



Data Activation 3rd Party Integration – Google Analytics

Instructions

Contents

1	Introduction	4
1.1	Purpose of this Document	4
1.2	Target Audience	4
1.3	Considerations	4
2	Common Use Cases	5
2.1	To enhance re-targeting capabilities for known customer	5
2.2	To build a Unified customer data view	5
3	Design Concepts.....	6
3.1	What is retargeting?	6
3.2	Customer Data Platform (CDP)	6
3.2.1	Identity Mapping.....	6
3.2.2	What is Pixel tracking.....	7
3.2.3	What is Server-to-Server tracking.....	7
4	Identity Mapping	9
4.1	Custom Event for identity mapping	9
5	Pixel Integration (Client Side)	11
5.1	Requirements.....	11
5.1.1	Define Custom Dimension field(s) in GA to store SAS segment(s)	11
5.1.2	Google Global site tag	11
5.1.3	CI360 Tag Deployed	11
5.1.4	CI360 Pixel Placeholder	12
5.2	Create Spot in CI360	12
5.3	Create Custom Event	13
5.4	Create creative with GA pixel installed.....	15
5.5	Define Web Task	16
5.6	Trigger GA Pixel via SAS Creative	19
5.7	Verification.....	19
5.8	Example.....	19
5.8.1	EXAMPLE: USE GA EVENT IN OTHER PLATFORMS	19
5.8.2	EXAMPLE: USE GA- CONVERSION PIXEL VIA SAS	21
6	API (Server-to-Server)	23
6.1	Requirements.....	23
6.1.1	Define Custom Dimension field(s) in GA to store SAS segment(s)	23
6.1.2	Google Global site tag	23
6.1.3	CI360 Tag Deployed	23
6.2	Connector setup.....	24
6.3	Task Setup	26
6.4	Verification.....	27
7	Appendix	29
7.1	External References	29
7.2	Source Code	29
7.2.1	Google Tag Example:.....	29
7.2.2	Google API Endpoint	29

7.2.3 SAS CI360 Tag Example Code	30
7.2.4 SAS Generic Pixel Placeholder	30
7.2.5 SAS Google Placeholder	30
7.2.6 Google Upload APIs.....	Error! Bookmark not defined.
7.3 Connector JSON Payloads	31

Information about This Document

Document Control	Error! Bookmark not defined.
Contacts.....	Error! Bookmark not defined.
Revision History	Error! Bookmark not defined.

1 Introduction

1.1 Purpose of this Document

The purpose of this document is to help guide in the integration of CI360 with Google products. This includes sending user identifiers after segmentation by CI360 platforms.

1.2 Target Audience

Those who will be implementing integrations with Google Services

1.3 Considerations

Platforms change and due to the fast changing nature of APIs and technology be sure to read the Google documentation to understand what is needed as some things might have changed

2 Common Use Cases

There are two main use cases for data activation on 3rd Party platforms:

2.1 To enhance re-targeting capabilities for known customer

Digital Advertising can reach very large audiences in both new and known visitors. It is great way to tell brand stories at scale and in context through ads on various devices or channels. However, digital campaigns may waste or miss the target audience without a re-targeting solution to manage the customer experience. The combination of third-party platform and first-party data will enrich the re-targeting capabilities for digital campaigns which run off site and increase marketing ROI through improved relevance and accuracy.

2.2 To build a Unified customer data view

A unified customer data view combines all the data points of visitors from data collected off site (third-party) and on site (first-party). These data are able to give a better picture of known visitors and to provide the right offer in the right moment in time regardless they were looking advertisement from the internet or navigating within your website. The unified customer data view is capable of enriching re-targeting capabilities for online advertising and marketing campaign as well as in site next best action offering.

3 Design Concepts

3.1 What is retargeting?

Retargeting is a method for putting the organization's brand in front of consumers after they have left the website or app and persuading them to reconsider an offer or value proposition from the brand. It often refers to reaching out to lost visitors by Display Ads and its typically performed by using the third-party platform's event (tracking pixel and cookie) to achieve.

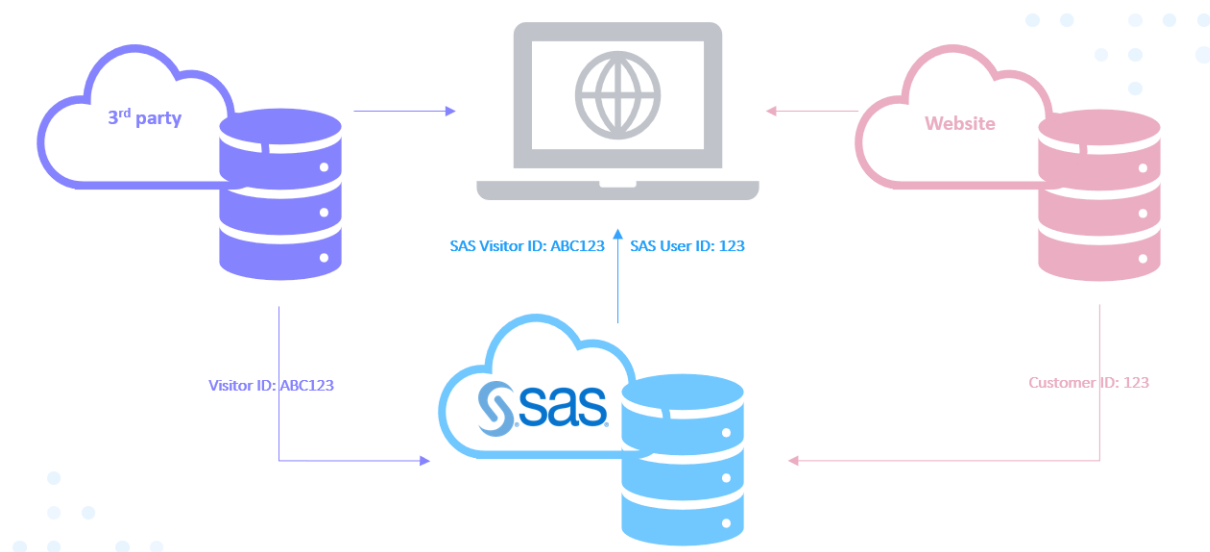
3.2 Customer Data Platform (CDP)

In this article, we are going to use CI360 to combine 1st party data (online and offline), marketing data, and third-party data. Hence, CI360 can be act as CDP to provide unified customer data view for enrich the re-targeting for online campaign which run off-site platform (such as Google Ads), and there are two steps and three methods to achieve this:

1. Identity Mapping (between each platform solutions)
2. Sending Data to third-party platform:
 - Pixel Tracking
 - Server-to-Server Tracking

3.2.1 Identity Mapping

The following diagram shows how marketing tool act as “middle agent” to map identifier and later to exchange data between organization website / app, marketing automation tool as well as third-party platform:

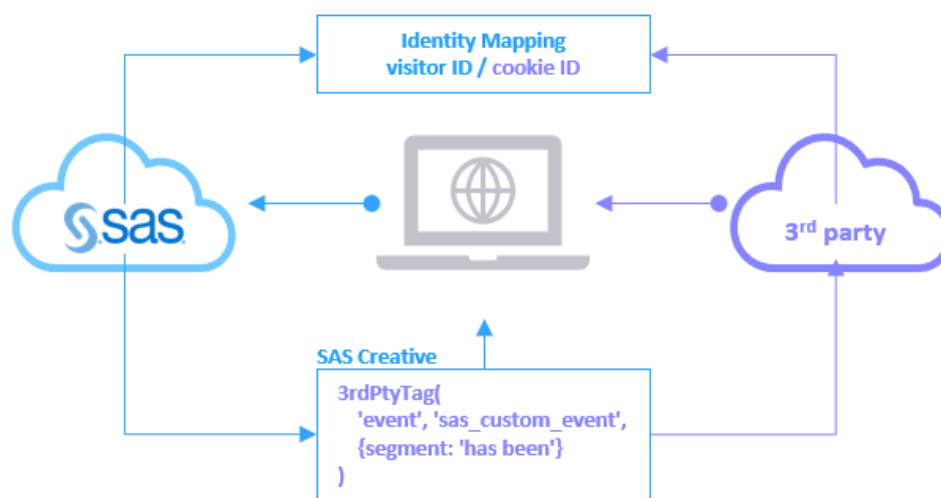


In CI360, the identity mapping can be done by Custom Event with additional attribution, it will be cover in later sections.

