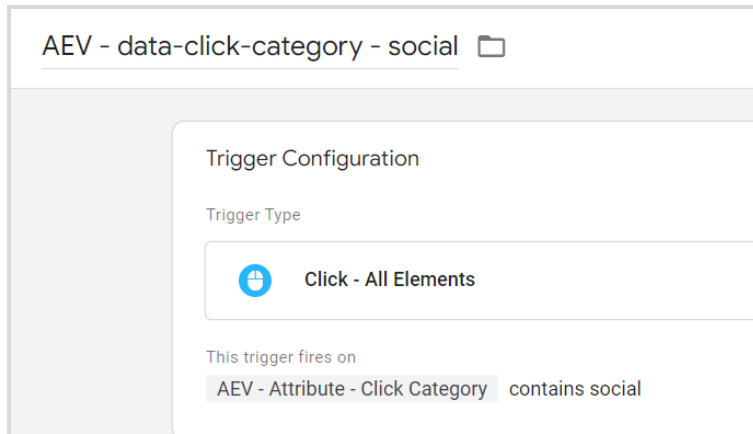
## Event Naming Convention for GA4

Event name: social\_interact

<i>Parameter name</i>	<i>Value</i>
event_cat	{{AEV - Attribute - Click Category}}
social_action	{{AEV - Attribute - Click Action}}
social_platform	{{AEV - Attribute - Click Label}}

## Trigger

AEV - data-click-category - social Click - All Events: AEV - Attribute - Click Category | contains | social



## Non-Tagged Links

This event will use the GTM container auto event variables to collect any link which may not fall under CTA, Social, or Navigation.

## User-defined variable required in GTM:

AEV - Attribute - Click Category 🗨

The screenshot shows the 'Variable Configuration' dialog for a user-defined variable. The 'Variable Type' is set to 'Auto-Event Variable'. Below this, the 'Variable Type' is set to 'Element Attribute' and the 'Attribute name' is 'data-click-category'.

## Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>non-tagged link</i>	<i>link</i>	<i>[CLICK TEXT]   [CLICK URL]</i>

## Event Naming Convention for GA4

Event name: non\_tagged\_link

<i>Parameter name</i>	<i>Value</i>
event_cat	non-tagged link
click_text	{{Click Text}}
click_URL	{{Click URL}}

## Trigger

AEV - Attribute - Click Category | matches RegEx (ignore case) | ^(undefined|null|0|false|NaN)\$

## CL - Non Tagged Link Click

### Trigger Configuration

Trigger Type



Click - Just Links

This trigger fires on

AEV - Attribute - Click Category matches RegEx (ignore case) `^(undefined|null|0|false|NaN)$`

# Chat Bots

## Opens

Each time a chat window is opened, push the ce\_chatOpen event to the dataLayer, along with the chat type.

```
example:  
<script>  
dataLayer.push({  
'event': 'ce_chatOpen', 'chatType': '[CHAT TYPE]'}); </script>
```

Dynamic Parameters:  
[CHAT TYPE]:

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
chat	open	{{DL - chatType}}

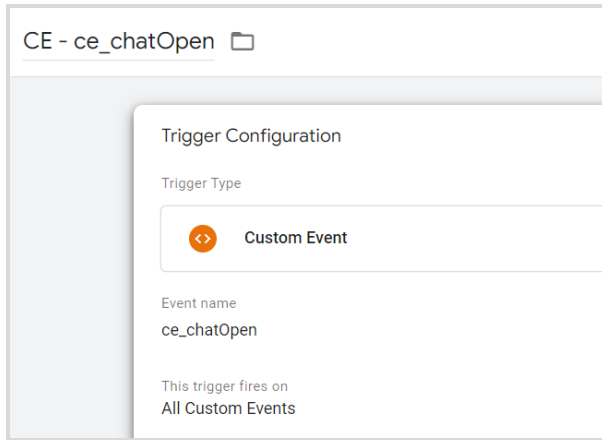
### Event Naming Convention for GA4

Event name: chat\_open

<i>Parameter name</i>	<i>Value</i>
event_cat	chat
chat_type	{{DL - chatType}}

### Trigger

CE - ce\_chatOpen Custom Event: Event name = ce\_chatOpen | All Custom Events



## Closes

Each time a chat window is opened, push the ce\_chatClose event to the dataLayer, along with the chat type.

**example:** `<script>  
dataLayer.push({  
'event': 'ce_chatClose', 'chatType': '[CHAT TYPE]'}); </script>`

Dynamic Parameters:  
[CHAT TYPE]:

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
chat	close	{{DL - chatType}}

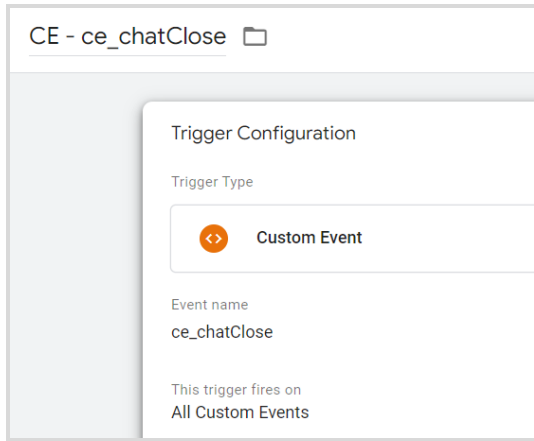
### Event Naming Convention for GA4

Event name: chat\_close

<i>Parameter name</i>	<i>Value</i>
event_cat	chat
chat_type	{{DL - chatType}}

# Trigger

CE - ce\_chatClose Custom Event: Event name = ce\_chatClose | All Custom Events



## Content Accordions

Each time an accordion is opened, push the **ce\_expandList** event into the datalayer, along with the **list name** and **list item**.

