Eckoh

Google Analytics Implementation -Version: 1.0-

1. Introduction

Eckoh Custom Google Analytics ("eckoh_ga") is a custom implementation of the well known **Google Universal Analytics** that, by using the developer documented methods[1], it allows users to send data to a "Google Analytics" backend for a specific project.

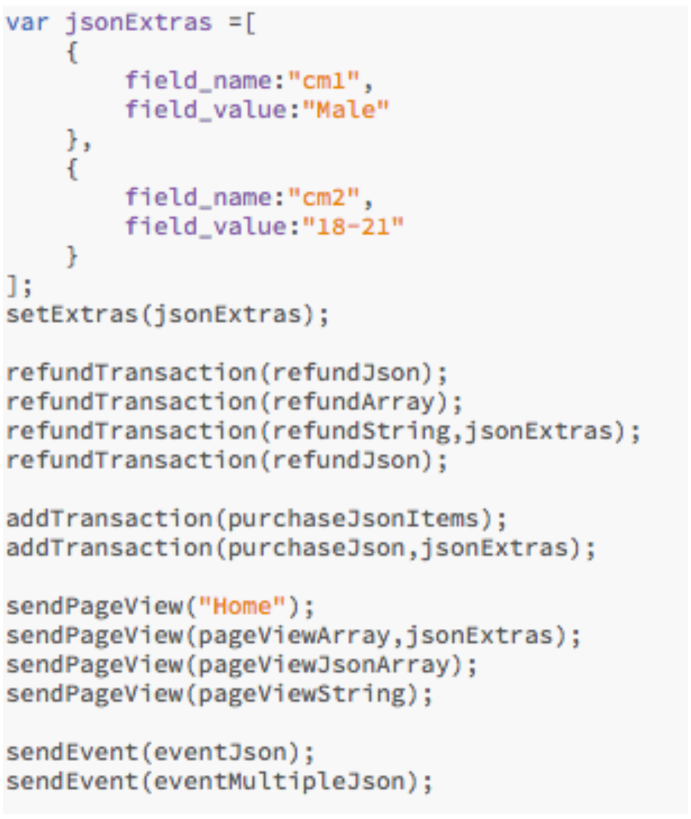
The idea for this project came with the concern of using 3rd parties libraries and how secure they are when the information is shared across the Internet. As a solution payment company, Eckoh brings this library that brings control about what is share securely. Furthermore, the library can grow up according to the business needs to one way or the other.

Before start reading this document, it's strongly recommended to be familiarised with Google analytics platform as the library goes from basic to advanced features and it does the same and more using the same style.

1.1. Hits: The structure of the data

Google Analytics is build by using "Hits" which contain the information for tracking.

The library send hits by using functions as "analytics.js" does. The functions supported are explained throughout this document. See a summary here:



```
var jsonExtras =[
    {
        field_name:"cm1",
        field_value:"Male"
    },
    {
        field_name:"cm2",
        field_value:"18-21"
    }
];
setExtras(jsonExtras);

refundTransaction(refundJson);
refundTransaction(refundArray);
refundTransaction(refundString,jsonExtras);
refundTransaction(refundJson);

addTransaction(purchaseJsonItems);
addTransaction(purchaseJson,jsonExtras);

sendPageView("Home");
sendPageView(pageViewArray,jsonExtras);
sendPageView(pageViewJsonArray);
sendPageView(pageViewString);

sendEvent(eventJson);
sendEvent(eventMultipleJson);
```

[1] https://developers.google.com/analytics/devguides/collection/protocol/v1/devguide

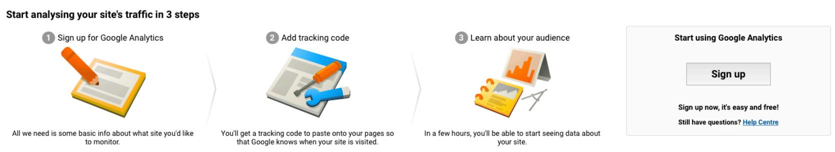
1.2. Quick start

1.2.1. Set up Google Analytics

The first thing needed for tracking a Google Analytics project is the "Project tracking ID". It looks like UA-XXXXXXXX-X.