3.2.2 What is Pixel tracking

Pixel tracking are snippets of code provided from third-party platform that are loaded on the website, in specific pages, to track visitors in-site or in-app behavior. The code is usually executed on client side (visitor's web browser or app) which sending the information along with unique cookie ID / visitor ID to third-party platform.

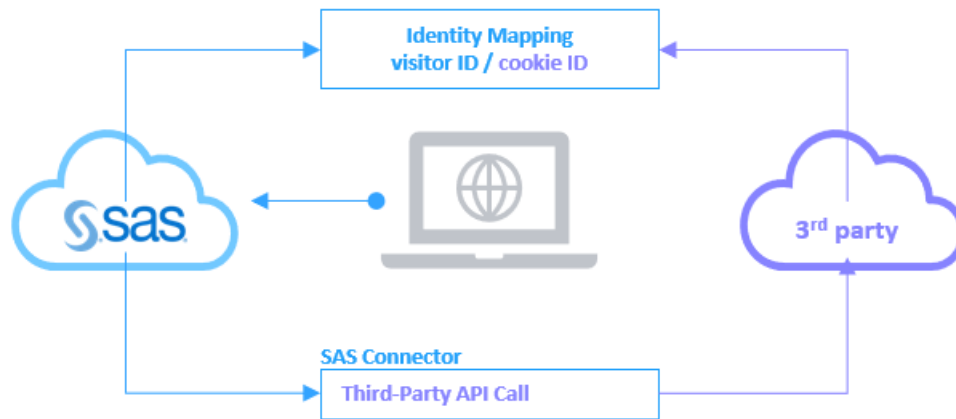
In this offering, we are going to use pixel integration via SAS CI360 Creative to serve third-party tracking code. This approach is different to traditionally third-party tracking which only embedded in the specified page; CI360 will decide the right moment of the customer journey to fire the event or share the segment.



3.2.3 What is Server-to-Server tracking

Server-to-Server tracking is using APIs call to fire tracking event directly from the solution backend to third party platform which extended the tracking beyond visitor exited the website or app. It is also enhancing the site security as server to server tracking doesn't require any third-party coding to be installed on the site and tracking event can be control and triggered by solution owned by the organizations.

In this offering, we are going to use Server-Side API framework (Connector) to execute third-party tracking server call. This approach allows to share event / segment without embedded third-party code on your website to maintain security / data privacy. As well as, for the visitor beyond the site exit where pixel integration unable to reach out.




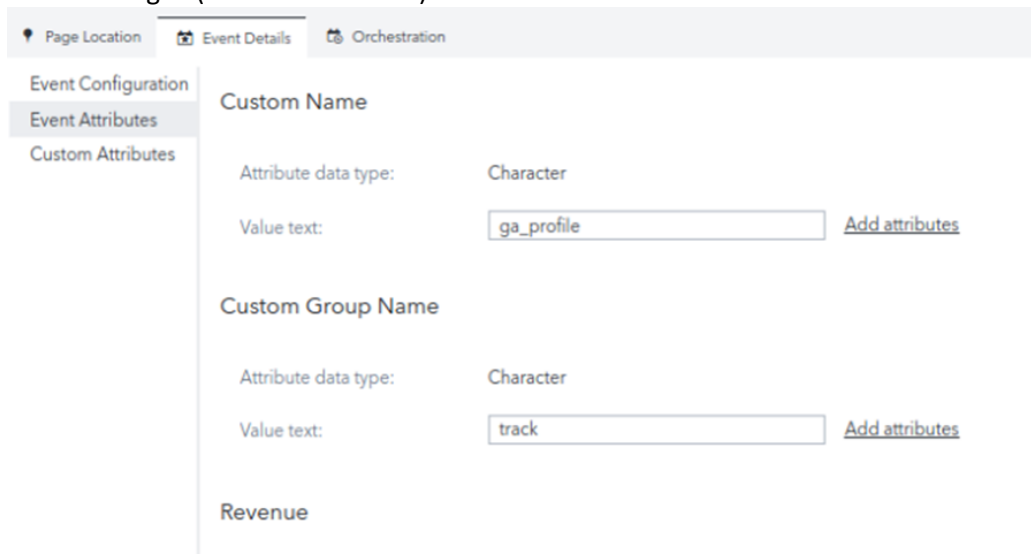
4 Identity Mapping


In order to get Pixel integration and Server-to-Server (API) to work with third-party platforms, the first step is mapping third-party platform ID with SAS ID.

4.1 Custom Event for identity mapping

This event will be setup to create to collect the Google Client ID from the client's computer. The ID we want is stored in the gaGlobal.vid variable. Note that this event won't necessarily fire our external event and Server-to-Server communication through the API, we will do that with an **External System Task** using this event as a trigger which we can further customize to add additional conditions and segmentation before sending data to GA.

- Click **Events** in the left navigation menu
- Create a new event by clicking on 
- You can create any type of event, but for this we will create a **Custom Event** based on a **Page View**
- Choose the page you want this to fire on (this could simply be every page if you restrict using segments in a later step, or it could be specific pages)
- Set the **Custom Name** and the **Custom Group Name**, note that these are the values that will be sent to Google. (Write these down)




- Create a new Custom Attribute by clicking 
- Choose **Page Details > JavaScript Variable**
- Set the attribute name to **ga_cid**
- Variable name to **gaGlobal.vid**
- Return: **Everything**
- Attribute data type: **Character**
- Personally identifiable information: **Unchecked** (note that we keep this unchecked so we can connect other information from other systems and connect it with our own identifiers if needed)

later in our data)

ga_cid

Attribute name: *	ga_cid
Source attribute:	JavaScript variable
Variable name: *	gaGlobal.vid
Return:	Everything ▼
Attribute data type:	Character ▼
Personally identifiable information: ⓘ	<input type="checkbox"/>

- m. Click  to save the event naming it whatever you want (remember the event name won't be used to send to GA, the Event Attributes will be used instead)

5 Pixel Integration (Client Side)

Google Analytics (GA) is using the Global Site Tags (gtag.js) to fire event pixel for collect data. This document attempts to standardize the process of SAS Ci360 to control GA event pixel. This also assumes that there is one Ci360 tenant and one GA instance, some code might need to be modified for testing environments or otherwise

5.1 Requirements

5.1.1 Define Custom Dimension field(s) in GA to store SAS segment(s)

Create **Custom Dimensions** under **GA Admin > Custom Definition** section, these fields are going to store SAS segment (note that GA's Custom Dimension did not support multiple values in single field, if you're making several SAS segment to GA, it would be best practice to add more Custom Dimension with different field name).

+ NEW CUSTOM DIMENSION	
Custom Dimension Name	Index
segment	5

In my example, custom dimension index 1 to 4 already in use for other purpose, hence it started with **Index 5**. And this index number is the key to map from the site code, please use the correct index number which mapped in your case.