Example:

```
dataLayer.push({  
  'event' : 'ce_expandList',  
  'listName' : [block_title], // Set to title of block  
  'listItem' : [accordion_item_text], // Set to text of accordion item  
});
```

Dynamic Parameters:

[LIST NAME]: Title of the block in the example this would be the value of the h3 in the div.sf-list-item

[LIST ITEM]: Value of the accordion item text, in the example this would be the text in i.text-muted sf-icon

Visual Example:

**School Info** | **5 Programs**

**Overview** +

**Performance** -

- 83% of students **graduate** in four years
- 57% of students **enroll in college or career programs**
- 86% student **attendance**
- 82% of students **feel safe** in the hallways, bathrooms, locker room, and cafeteria
- 93% of students feel that this school offers a **wide enough variety of programs, classes, and activities** to keep them interested in school

Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
expandable list	{{DL - listName}}	{{DL - listItem}}

Event Naming Convention for GA4

Event name: expand\_list

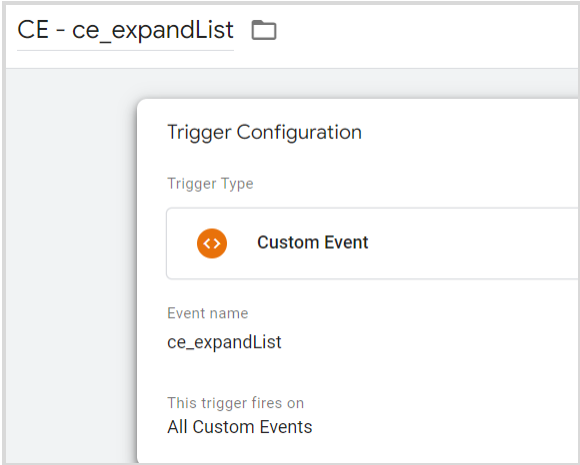
<i>Parameter name</i>	<i>Value</i>
event_cat	expandable list
list_name	{{DL - listName}}



list_item	{{DL - listItem}}
-----------	-------------------

### Trigger

CE - ce\_expandList Custom Event: Event name = ce\_expandList | All Custom Events



## Click-To-Calls

For any buttons on the page that trigger a phone call, please add the following data attributes to the <a> or <button> element:

- data-click-category: 'phone'
- data-click-action: '[CLICK TEXT]'
- data-click-label: '[PHONE NUMBER]'

example:

```
<a href="https://api.whatsapp.com/send?phone=13122002000" target="_blank" data-click-category="phone" data-click-action="WhatsApp (312) 200-2000" data-click-label="13122002000"> WhatsApp (1) 312-200-2000 </a>
```

example:

```
<a href="tel:1-555-555-5555" data-click-category="phone" data-click-action="Call Now!" data-click-label="1-555-555-5555"> Call Now! </a>
```

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
{{AEV - Attribute - Click Category}}	{{AEV - Attribute - Click Action}}	{{AEV - Attribute - Click Label}}

### Event Naming Convention for GA4

Event name: click\_to\_call

<i>Parameter name</i>	<i>Value</i>
event_cat	phone
click_text	{{AEV - Attribute - Click Action}}
click_URL	{{AEV - Attribute - Click Label}}

### Trigger

AEV - data-click-category - phone Click - All Elements: AEV - Attribute - Click Category | contains | phone

### Trigger Configuration

Trigger Type



Click - All Elements

This trigger fires on

AEV - Attribute - Click Category contains phone

### References to this Trigger



UA - Event - Click-to-Calls  
Tag

## Modals, PopUps, and Lightboxes

Any modal views or clicks should be tracked from the CSS selectors. Please add the **data-modal-id** and **data-modal-action** attributes to the <div> and <button> tags when applicable.

- **data-modal-id:** 'modal: [modal name]'
- **data-modal-action:** '[action type]'

Example:

```
<div class="my-modal" data-modal-id="modal: glossary">  
<button data-modal-action="close" class="modal-close"></button>  
<h1>Header here</h1>  
<p>  
Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
</p>  
<br>  
</div>
```

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>modal</i>	{{AEV - Attribute - Data Modal ID}}	{{AEV - Attribute - Data Modal Action}}


### Event Naming Convention for GA4

Event name: modal\_interact

<i>Parameter name</i>	<i>Value</i>
event_cat	modal
modal_id	{{AEV - Attribute - Data Modal ID}}
modal_action	{{AEV - Attribute - Data Modal Action}}

## Trigger


AEV - data-modal-id Click - All Elements: AEV - Attribute - Data Modal ID does not match RegEx (ignore case)  $^{\wedge}(\text{undefined}|\text{null}|0|\text{false}|\text{NaN})\text{\$}$

AEV - data-modal-id 

---

Trigger Configuration

Trigger Type

 Click - All Elements

This trigger fires on

AEV - Attribute - Data Modal ID does not match RegEx (ignore case)  $^{\wedge}(\text{undefined}|\text{null}|0|\text{false}|\text{NaN})\text{\$}$

# Search

## Search Results

On the search results page, push the **ce\_searchResults** event into the dataLayer, along with the search term and the number of results.

```
example: <script>
dataLayer.push({
'event': 'ce_searchResults', 'searchTerm': '[SEARCH TERM]', 'results': '[NUMBER OF RESULTS]' });
</script>
```

Dynamic Parameters:

[SEARCH TERM]: The search term that the user input to the search box.

[NUMBER OF RESULTS]: The number of search results returned by the search.

## Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
search	results returned	{{DL - searchTerm}} ~ Results: {{DL - results}}

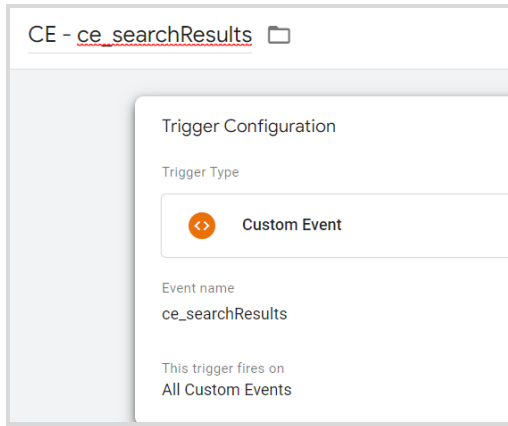
## Event Naming Convention for GA4

Event name: view\_search\_results

<i>Parameter name</i>	<i>Value</i>
event_cat	search
search_term	{{DL - searchTerm}}
no_of_results	{{DL - results}}

## Trigger

CE - ce\_searchResults Custom Event: Event name = ce\_searchResults | All Custom Events



## Search Filter

On selection of a filter or faceted navigation, push the **ce\_searchFilter** event into the dataLayer, along with the filter text and number of results.