If the project does not exist:

- Visit https://analytics.google.com/analytics/web/



Start analysing your site's traffic in 3 steps

1 Sign up for Google Analytics — All we need is some basic info about what site you'd like to monitor.

2 Add tracking code — You'll get a tracking code to paste onto your pages so that Google knows when your site is visited.

3 Learn about your audience — In a few hours, you'll be able to start seeing data about your site.

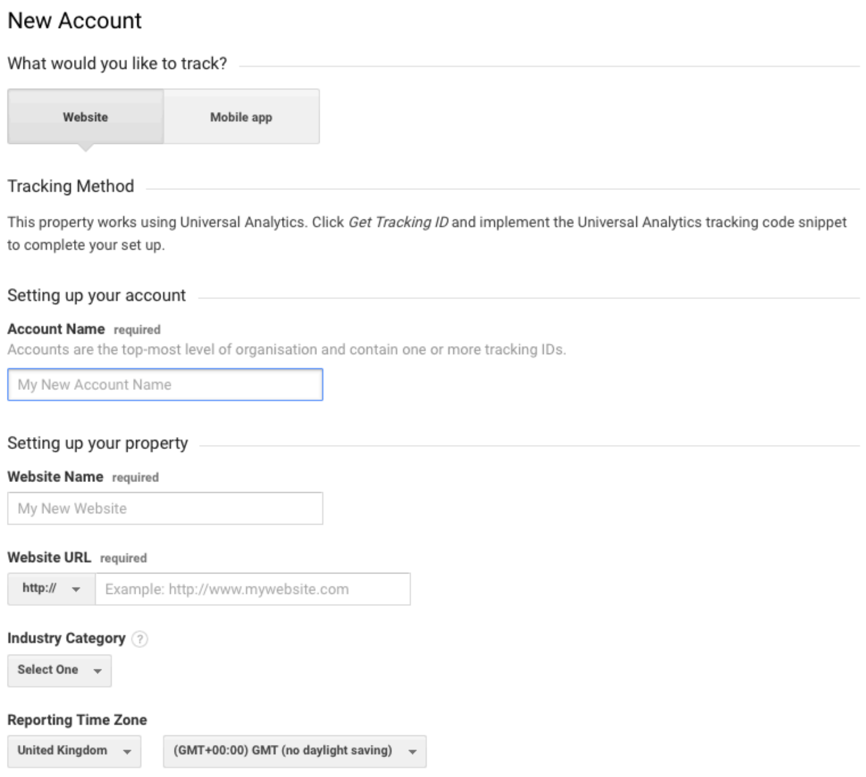
Start using Google Analytics

Sign up

Sign up now, it's easy and free!

Still have questions? Help Centre

- Provide de data required and press submit.



New Account

What would you like to track?

Website | Mobile app

Tracking Method

This property works using Universal Analytics. Click *Get Tracking ID* and implement the Universal Analytics tracking code snippet to complete your set up.

Setting up your account

Account Name required
Accounts are the top-most level of organisation and contain one or more tracking IDs.