5.1.2 Google Global site tag

Embed the gtag.js with the correct <product_id> (which provided by Google Analytics Account). And add "custom_map" property to set **custom dimension index** and **name** which created from last step accordingly, in my case, will be dimension5 and segment as below:

```
<!-- Global site tag (gtag.js) - Google Products -->
<script async src="https://www.googletagmanager.com/gtag/js">
</script>
<script>
window.dataLayer = window.dataLayer || [];
function gtag(){dataLayer.push(arguments);}
gtag('js', new Date());

gtag('config', '<product_id>', {
  custom_map: {
    dimension5: "segment"
  }
});
</script>
```

5.1.3 CI360 Tag Deployed

Add in the CI360 tag into the site (note that if you're making several pixel calls for different services, it would be best practice to deploy the below SAS Pixel Placeholder code with several different div's using different IDs, as setting them up the same way will conflict). Please copy the code from your CI360 from the **General Settings > SAS Tag Instruction** section.

```
SAS Ci360 Tag Example:
<!-- SAS Ci360 Tag -->
<script>
(function(ci){
  var ef=window[ci]=function(){
    return ef.q.push(arguments);
  };
  ef.q=[];ef.a={};
})('ci360');
</script>
<script async data-efname='ci360' id='ob-script-async'
  a='<tenant ID>'
  src='https://<server>/js/ot-all.min.js'></script>
```

5.1.4 CI360 Pixel Placeholder


This is specifically to later have a spot we can inject the code into in order to easily send the pixel to the user for them to trigger on their local machine

```
<!-- SAS Pixel Placeholder -->
<div id="sas_google_pixel"></div>
```

Tip: both **SAS Ci360 Tag** and **SAS Pixel Placeholder** can be embedding via tag manager products.

5.2 Create Spot in CI360

This will create a spot for CI360 where we can then inject content, in this case we will inject a tag in order to send data from the website client to the Google servers

- From navigation bar, click **Spots**
- Click the  button to create a new spot
- Select **Web** as the spot type
- Navigate to the **URL** of the web page with **SAS CI360 Tag** and **SAS Pixel placeholder**
- Click on the **Select selector** button and type in “**div#sas_google_pixel**” as below (note that this is specific to the code provided, you can use any div name but it must match the tag that you inject using your tag manager from the above requirements):

☐ Navigate ☒ Select Selector:

- Then, click **Create Spot**
- In **Spot Details Tab**,
 - Select **Spot Configuration** from the menu bar


1. Change the Page matches under Spot Location section to **Any Page:**

Spot Location


Specify the page or pages on which the spot appears.

Page matches:

- ii. select **Spot Attribution** from the menu bar

1. click the  button to add new Attribution for the spot
2. select **URL path (entire path)** as Attribute
3. click **OK**
4. amend the Attribute name to “**page_url**”:

Attribute name: *

- h. click the  button to save the Spot
- i. type in a name for the Spot and then click **Save**
- j. Go to the **Orchestration Tab**
 - i. Click on **Mark Ready and Publish** from drop-down menu:

Mark Ready ▼

Mark Ready


Mark Ready and Publish

- ii. Click **Yes** when promoted by **Publishing the item will change the status to Scheduled or Active**

Tip: The example below fires a Google event if user had been visited X page Y times during Z period.

5.3 Create Custom Event

Here we will create a Custom Event, this is required as the code used in our creative to trigger the pixel will use merge tags in order to replace information to be sent to GA. The instructions will provide how to create an event and which data will be sent to GA but you can create your own event with different criteria and naming.

- a. From navigation bar, click **Events**
- b. Click the  button for create a new event
- c. Select **Custom Event**
- d. Select **Page View** as event type

- e. Enter the **Page URL** for re-targeting
- f. Click **Create Event**
- g. In the **Event Details** Tab
 - i. Go to **Event Configuration**
 1. To confirm the site is correct:

Pages in Event

Specify the page or pages that are included for this event.

Selected page

Page URL:


<https://www.gciap.com/n7.html>

- ii. Go to **Event Attributes**
 1. In Custom Name's value text: type in "**has_been**":

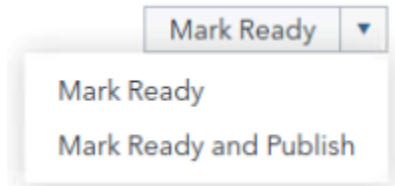
This is a custom value that will be collected in GA and stored in **custom dimension** field
 2. In Custom Group Name's value text: type in "**segment**":

This is a custom value that map with **custom dimension** name created in GA.
- iii. The configuration should look similar to the below:

The screenshot shows the 'Event Configuration' tab in the 'Event Details' section. On the left, there is a sidebar with 'Event Configuration', 'Event Attributes', and 'Custom Attributes'. The main area is titled 'Custom Name' and 'Custom Group Name'. For 'Custom Name', the 'Attribute data type' is 'Character' and the 'Value text' is 'has_been'. For 'Custom Group Name', the 'Attribute data type' is 'Character' and the 'Value text' is 'segment'. Both sections have an 'Add attributes' link.

- h. click the  button to save the Custom Event
- i. type in "**Event: GA Pixel**" (this is just a placeholder name, you can use any event you want), then click **Save**
- j. Go to the **Orchestration** Tab


- i. Click on **Mark Ready and Publish** from drop-down menu:




- ii. Click **Yes** when prompted by **Publishing the item will change the status to Scheduled or Active**

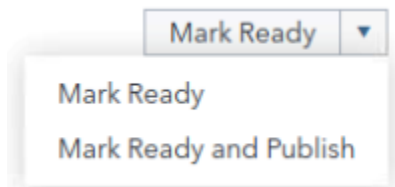
5.4 Create creative with GA pixel installed

This is where we will define our code to be replaced on the client machine in order to be executed. In this case it will be requesting a pixel from the GA servers with specific link parameters to transfer information

- a. From navigation bar, click **Creatives**
- b. Click the  button for create new Creative
- c. Select **HTML** as creative type
- d. Copy the **Google Event Tracking Pixel** into the body tag as below

```
<html>
  <head>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />
    <title></title>
  </head>
  <body>
    <!-- Enter the HTML source code for your creative.
    Note: If this creative contains a reference to an external resource
    such as an image, CSS file, or JavaScript file, then that resource must be
    publicly accessible. Otherwise, the creative cannot render properly. -->
    
  </body>
</html>
```


- e. Click **Done**
- f. click the  button to save the creative
- g. type in **"Creative: GA Pixel: Custom Dimension"** then click **Save**
- h. Go To **Orchestration Tab**
 - a. Click on **Mark Ready and Publish** from drop-down menu:

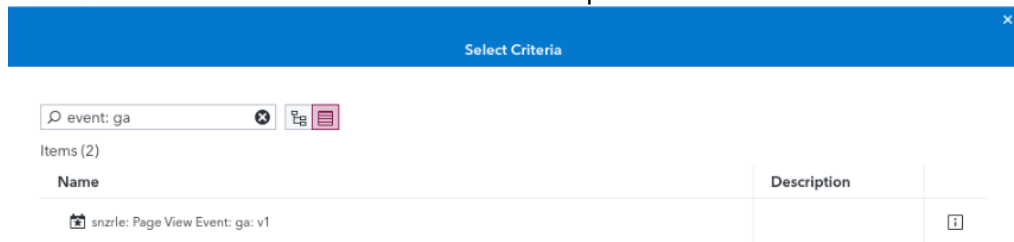


- b. Click **Yes** when prompted by **Publishing the item will change the status to Scheduled or Active**