### Example

```
<script>
  dataLayer.push({
    'event': 'ce_searchFilter',
    'searchTerm': '[SEARCH FILTER]',
    'results': '[NUMBER OF RESULTS]'
  });
</script>
```

### Dynamic Parameters

[SEARCH FILTER]: The filter term that the user clicked on.

[NUMBER OF RESULTS]: The number of search results returned by the search.

## Event Naming Convention for GA Universal

Category	Action	Label
search	filter	{{DL - searchTerm}} ~ Results: {{DL - results}}

## Event Naming Convention for GA4

Event name: search\_filter

Parameter name	Value
----------------	-------

event_cat	search
search_term	{{DL - searchTerm}}
no_of_results	{{DL - results}}

## Trigger

CE - ce\_searchFilter Custom Event: Event name = ce\_searchFilter | All Custom Events

The screenshot shows a 'Trigger Configuration' dialog box. At the top, it says 'Trigger Type' with a selected 'Custom Event' option. Below that, the 'Event name' field contains 'ce\_searchFilter' and there is an unchecked checkbox for 'Use regex matching'. At the bottom, under 'This trigger fires on', the 'All Custom Events' radio button is selected, and the 'Some Custom Events' radio button is unselected.



# Forms

## Submissions

When a form is submitted, push the **ce\_formSubmit** event into the dataLayer along with the **form name**.

**example:** `<script>  
dataLayer.push({  
'event': 'ce_formSubmit', 'formName': '[FORM NAME]'}); </script>`

Dynamic Parameters:

[FORM NAME]: The unique name that identifies the form

## Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
<i>form</i>	<i>submitted</i>	<i>{{DL - formName}}</i>

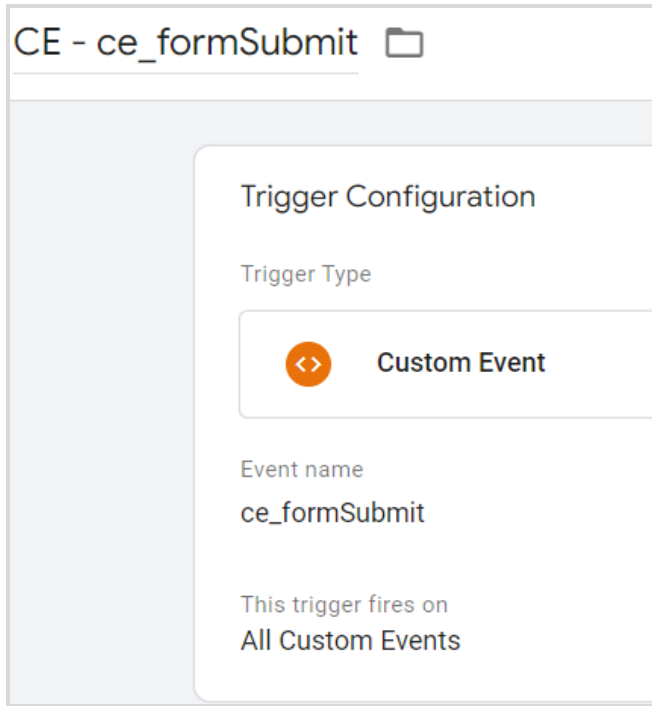
## Event Naming Convention for GA4

Event name: form\_submit

<i>Parameter name</i>	<i>Value</i>
event_cat	form
form_name	{{DL - formName}}

## Trigger

CE - ce\_formSubmit Custom Event: Event name = ce\_formSubmit | All Custom Events



## Field Input

After a form is submitted. Collect the user input or selection. Push the **ce\_formField** event into the dataLayer, along with the **form name**, **field name** and the **field value**.

**example:** `<script>  
dataLayer.push({  
'event': 'ce_formField', 'fieldName': '[FIELD NAME]', 'formName': '[FORM NAME]', 'fieldValue': '[FIELD VALUE]'}); </script>`

Dynamic Parameters:

[FIELD NAME]: The name of the field populated

[FORM NAME]: The name of the form that was submitted

[FIELD VALUE]: The text that the user input or selected

## Event Naming Convention for GA Universal

Category	Action	Label
form	input ~ {{DL - formName}}	{{DL - fieldName}}: {{DL - fieldValue}}

## Event Naming Convention for GA4

Event name: form\_input

<i>Parameter name</i>	<i>Value</i>
event_cat	form
form_name	{{DL - formName}}
field_name	{{DL - fieldName}}
field_value	{{DL - fieldValue}}

## Trigger

CE - ce\_formField Custom Event: Event name = ce\_formField | All Custom Events

The screenshot shows a configuration window for a trigger. At the top, it says "CE - ce\_formField" with a folder icon. The main title is "Trigger Configuration". Under "Trigger Type", there is a selection box with an orange double-arrow icon and the text "Custom Event". Below this, the "Event name" is set to "ce\_formField". At the bottom, it states "This trigger fires on All Custom Events".

## Multipage Form Usage

For multi-page forms, each time a page is submitted, push the **ce\_formPageSubmit** event into the dataLayer along with the **form name**, **form page** and **form step number**.

**example:** <script>  
 dataLayer.push({  
 'event': 'ce\_formPageSubmit', 'formName': '[FORM NAME]', 'formPage': '[FORM PAGE]', 'formStepNumber':  
 '[FORM STEP NUMBER] }); </script>

Dynamic Parameters:

[FORM PAGE]: The name of the form page that was completed (e.g. 'Personal Information', 'Contact Info', 'Interests').