My New Account Name

Setting up your property

Website Name required

My New Website

Website URL required

http:// | Example: http://www.mywebsite.com

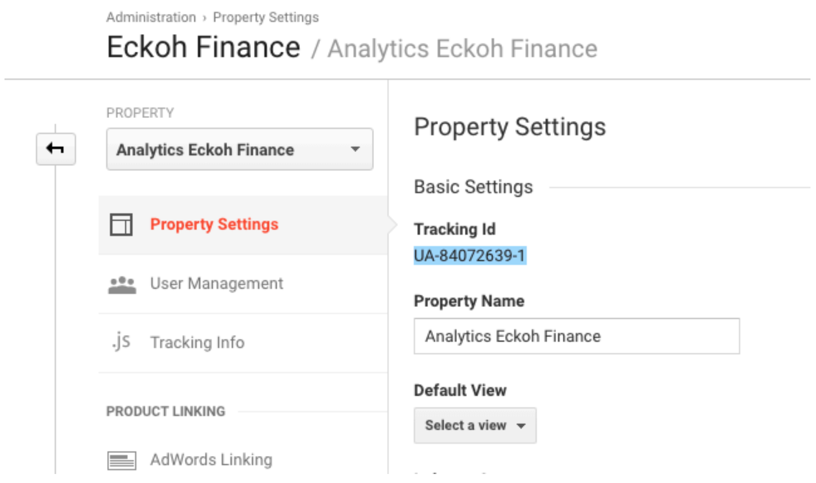
Industry Category ?

Select One

Reporting Time Zone

United Kingdom | (GMT+00:00) GMT (no daylight saving)

- Under Administration/Property settings the tracking id can be found.



Administration › Property Settings

Eckoh Finance / Analytics Eckoh Finance

PROPERTY

Analytics Eckoh Finance

Property Settings

Basic Settings

Tracking Id
UA-84072639-1

Property Name
Analytics Eckoh Finance

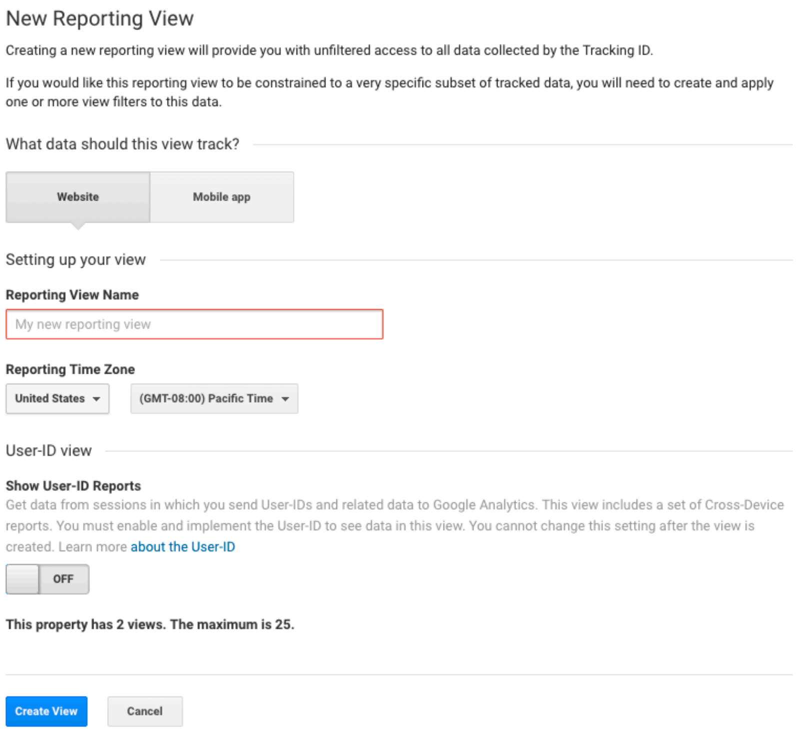
Default View
Select a view

1.2.2. Importing and declaring the library

By including the following declaration into the header file, the library is imported. A more extended explanation is going to be shown during the document.

```
<script data-tid="UA-86314412-1" data-ga="_ga" data-uid="ABCD1234" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```

1. Data-tid: Project tracking id.
2. Data-ga: name for the Google analytics CID Cookie. By default "`_ga`". Because the name can be changed, if so, the custom name needs to be provided. Other ways the customer journey will be tracked into 2 different CID (Consistency on tracking). (Optional).
3. Data-uid: As specified in Google analytics developer documentation, UID's identifies a customer cross platform. If the tracking has been implemented, the UID can be also provided. (Optional).
    a. For a UID tracking, a view needs to be created and **`User-ID reports`** needs to be enabled.



New Reporting View

Creating a new reporting view will provide you with unfiltered access to all data collected by the Tracking ID.

If you would like this reporting view to be constrained to a very specific subset of tracked data, you will need to create and apply one or more view filters to this data.

What data should this view track?

Website | Mobile app

Setting up your view

Reporting View Name

My new reporting view

Reporting Time Zone

United States (GMT-08:00) Pacific Time

User-ID view

Show User-ID Reports

Get data from sessions in which you send User-IDs and related data to Google Analytics. This view includes a set of Cross-Device reports. You must enable and implement the User-ID to see data in this view. You cannot change this setting after the view is created. Learn more about the User-ID

OFF

This property has 2 views. The maximum is 25.

4. Src: Location for the library.

1.2.3. Start tracking

If all the data has been provided, by calling the following functions, the tracking will be set up and a "pageview" will be send.