5.5 Define Web Task

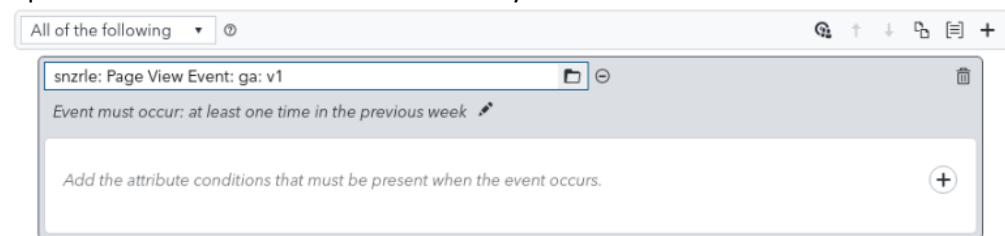
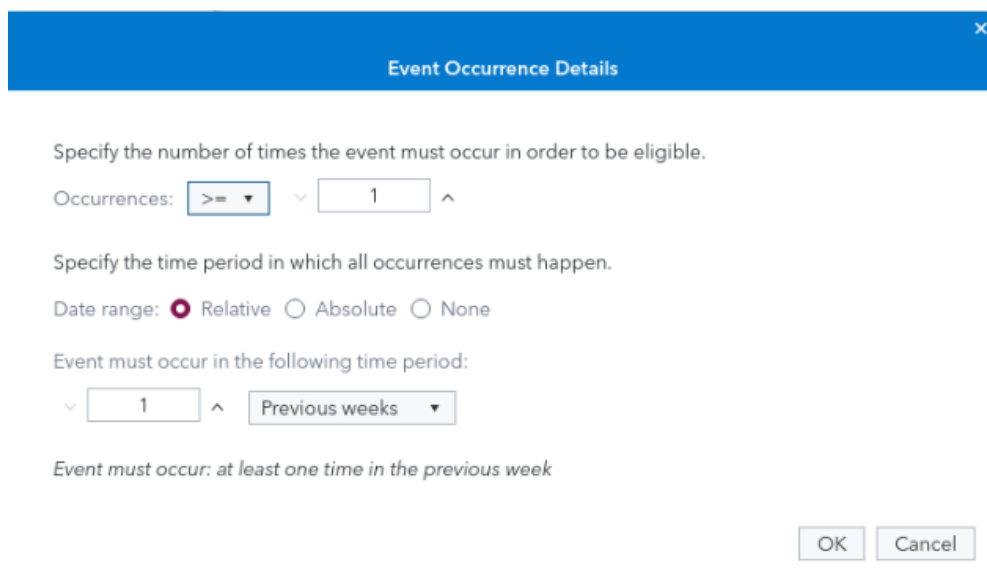
Here we will create a task, this connects the Spot and the Creative with Segmentation in order to conditionally fire the GA pixel on the client machine. We will use an event and information defined in the event, combined with merge tags to create the send information to GA servers

- a. From navigation bar, click **Task**
- b. Click the  button for create new Task
- c. Select **Web** as Task type
- d. Choose the **Spot** created from “Create Spot in CI360” section
- e. Go to **Targeting** Tab
 - i. Click the  button to add criteria
 - ii. Select the **Custom Event** which defined from Step #2



Name	Description
snzrl: Page View Event: ga: v1	

- iii. Optional: Define Even Occurrence Detail by click the  button:

Specify the number of times the event must occur in order to be eligible.

Occurrences: ^

Specify the time period in which all occurrences must happen.

Date range: ☒ Relative ☐ Absolute ☐ None

Event must occur in the following time period:

Previous weeks

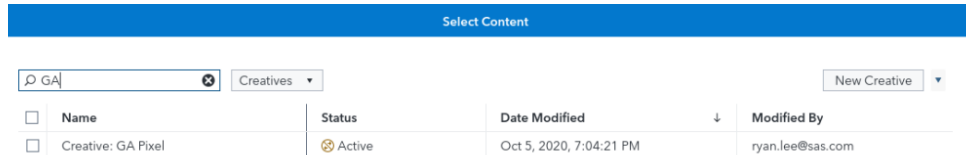
Event must occur: at least one time in the previous week


OK Cancel

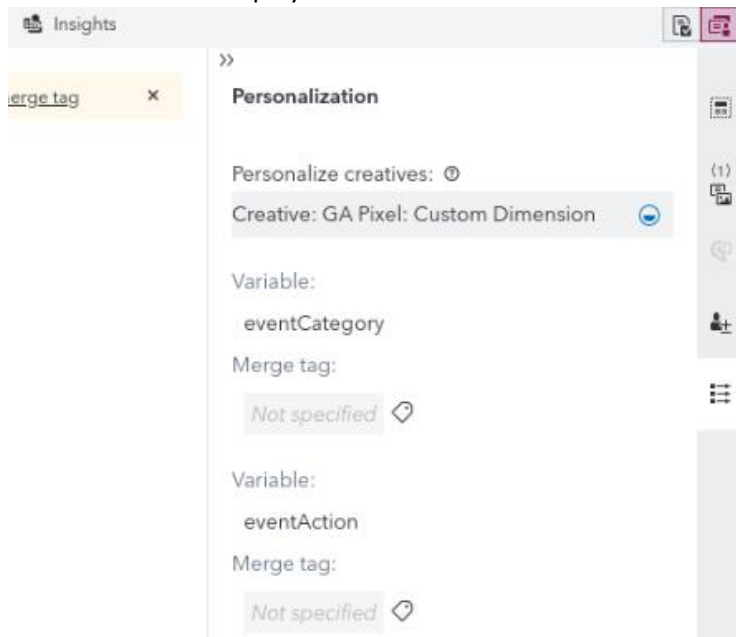
- iv. Or leave it by default as **once per pervious weeks**


- f. Go to **Content** Tab

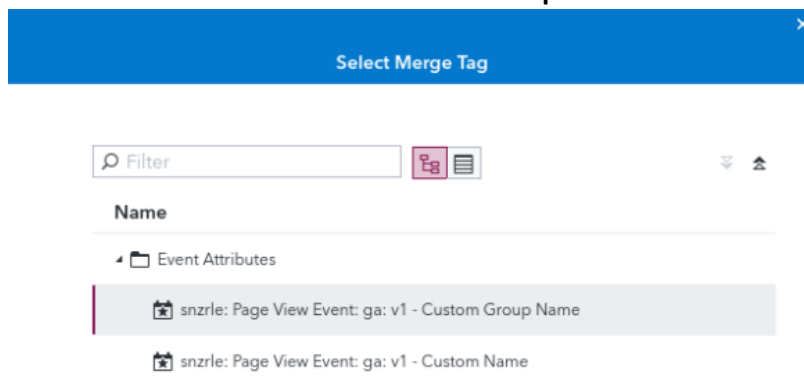
- i. Click **Select Content**
- ii. Choose **Creatives** from the drop-down menu
- iii. Select the **Creative** defined from “Create creative with GA pixel installed” section




- iv. Click **OK**
- v. Click the  button from the right-hand side menu bar
- vi. Screen should be displayed as below:

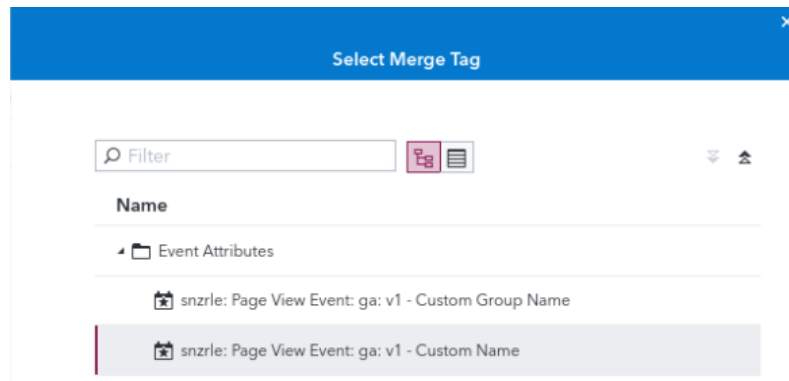


- vii. Click the  button from the **eventAction** merge tag field
- viii. Select the Event Attribute with **Custom Group Name**:

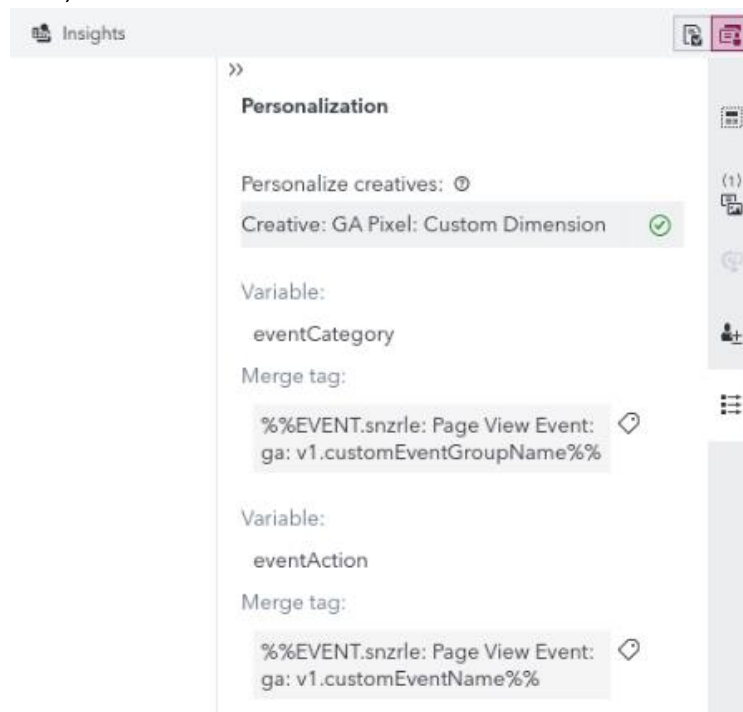



- ix. then, click **OK**
- x. Click the  button from the **eventCategory** merge tag field

- xi. Select the Event Attribute with **Custom Name**:



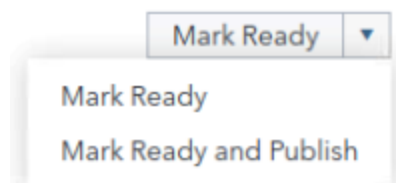
- xii. then, click **OK**.



- xiii. click the  button to save the task

- g. Go To **Orchestration** Tab

- i. Click on **Mark Ready and Publish** from drop-down menu:



- ii. Click **Yes** when promoted by **Publishing the item will change the status to Scheduled or Active**

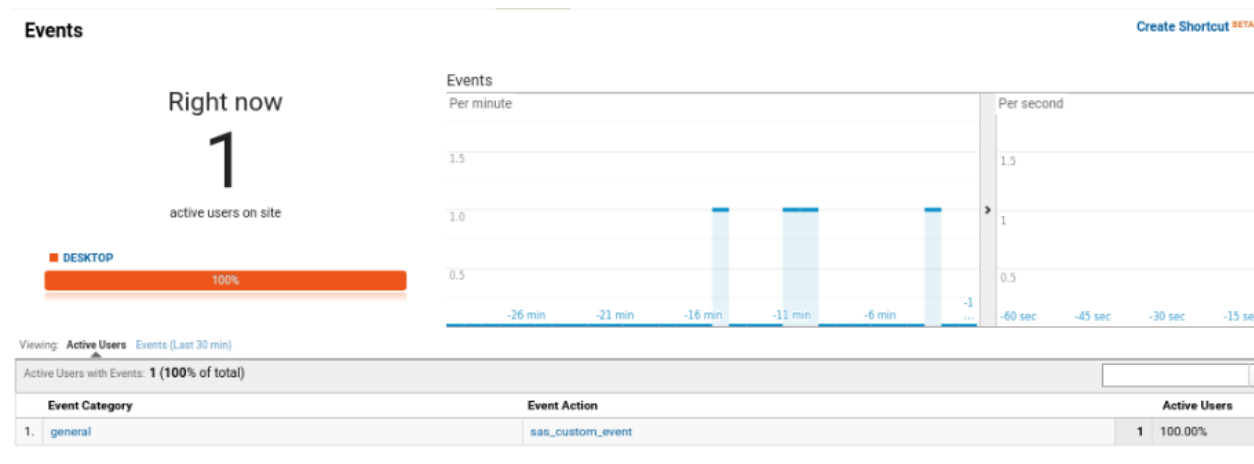
5.6 Trigger GA Pixel via SAS Creative

- Go to the page where your **Custom Event** is fired
- Then refresh the page or visit other page contain the **CI360 Pixel Placeholder**
- The “**Event: GA Pixel**” Custom Event should be now triggered the Google Pixel via SAS Creative, as below

```
<div id="sas_pixel" style="visibility: visible;">
  <data data-taskid="ba4be56d-0251-404d-9960-af7eeb6404df"></data>
  <data data-creativeid="d47bdc45-f0df-4886-812c-b0bd7cflc81c"></data>
  <data data-variant_id"></data>
  <meta content="text/html; charset=UTF-8" http-equiv="Content-Type">
  <title></title>
  <!-- Enter the HTML source code for your creative.      Note: If
  this creative contains a reference to an external resource
  such as an image, CSS file, or JavaScript file, then that resource
  must be publicly accessible. Otherwise, the creative cannot
  render properly. -->
  
</div>
```

5.7 Verification

Here we will verify if the Pixel is properly sending data to GA servers. Using the values we send to Google we will see if those values show up in the **Event Real Time Report** section of the GA interface.

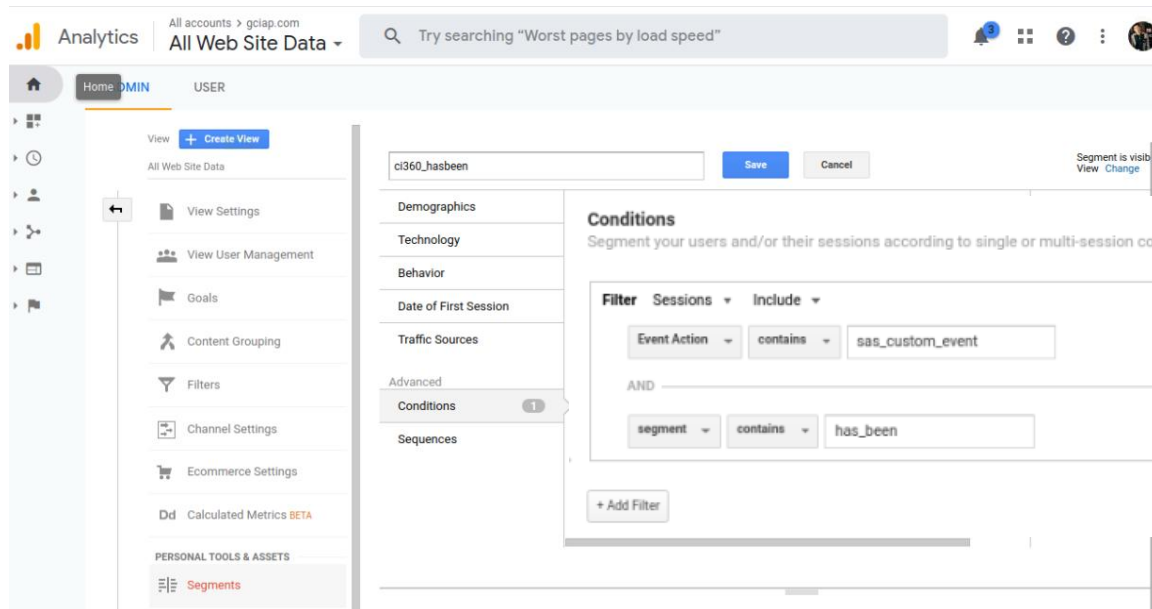


Once verified we can then use these in segments for activation on other platforms connected to GA