[FORM NAME]: The name of the form in progress

[FORM STEP NUMBER]: If the form contains 4 pages, and the user has just completed the 2nd page, then this value will be 2.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>form</i>	<i>page completed</i>	<i>{{DL - formName}}   {{DL - formPage}}   Step {{DL - formStepNumber}}</i>

### Event Naming Convention for GA4

Event name: form\_page\_submit

<i>Parameter name</i>	<i>Value</i>
event_cat	form
form_page	{{DL - formPage}}
form_name	{{DL - formName}}
form_step_no	{{DL - formStepNumber}}


### Trigger

CE - ce\_formPageSubmit Custom Event: Event name = ce\_formPageSubmit | All Custom Events

CE - ce\_formPageSubmit

Trigger Configuration

Trigger Type

 Custom Event

Event name  
ce\_formPageSubmit

This trigger fires on  
All Custom Events

# User Variables

## Registration

When a user registers their account, push the **ce\_accountRegistration** event into the dataLayer.

```
example: <script>  
  dataLayer.push({  
    'event': 'ce_accountRegistration', 'userId': '[USER ID]', 'loginStatusSession': true }); </script>
```

Dynamic Parameters:

[USER ID]: the user id number for the new account.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
account	register	{{DL - userID}}

### Event Naming Convention for GA4

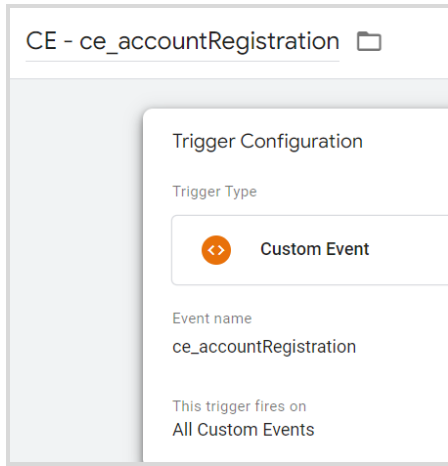
Event name: account\_register

<i>Event parameter name</i>	<i>Value</i>
event_cat	account
user_id	{{DL - userID}}
login_status	{{DL - loginStatusSession}}

<i>User property name</i>	<i>Value</i>
user_id	{{DL - userID}}

### Trigger

CE - ce\_accountRegistration Custom Event: Event name = ce\_accountRegistration | All Custom Events



## Logout

When a user logs out of their account, push the **ce\_accountLogout** event into the dataLayer.

**example:** `<script>`  
`dataLayer.push({`  
`'event': 'ce_accountLogout', 'userId': '[USER ID]', 'loginStatusSession': false`  
`}); </script>`

**Dynamic Parameters:**

[USER ID]: The user id number for the account.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
account	logout	-

### Event Naming Convention for GA4

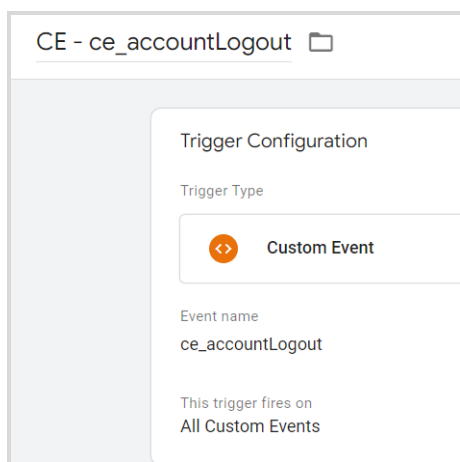
Event name: account\_logout

<i>Event parameter name</i>	<i>Value</i>
event_cat	account
user_id	{{DL - userId}}
login_status	{{DL - loginStatusSession}}

User property name	Value
user_id	{{DL - userId}}

## Trigger

CE - ce\_accountLogout Custom Event: Event name = ce\_accountLogout | All Custom Events



## Login

When a user logs in to their account, push the **ce\_accountLogin** event into the dataLayer.

```
example: <script>
  dataLayer.push({
    'event': 'ce_accountLogin', 'userId': '[USER ID]', 'loginStatusSession': true
  }); </script>
```

Dynamic Parameters:

[USER ID]: the user id number for the new account.

## Event Naming Convention for GA Universal

Category	Action	Label
account	login	-



## Event Naming Convention for GA4

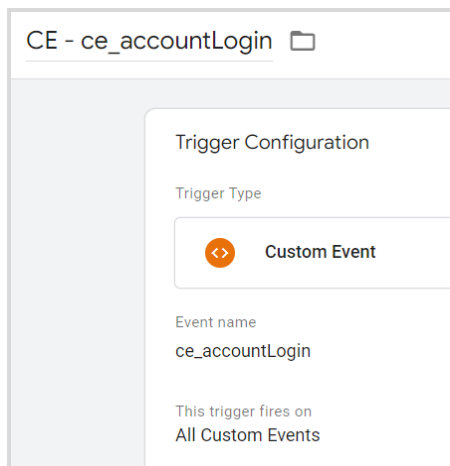
Event name: login

<i>Event parameter name</i>	<i>Value</i>
event_cat	account
user_id	{{DL - userId}}
login_status	{{DL - loginStatusSession}}

<i>User property name</i>	<i>Value</i>
user_id	{{DL - userId}}

## Trigger

CE - ce\_accountLogin Custom Event: Event name = ce\_accountLogin | All Custom Events



## Cookie Permissions

When the user indicates cookie access on the "Cookies Allowed" popup, push the **ce\_cookiesAllowed** event to the dataLayer along with the **cookiesAccepted** field.

```
example: <script>
  dataLayer.push({
    'event': 'ce_cookiesAllowed',
    'cookiesAccepted': '[COOKIES ACCEPTED]'
  })
</script>
```

```
}); </script>
```

#### Dynamic Parameters:

[COOKIES ACCEPTED]: "Cookies Allowed" if the user chose to allow cookies, "Cookies Denied" otherwise.

### Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
cookie_permissions	{{DL - cookiesAccepted}}	-

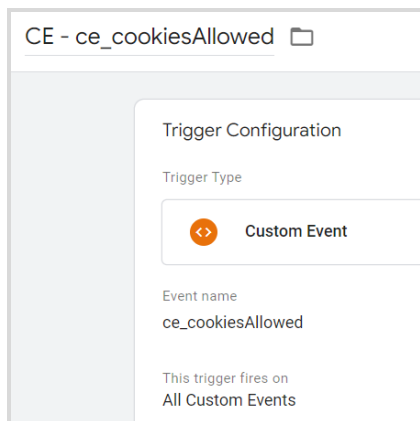
### Event Naming Convention for GA4

Event name: cookie\_permissions

<i>Parameter name</i>	<i>Value</i>
event_cat	cookie_permissions
cookies_accepted	{{DL - cookiesAccepted}}

### Trigger

CE - ce\_cookiesAllowed Custom Event: Event name = ce\_cookiesAllowed | All Custom Events



# Ecommerce

Ecommerce actions are implemented primarily by pushing ecommerce event details into the dataLayer as JSON objects according to the format laid out by Google. The following sections will describe in more detail how to implement each of the Enhanced Ecommerce events.

## Product Tracking

### Product Impressions

Measure product impressions by using the **ee\_productImpression** event and an impressions array containing one or more products. A product impression is fired when a product thumbnail link is visible to the user.

example:

```
<script>
dataLayer.push({
  'event': 'ee_productImpression',
  'ecommerce': {
    'currencyCode': '[CURRENCY CODE]', // (ex. 'USD')
    'impressions': [{
      'name': '[PRODUCT NAME]', // Name or ID is required.
      'id': '[PRODUCT ID]', // Name or ID is required.
      'price': '[PRODUCT PRICE]',
      'category': '[PRODUCT TYPE]',
      'list': '[PRODUCT LIST]',
      'position': '[PRODUCT POSITION]'
    }, {
      'name': '[PRODUCT NAME]', // Name or ID is required.
      'id': '[PRODUCT ID]', // Name or ID is required.
      'price': '[PRODUCT PRICE]',
      'category': '[PRODUCT TYPE]',
      'list': '[PRODUCT LIST]',
      'position': '[PRODUCT POSITION]'
    }
  ]
});
</script>
```

Dynamic Parameters:

[CURRENCY CODE]: Identifies the currency of the product values, such as 'USD' or 'BRL'.

[PRODUCT NAME]: The name of the product. This is required.

[PRODUCT ID]: A unique identifier of the product, if available.

[PRODUCT PRICE]: The price of the product, ex. '274.00'.

[PRODUCT CATEGORY]: The type of product, ex. 'Outerwear', 'Shoes', 'Accessories'

[PRODUCT LIST]: The name of the list where the product is shown, ex. 'Featured Products', 'Related Products'. If applicable.

[PRODUCT POSITION]: The position of the product in the list, represented by an integer starting with '1'. If applicable.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>ecommerce</i>	<i>product impression</i>	-

More settings > Ecommerce: true > Use Data Layer

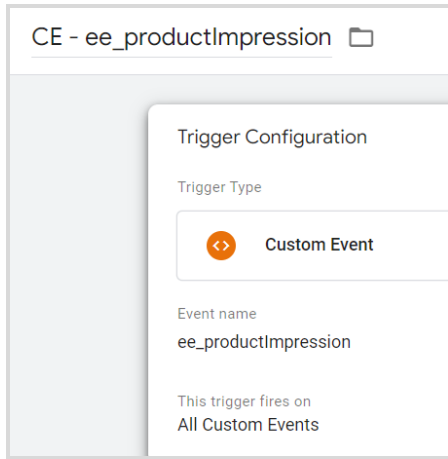
### Event Naming Convention for GA4

Event name: view\_item\_list

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.impressions}}

### Trigger

CE - ee\_productImpression Custom Event: Event name = ee\_productImpression | All Custom Events



## Product Clicks

Measure clicks on product links by pushing the **ee\_productClick** action to the data layer, along with a products array, as in the example below. The product array must match to the product the user clicked on.

code example:

```
<script>
dataLayer.push({
  'event': 'ee_productClick',
  'ecommerce': {
    'click': {
      'actionField': {'list': '[PRODUCT LIST]'}, // List if applicable
      'products': [{
        'name': '[PRODUCT NAME]', // Name or ID is required.
        'id': '[PRODUCT ID]', // Name or ID is required.
        'price': '[PRODUCT PRICE]',
        'category': '[PRODUCT CATEGORY]',
        'position': '[PRODUCT POSITION]'
      }]
    }
  }
});
</script>
```

### Dynamic Parameters:

[PRODUCT NAME]: The name of the product. This is required.

[PRODUCT ID]: A unique identifier of the product, if available.

[PRODUCT PRICE]: The price of the product, ex. '274.00'.

[PRODUCT CATEGORY]: The type of product, ex. 'Outerwear', 'Shoes', 'Accessories'

[PRODUCT LIST]: The name of the list where the product is shown, ex. 'Featured Products', 'Related Products'. If applicable.  
 [PRODUCT POSITION]: The position of the product in the list, represented by an integer. If applicable.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>ecommerce</i>	<i>product click</i>	-

More settings > Ecommerce: true > Use Data Layer

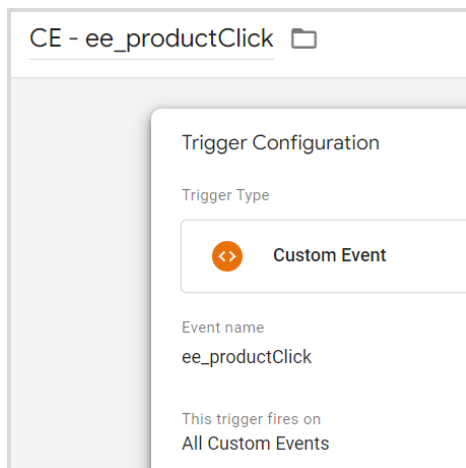
### Event Naming Convention for GA4

Event name: `select_item`

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.click.products}}

### Trigger

CE - ee\_productClick Custom Event: Event name = ee\_productClick | All Custom Events



## Product Detail Views

Measure a view of product details (i.e. the product details page) by pushing a **ee\_productDetailView** action to the data layer, along with the products array containing the product being viewed.

code example

```
<script> dataLayer.push({
  'event': 'ee_productDetailView',
  'ecommerce': { 'detail': {
    'actionField': {'list': '[PRODUCT LIST]}, // List if applicable
    'products': [{
      'name': '[PRODUCT NAME]', // Name or ID is required.
      'id': '[PRODUCT ID]', // Name or ID is required.
      'price': '[PRODUCT PRICE]',
      'category': '[PRODUCT CATEGORY]'
    }]
  }
});
</script>
```

### Dynamic Parameters

[PRODUCT NAME]: The name of the product. This is required.

[PRODUCT ID]: A unique identifier of the product, if available.

[PRODUCT PRICE]: The price of the product, ex. '274.00'.

[PRODUCT CATEGORY]: The type of product, ex. 'Membership', 'Subscription', 'Exam Prep', 'Book', 'Event Registration'.

[PRODUCT LIST]: The name of the list where the product is shown, ex. 'Featured Products', 'Related Products'. If applicable.

## Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>ecommerce</i>	<i>pdp view</i>	-

More settings > Ecommerce: true > Use Data Layer

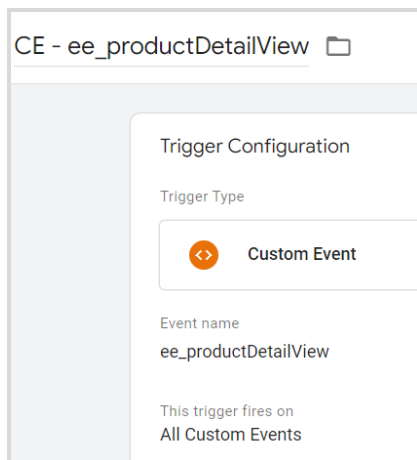
## Event Naming Convention for GA4

Event name: view\_item

Parameter name	Value
event_cat	ecommerce
items	{{DL - ecommerce.detail.products}}

## Trigger

CE - ee\_productDetailView Custom Event: Event name = ee\_productDetailView | All Custom Events



## Cart Tracking

### Add to cart

When a user adds a product to the cart, push the **ee\_addToCart** event into the dataLayer, along with the ecommerce add action and products array.

code example

```
<script>
dataLayer.push({
  'event': 'ee_addToCart',
  'ecommerce': {
    'currencyCode': '[CURRENCY CODE]',
    'add': {
      'Products': [{
        'name': '[PRODUCT NAME]', // Name or ID is required.
        'id': '[PRODUCT ID]', // Name or ID is required.
      }
    ]
  }
}
```



```

        'price': '[PRICE]',
        'category': '[CATEGORY]',
        'quantity': '[PRODUCT QTY]',
        'coupon': '[COUPON CODE]' // if applicable
        'variant': '[VARIANT]'
    },
    {... additional products}
]
}
}
});
</script>

```

#### Dynamic Parameters

- [PRODUCT NAME]: The name of the product. This is required.
- [PRODUCT ID]: A unique identifier of the product, if available.
- [PRODUCT PRICE]: The price of the product, ex. '274.00'.
- [PRODUCT CATEGORY]: The type of product, ex. 'Outerwear', 'Shoes', 'Accessories'
- [PRODUCT QTY]: The number of products added to the cart.
- [COUPON CODE]: If a promo code applies to the product.

#### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
ecommerce	add to cart	-

More settings > Ecommerce: true > Use Data Layer

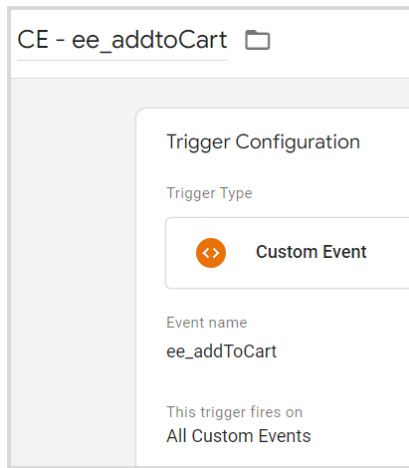
#### Event Naming Convention for GA4

Event name: add\_to\_cart

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.add.products}}

#### Trigger

CE - ee\_addToCart Custom Event: Event name = ee\_addToCart | All Custom Events



## Remove from the cart

When a user removes a product from the cart, push the **ee\_removeFromCart** event into the dataLayer, along with the ecommerce remove action and products array.

code example

```
<script>

dataLayer.push({
  'event': 'ee_removeFromCart',
  'ecommerce': {
    'currencyCode': '[CURRENCY CODE]',
    'remove': {
      'products': [{
        'name': '[PRODUCT NAME]', // Name or ID is required.
        'id': '[PRODUCT ID]', // Name or ID is required.
        'price': '[PRICE]',
        'category': '[CATEGORY]',
        'quantity': '[PRODUCT QTY]',
        'coupon': '[COUPON CODE]' // if applicable'
      }
    ]
  }
});
</script>
```

### Dynamic Parameters

[PRODUCT NAME]: The name of the product. This is required.

[PRODUCT ID]: A unique identifier of the product, if available.

[PRODUCT PRICE]: The price of the product, ex. '274.00'.

[PRODUCT CATEGORY]: The type of product, ex. 'Outerwear', 'Shoes', 'Accessories'

[PRODUCT QTY]: The number of products removed from the cart.

## Event Naming Convention for GA Universal

<i>Category</i>	<i>Action</i>	<i>Label</i>
ecommerce	remove from cart	-

More settings > Ecommerce: true > Use Data Layer

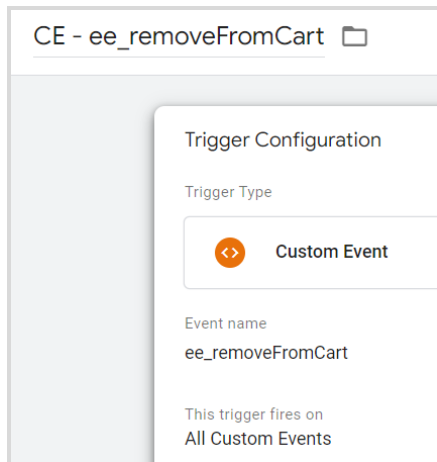
## Event Naming Convention for GA4

Event name: remove\_from\_cart

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.remove.products}}

## Trigger

CE - ee\_removeFromCart Custom Event: Event name = ee\_removeFromCart | All Custom Events