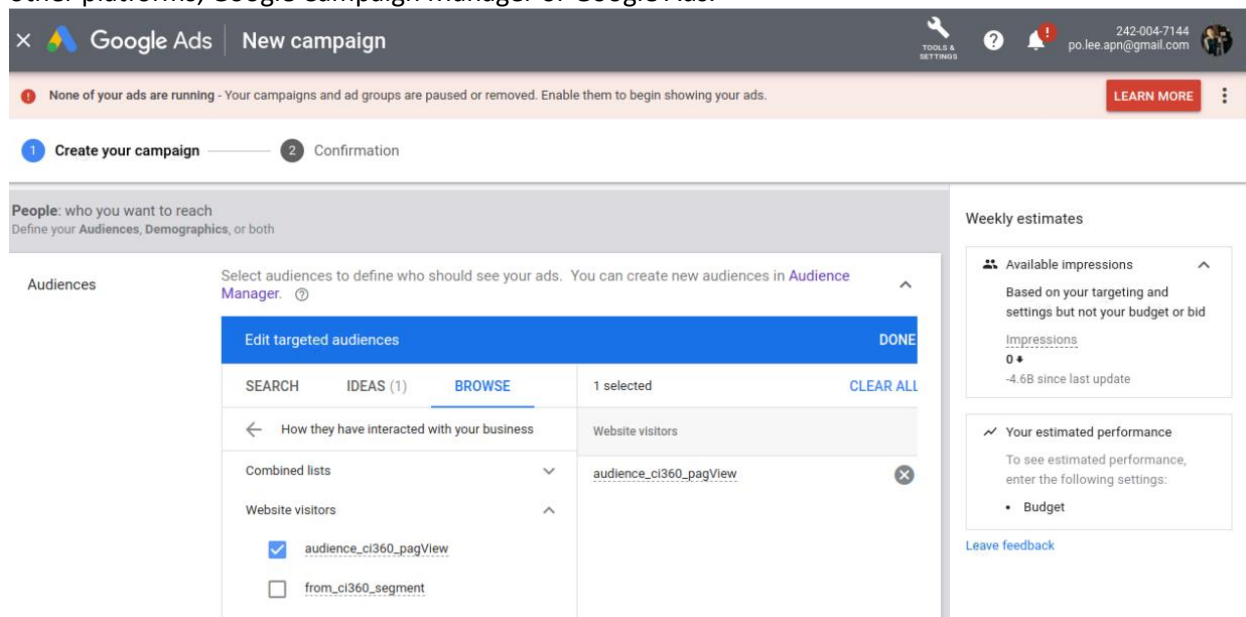
5.8 Example

5.8.1 EXAMPLE: USE GA EVENT IN OTHER PLATFORMS

In GA, go to Admin > Segments, it is where you can define the GA Segment as below.

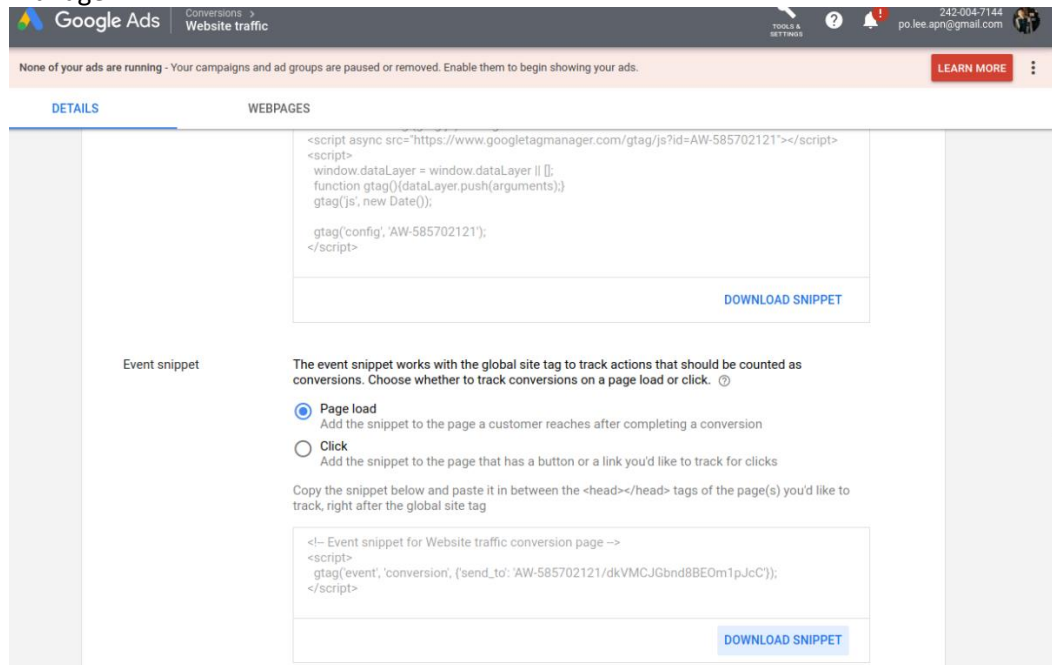



The GA Segment will be converting into “Audience” which will be the remarketing target criteria from other platforms, Google Campaign Manager or Google Ads.




5.8.2 EXAMPLE: USE GA- CONVERSION PIXEL VIA SAS

- a. Download the conversion snippet from the ad campaign in Google Ads / Google Campaign Manager

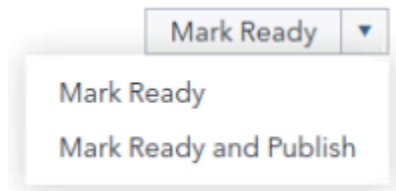


- b. From navigation bar, click **Creatives**
- c. Click the  button for create new Creative
- d. Select **HTML** as creative type
- e. Copy the **Google Conversion Pixel** into the body tag as below

```
<html>
  <head>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />
    <title></title>
  </head>
  <body>
    <!-- Enter the HTML source code for your creative.
    Note: If this creative contains a reference to an external resource
    such as an image, CSS file, or JavaScript file, then that resource must be
    publicly accessible. Otherwise, the creative cannot render properly. -->
    
  </body>
</html>
```

- f. Click **Done**
- g. click the  button to save the creative
- h. type in "**Creative: GA - Conversion Pixel**" then click **Save**
- i. Go To **Orchestration Tab**

- i. Click on **Mark Ready and Publish** from drop-down menu:



- ii. Click **Yes** when promoted by **Publishing the item will change the status to Scheduled or Active**

6 API (Server-to-Server)

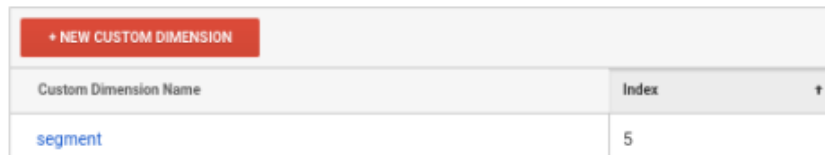
This section will show how to make a server-to-server connection using a CI360 Connector. This doesn't require any bridge code as it is a one-way communicator from the CI360 tenant to the GA servers. **This can also be proxied through on-premise agent if needed.**

6.1 Requirements

With the API integration, it cannot use GA pixel to fire event to the Google, alternatively we are going to use the **GA measurement protocol** to submit GA event through the API call.