## Adding a discount

If a user adds a coupon code to the order, push the **ee\_orderDiscount** event to the dataLayer along with the name and code of the discount.

code example

```
dataLayer.push({  
  'event': 'ee_orderDiscount',
```

```
'discountCode': '[DISCOUNT CODE]',
'discountName': '[DISCOUNT NAME]'
});
```

#### Dynamic Parameters

- [DISCOUNT CODE]: The discount code that was applied.
- [DISCOUNT NAME]: the name of the promotion or discount, ex. "20% Off For New Students"

#### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>order discount</i>	<i>order discount</i>	{{DL - discountName}} ~ {{DL - discountCode}}

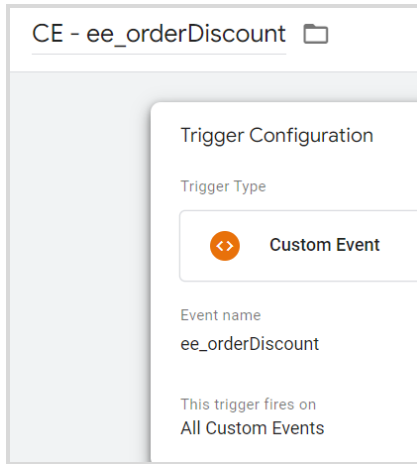
#### Event Naming Convention for GA4

Event name: apply\_discount

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
discount_name	{{DL - discountName}}
discount_code	{{DL - discountCode}}

#### Trigger

CE - ee\_orderDiscount Custom Event: Event name = ee\_orderDiscount | All Custom Events



## Checkout Tracking

To measure the checkout process, which might include a checkout button and one or more checkout pages where users enter shipping and payment information, use the checkout action and the step field to indicate which stage of the checkout process is being measured.

code example

```
<script>
dataLayer.push({
  'event': 'ee_checkout',
  'ecommerce': {
    'checkout': {
      'actionField': {'step': '[CHECKOUT STEP]', // integers that begin with '1'
      'option': '[PAYMENT METHOD]', // 'option' field used for user selections in this
      step
      'products': [{
        'name': '[PRODUCT NAME]', // ID or Name is required.
        'id': '[PRODUCT ID]', // ID or Name is required.
        'price': '[PRODUCT PRICE]',
        'category': '[PRODUCT CATEGORY]',
        'quantity': '[PRODUCT QTY]',
        'coupon': '[PROMO CODE]'
      }] //Multiple Products if applicable
    }
  }
});
</script>
```

### Dynamic Parameters

- **[CHECKOUT STEP]**: The step completed, represented by an integer starting with '1' (These will be mapped to the step name in the Google Analytics Ecommerce reports).

- [PAYMENT METHOD]: The payment method as selected by the user. Omit if this payment has been chosen yet. Examples: 'Visa', 'Mastercard', 'Financing' ...
- [PROMO CODE]: If a promo code applies to the product.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
ecommerce	checkout step	{{DL - ecommerce.checkout.actionField.st ep}}

More settings > Ecommerce: true > Use Data Layer

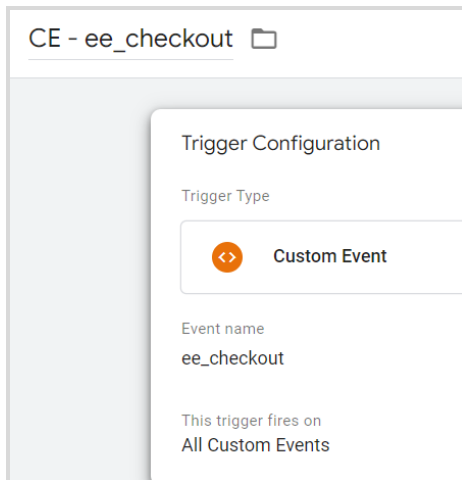
### Event Naming Convention for GA4

Event name: checkout\_step

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
step	{{DL - ecommerce.checkout.actionField.step}}

### Trigger

CE - ee\_checkout Custom Event: Event name = ee\_checkout | All Custom Events



## Checkout Options

If you have already measured a checkout step, but you would like to capture additional information about the same step, you may use **ee\_checkoutOption** event with the **checkout\_option** action.

Example:

```
function onCheckoutOption(step, checkoutOption) {  
  //This function can be called when the option for a checkout step changes (ex. User chooses payment  
  method)  
  dataLayer.push({  
    'event': 'ee_checkoutOption',  
    'ecommerce': {  
      'checkout_option': {  
        'actionField': {'step': step, 'option': checkoutOption}  
      }  
    }  
  });  
}
```

## Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
ecommerce	checkout option	-

More settings > Ecommerce: true > Use Data Layer

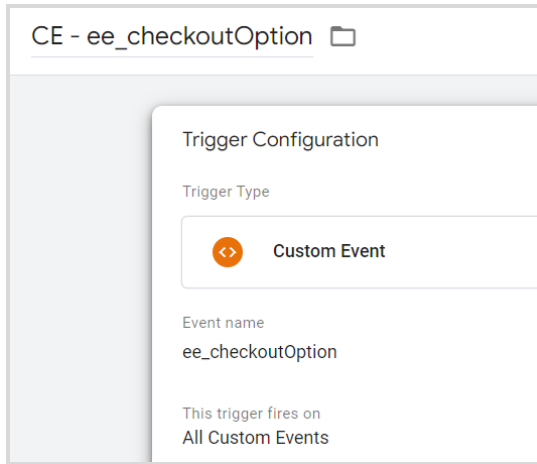
## Event Naming Convention for GA4

Event name: checkout\_option

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
step	{{DL - ecommerce.checkout.actionField.step}}
option	{{DL - ecommerce.checkout.actionField.option}}

## Trigger

CE - ee\_checkoutOption Custom Event: Event name = ee\_checkoutOption | All Custom Events



## Purchase Tracking

To track a purchase, the transaction details must be pushed into the dataLayer using the ecommerce **ee\_purchase** action. The **actionField** object contains details about the transaction itself, and the **products** list contains a list of purchased products.