6.1.1 Define Custom Dimension field(s) in GA to store SAS segment(s)

Create **Custom Dimensions** under **GA Admin > Custom Definition** section, these fields are going to store SAS segment (note that GA's Custom Dimension did not support multiple values in single field, if you're making several SAS segment to GA, it would be best practice to add more Custom Dimension with different field name).



Custom Dimension Name	Index
segment	5

In my example, custom dimension index 1 to 4 already in use for other purpose, hence it started with **Index 5**. And this index number is the key to map from the site code, please use the correct index number which mapped in your case.

6.1.2 Google Global site tag

Embed the gtag.js with the correct <product_id> (which provided by Google Analytics Account). And add "custom_map" property to set **custom dimension index** and **name** which created from last step accordingly, in my case, will be dimension5 and segment as below:

```
<!-- Global site tag (gtag.js) - Google Products -->
<script async src="https://www.googletagmanager.com/gtag/js">
</script>
<script>
window.dataLayer = window.dataLayer || [];
function gtag(){dataLayer.push(arguments);}
gtag('js', new Date());

gtag('config', '<product_id>', {
  custom_map: {
    dimension5: "segment"
  }
});
</script>
```

6.1.3 CI360 Tag Deployed




Add in the CI360 tag into the site. Please copy the code from your CI360 from the **General Settings > SAS Tag Instruction** section.

```

SAS Ci360 Tag Example:
<!-- SAS Ci360 Tag -->
<script>
(function(ci){
  var ef=window[ci]=function(){
    return ef.q.push(arguments);
  };
  ef.q=[];ef.a={};
})('ci360');
</script>
<script async data-efname='ci360' id='ob-script-async'
  a='<tenant ID>'
  src='https://<server>/js/ot-all.min.js'></script>

```



6.2 Connector setup

- a. Navigate to **General Settings > Connectors**
- b. Click on the  button
- c. Enter the details
 - i. Name: **Google GA**
 - ii. Description: ***Anything you want can go here to describe this***
 - iii. Support Contact: Set this up as anyone that would receive emails in case of errors
 - iv. Set as **Active** by changing the slider from  to 
 - v. Attach an image if you want (can find an image through Google for “Google Analytics” or otherwise)
 - vi. Click **Apply**

Google GA

Details ⓘ

Specify information to configure a connector.

Code:	CONN7
Name: *	<input type="text" value="Google GA"/>
Description:	<input type="text" value="For connecting directly with Google Analytics for use in external activation of users on ad platforms"/>
Type:	Custom
Support contact:	<input type="text" value="ryan.lee@sas.com"/>
Mark Active:	<input checked="" type="checkbox"/>
Associated image:	<div style="text-align: center;">  </div> <div style="text-align: right;">  Change Remove </div>

- d. Re-Open the Google Connector

e. Click **New Endpoint**

- i. Name: **Google GA**
- ii. Description: ***Anything you want can go here to describe this***
- iii. URL: `https://www.google-analytics.com/collect?v=1&tid=UA-8886598-2&t=event&cid={{$clientid}}&uid={{$userid}}&cd5={{$event}}&el=ci360_api&ea=sas_custom_event&ec=general`
(note the **cd5** highlighted in the URL, it is the custom dimension index if it is using different custom dimension index number in your GA account, please update this accordingly.)
- iv. Method: **GET**
- v. Time-out: **15 seconds**
- vi. Authorization: **n/a**

Google GA

Details Orchestration

▼ Details

Specify information to configure an endpoint. ⓘ

Code: CONN7-16

Name: * Google GA

Description: Google Analytics Event Tracking

URL: * `https://www.google-analytics.com/collect?v=1&tid=UA-8886598-2&t=event&cid={{$clientid}}&uid={{$userid}}&cd5={{$event}}&el=ci360_api&ea=sas_custom_event&ec=general` Define variable ⓘ

Method: GET ▼

Time-out: 15 seconds ▼

Connector: Google GA

> Authorization

> Headers

> Parameters

vii.

- f. Click on **define variables** (this will replace the {{variables}} in the URL with values from the JSON payload that will be sent each time the connector is invoked and these values will be populated from values we create in the custom event and task we create). Unfortunately at this time, there are no easy way to view the JSON payloads without setting up your own server to capture these values first but once you know their locations you can use JSON path selectors to define them (Check appendix for website to test your JSON payload and selectors) **Check to see if merge tags available from other.**
 - i. Set event to **"\$.customName"**
 - ii. Set clientid to **"\$.properties.ga_cid"**

- iii. Set userid to “\$.identityId”


Define variable

Key	Value *	Description
clientid	<input type="text" value="\$.properties.ga_cid"/>	<input type="text"/>
userid	<input type="text" value="\$.identityId"/>	<input type="text"/>
event	<input type="text" value="\$.customName"/>	<input type="text"/>

- iv. Click **OK**
 v. Click **Save** to save the endpoint
 vi. In the **Endpoints** section, click ☒ to enable the endpoint
 vii. Click **Apply**
 viii. You will now have a new connector as show below with 1 endpoint setup for Google Analytics

Connectors

Add and manage configurations for connectors. ⓘ


Filter 

<input type="checkbox"/>	Name	ID	Type	Active	Endp...	Date Modified
<input type="checkbox"/>	Google GA	CONN7	Custom	<input checked="" type="checkbox"/>		1 Oct 6, 2020

Optional: Setup the connector to proxy through an on-premise agent

6.3 Task Setup

Here we will create an external system task to invoke the connector using the event we created last step as a trigger. In the case of GA, all the information is sent through link parameters so we don't need any bridge code, but if needed we can use the content creation combined with merge tags to send additional information to the external system.

- a. Click on **Tasks** in the side panel
- b. Create a new task by clicking on 
- c. Choose **External System**
- d. Here you have the choice to select an Agent (even if you are using an on-premise agent to send your connector through, selecting an Agent is not required here) so simply click on

3

Select Connector Endpoints

 to skip this section
- e. Check the **Endpoint Name/Connector Name** we set before (Google GA/Google GA)
- f. Click **Create Task**

- g. Navigate to the **Orchestration Tab > Trigger** section and specify the Event you created in last step

The 'Select Criteria' dialog box is shown. It has a search bar with 'profile' entered. Below the search bar, there is a list of items: 'Customer Behavior' and 'Custom Event'. A table below the list shows one item: 'snzrl: Profile Event: ga: v1'.

- h. Click **Ok**
- i. Optional
- Add any other targeting criteria you want based on your requirements by navigating to the **Targeting** tab
 - Add a schedule making sure this task only fires during specific times

6.4 Verification

Here we will verify if the Pixel is properly sending data to GA servers. Using the values we send to Google we will see if those values show up in the **Custom Report** section of the GA interface. The Custom Report setting as per screen shot:

The screenshot shows the Google Analytics Custom Report configuration interface. The 'General Information' section has a title 'New Custom Report'. The 'Report Content' section has a report name 'Report Tab', type 'Flat Table', and dimensions 'segment', 'Event Action', 'Event Label', and 'Event Category'. The 'Metrics' section has 'Sessions' as the metric. The 'Filters' and 'Views' sections are optional and currently empty.

Once verified we can then use these in segments for activation on other platforms connected to Google.

Data Activation 3rd Party Integration

Analytics | All accounts > gciap.com | All Web Site Data

Try searching for "acquisition overview"

Home

Customization

Dashboards

Custom Reports

Saved Reports

Custom Alerts

REPORTS

Realtime

Audience

Acquisition

Behavior

Conversions

New Custom Report

SAVE EXPORT SHARE EDIT

All Users 100.00% Sessions

+ Add Segment

Oct 25, 2020 - Oct 25, 2020

Report Tab

Advanced Filter ON

segment	Event Action	Event Label	Event Category	Sessions
1. has_been	sas_custom_event	ci360_api	general	1 (100.00%)

Show rows: 10 Go to: 1 1 - 1 of 1

This report was generated on 10/25/20 at 12:44:03 PM - Refresh Report

7 Appendix

7.1 External References

Item	Link
<i>Sending Data to DCS (Pixel/API)</i>	https://developers.google.com/analytics/devguides/collection/gtagjs/sending-data
<i>Define CSV for GA</i>	https://support.google.com/analytics/answer/3191417?hl=en
<i>JSON Selector (API/Connectors)</i>	http://jsonpath.com/

7.2 Source Code

7.2.1 Google Tag Example:

Note: You do need to replace “<product_id>”, “<cd_index>” and “<cd_name>” part of the code!

```
<html>

<head>
  <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" /
  <title></title>
</head>

<body>
  <!-- Global site tag (gtag.js) - Google Products -->
  <script async src="https://www.googletagmanager.com/gtag/js"></script>
  <script>
    window.dataLayer = window.dataLayer || [];
    function gtag(){dataLayer.push(arguments);}
    gtag('js', new Date());

    gtag('config', '<product_id>', {
      custom_map: {
        dimension<cd_index>: "<cd_name>"
      }
    });
  </script>
</body>
</html>
```

7.2.2 Google API Endpoint

```
https://www.google-analytics.com/collect?v=1&tid=UA-8886598-
2&t=event&cid={{${clientid}}}&uid={{${userid}}}&cd5={{${event}}}&el=ci360_api&ea=sas_custom_event&ec=general
```

- i. Set event to “\$.customName”
- ii. Set clientid to “\$.properties.ga_cid”
- iii. Set userid to “\$.identityId”

7.2.3 SAS CI360 Tag Example Code

Note: This is for examples use only, please copy the code from your CI360 from the General Settings > SAS Tag Instruction section

SAS CI360 Tag Example:

```
<!-- SAS CI360 Tag -->
<script>
(function(ci){
  var ef=window[ci]=function(){
    return ef.q.push(arguments);
  };
  ef.q=[];ef.a={};
})('ci360');
</script>
<script async data-efname='ci360' id='ob-script-async'
  a='<tenant ID>'
  src='https://<server>/js/ot-all.min.js'></script>
```

7.2.4 SAS Generic Pixel Placeholder

For generic spot creation

```
<!-- SAS Pixel Placeholder -->
<div id="sas_pixel"></div>
```

7.2.5 SAS Google Placeholder

For Google spot creation

```
<!-- SAS Pixel Placeholder -->
<div id="sas_google_pixel"></div>
```

7.3 Connector JSON Payloads

Below is a cut down version of the json payload for reference, you can use <http://www.jsonpath.com/> to find the value you want using JSON selector notation

7.3.1.1 Custom Event

Example JSON Selectors

Selector	Value (using the JSON example below)
<code>\$.guid</code>	d4a1b51b-cc9b-4b25-8d76-1b5021625a5a
<code>\$.eventName</code>	Outgoing_1
<code>\$.customName</code>	has_been
<code>\$.customGroupName</code>	segment
<code>\$.sessionID</code>	e105513fe064112b551dccc3
<code>\$.channelType</code>	external
<code>\$.date.generatedTimestamp</code>	1597647116198
<code>\$.externalTenantId</code>	Abcdef0123456789
<code>\$.internalTenantId</code>	xxxx
<code>\$.identityId</code>	2d25b302-6ee9-35fc-ba41-8c3033f90266
<code>\$.visitId</code>	930981fa7e3a4c58623bc80d
<code>\$.properties.ga_cid</code>	1832800649.1603260950
<code>\$.properties.externalCode</code>	TSK_104
<code>\$.identity.identityId</code>	2d25b302-6ee9-35fc-ba41-8c3033f90266
<code>\$.identity.sessionId</code>	e105513fe064112b551dccc3

Ex. Using the date value below in a connector variable you would use `$.date.generatedTimestamp` and the value returned would be 1597647116198

7.3.1.2 Example JSON Payload

It is highly recommended that you setup a server to receive the JSON payloads in order to see any differences and confirm the code is correct. This should be used for reference only!

```
{
  "guid": "d4a1b51b-cc9b-4b25-8d76-1b5021625a5a",
  "apiEventKey": null,
  "eventDesignedId": "d780a02f-2e96-4e46-a80a-9885e23bb4ec",
  "eventDesignedName": "event", //Defined in the external task under Orchestration > Outgoing Information >
  Event Name
  "eventName": "Outgoing_1", //Defined in the external task under Orchestration > Outgoing Information > Event
  Name
  "customName": "has_been", //Custom Name defined in the trigger event (custom event) under Event Details >
  Event Attributes > Custom Name > Value text
}
```

```
"eventType": "outboundSystem", //Defines which kind of task, external tasks are outboundSystem tasks, events
would be customEvent etc.
"sessionId": "e105513fe064112b551dccc3", //Defines the sessionId of the current user
"channelId": "6ffeb2f866cb3044264262ab",
"channelType": "external", //Defines the channelType
"ipAddress": null,
"date": {
  "generatedTimestamp": 1597647116198, //Timestamp in Unix Epoch format
},
"externalTenantId": "abcdef0123456789", //The external Tenant ID (used in the tag on the site)
"internalTenantId": xxxx, //The internal Tenant ID
"identityId": "2d25b302-6ee9-35fc-ba41-8c3033f90266", //The datahub ID used to define a user
"page": {
  "loadId": "9df574667a3a4c587dca6ec7",
  "viewSequenceNum": 0,
},
"visitId": "930981fa7e3a4c58623bc80d", //The visitID for the user associated with this event
"visitorGroup": null,
"visitorState": "returning",
"visitSequenceNum": null
},
"properties": { //This section contains most of the custom attributes and custom properties
  "ga_cid": "1832800649.1603260950",
  "parent_eventname": "load",
  "externalCode": "TSK_104", //The external code defined in the external task under Properties
  "event_datetime_utc": "1597647116198", //Event time in Unix Epoch
  "parent_event": "customEvent",
  "contributing_guid_1": "a217d4c4-34f9-460a-afb7-0b7ade117c7e"
},
"identity": {
  "identityId": "2d25b302-6ee9-35fc-ba41-8c3033f90266",
  "identityType": null,
  "identitySource": null,
  "identityEventName": null,
  "identityAttribute": null,
  "identityAssociation": null,
  "userId": null,
  "visitId": null,
  "ipAddress": null,
  "sessionId": "e105513fe064112b551dccc3",
  "visitorId": null,
  "loginEventType": null,
  "loginType": null,
  "loginValue": null
},
"customGroupName": "segment",
"extendedCustomEventWithRevenueFlag": false,
"parentEventUid": "9155c61d-dd69-4fbb-9b49-c6f73490b900"
}
```