example:

```
dataLayer.push({
  'event': 'ee_purchase',
  'ecommerce': {
    'purchase': {
      'actionField': {
        'id': '[TRANSACTION ID]', // Transaction ID. Required for purchases
        'revenue': [REVENUE], // Total transaction value (incl. tax and shipping)
        'tax': '[TAX]',
        'shipping': '[SHIPPING]'
      },
      'Products': [{ // List of productFieldObjects
        'name': '[PRODUCT NAME]', // One of Name or ID is required.
        'id': '[PRODUCT ID]',
        'price': '[PRICE]',
        'category': '[CATEGORY]', // Category of the product i.e "Books",
        "Membership", "Conference" ...
        'quantity': [QUANTITY],
        'coupon': '[COUPON CODE]' // if applicable
      }]
    }
  }
});
```

Dynamic Parameters



- [REVENUE]: The total amount paid for the transaction. Including tax, shipping and discounts
- [TAX]: The amount of taxes paid on the purchase.
- [SHIPPING]: The amount paid for shipping
- [TRANSACTION ID]: The unique identifier for the transaction.
- [COUPON CODE]: Add a promo code if one is applied.

### Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
ecommerce	transaction complete	-

More settings > Ecommerce: true > Use Data Layer

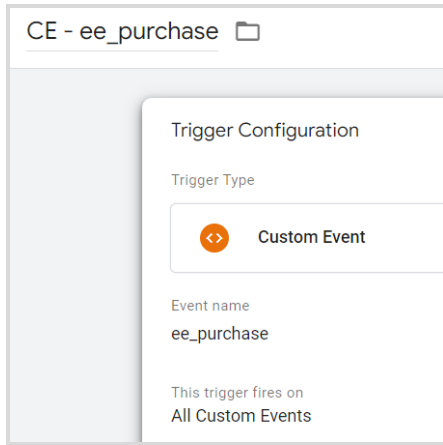
### Event Naming Convention for GA4

Event name: purchase

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.purchase.products}}
currency	USD
value	{{DL - ecommerce.purchase.actionField.revenue}}
tax	{{DL - ecommerce.purchase.actionField.tax}}
shipping	{{DL - ecommerce.purchase.actionField.shipping}}
transaction_id	{{DL - ecommerce.purchase.actionField.id}}
coupon	{{DL - ecommerce.purchase.actionField.coupon}}

### Trigger

CE - ee\_purchase Custom Event: Event name = ee\_purchase | All Custom Events



## Purchasing with a discount

When a user completes a purchase with a coupon code applied, push the **ee\_discountPurchased** event to the dataLayer along with the name and code of the discount.

example:

```
dataLayer.push({
  'event': 'ee_discountPurchased',
  'discountCode': '[DISCOUNT CODE]',
  'discountName': '[DISCOUNT NAME]',
  'transactionID': '[TRANSACTION ID]'
});
```

### Dynamic Parameters

[DISCOUNT CODE]: The discount code that was applied.

[DISCOUNT NAME]: The name of the promotion or discount, ex. "20% Off For New Students"

[TRANSACTION ID]: The unique identifier for the transaction.

## Event Naming Convention for GA Universal

<b>Category</b>	<b>Action</b>	<b>Label</b>
<i>order discount</i>	<i>discount purchased</i>	<i>{{DL - discountName}} ~ {{DL - discountCode}}</i>

More settings > Ecommerce: true > Use Data Layer

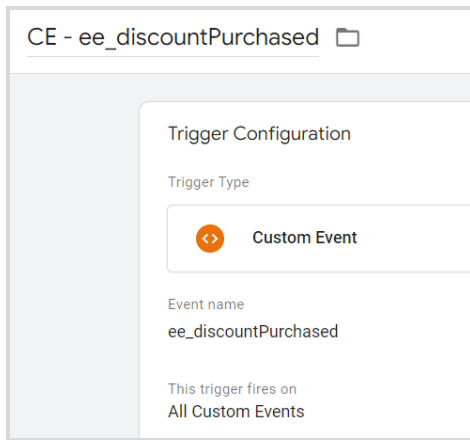
## Event Naming Convention for GA4

Event name: purchase\_discount

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
discount_code	{{DL - discountCode}}
discount_name	{{DL - discountName}}
transaction_id	{{DL - transactionID}}

## Trigger

CE - ee\_discountPurchased Custom Event: Event name = ee\_discountPurchased | All Custom Events



## Refund Tracking

A full refund can be tracked by pushing the **ee\_refund** event and the ecommerce refund action into the dataLayer with the transaction id and no other information.

example

```
dataLayer.push({
  'event': 'ee_refund',
  'ecommerce': {
    'refund': {
      'actionField': {'id': '[TRANSACTION ID]} // Transaction ID Required
    }
  }
});
```

Dynamic Parameters

- [TRANSACTION ID]: The same unique ID that was included with the original purchase.

To track **partial refund** of only a few products from a transaction, include the list of products to be refunded alongside the transaction data.

example

```
dataLayer.push({
  'event': 'ee_refund',
  'ecommerce': {
    'refund': {
      'actionField': {'id': '[TRANSACTION ID]}, // Transaction ID Required
      'products': [
        {'id': '[PRODUCT ID]', 'quantity': [QUANTITY]}, // ID and Quantity Required
        {'id': '[PRODUCT ID]', 'quantity': [QUANTITY]}
      ]
    }
  }
});
```

Dynamic Parameters

- [PRODUCT ID]: Must include either the product id or the product name.
- [QUANTITY]: The number of items that were refunded.

## Event Naming Convention for GA Universal

Category	Action	Label
ecommerce	refund processed	-

More settings > Ecommerce: true > Use Data Layer

## Event Naming Convention for GA4

Event name: refund

<i>Parameter name</i>	<i>Value</i>
event_cat	ecommerce
items	{{DL - ecommerce.refund.products}}
transaction_id	{{DL - ecommerce.refund.actionField.id}}

## Trigger

CE - ee\_refund Custom Event: Event name = ee\_refund | All Custom Events