```
<script>
    setTracking(projectTrackingID,Cookie.Get("cid"),userTrackingId);
    sendPageView("Home");
</script>
```

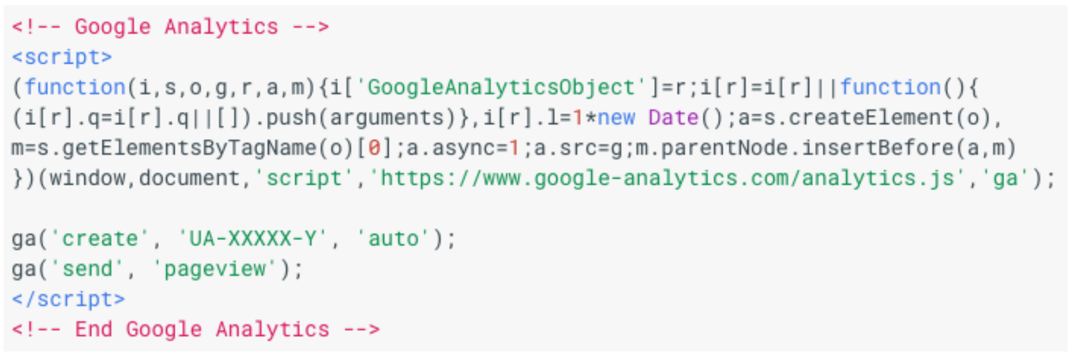
2. Declaration, usage and structure

In this section it is going to be shown how both analytics.js (Google) and eckoh_ga.js (Eckoh) are declared and what they need to be initialised. Also, differences and similarities are going to be described for both.

2.1. Analytics.js and ecoh_ga.js

2.1.1. Analytics.js

For implementing Google Analytics, the library "analytics.js " needs to be imported. Also, when the library is declared it stores a cookie into the browser for tracking customer ID's (explained later).



```
<!-- Google Analytics -->
<script>
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-XXXXX-Y', 'auto');
ga('send', 'pageview');
</script>
<!-- End Google Analytics -->
```

The image above does the following processes:

1. Declaration and initialisation for the library.
2. Setting up the **unique project tracking id** ("UA-XXXXX-Y") and the **customer-tracking id** (auto means from the cookie).
3. Finally it sends a "pageview" hit to Google analytics backend. It is not until this point that there is a connection between the site (or app) and the backend.

Finally, it is important to know how the actions are called:

Ga('create',…) is calling a function called "create" inside the analytics.js. ga(…) is used as a convention so it is simpler to distinguish what is coming from the analytics.js implementation.

2.1.2. Eckoh_ga.js

On the other hand, eckoh_ga.js is intended to simulate analytics.js functionalities and make it, if possible, better and simpler.

A good example of its simplicity is its own declaration process (assuming it is located into the described route).

```
<script data-url="USER_API_URL" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```

1. By declaring the library, all the first step process from analytics.js has been achieved with only one line of code. Once it is declared the library does:
   1) Imports the library and all the code needed (No 3rd party libraries).
   2) Creates a custom cookie handler and:
      a. Checks if "cid" variable has been already defined in the browser.
      b. If it does not exist, if creates one using an 8 digits hexadecimal random number with and expiry date of one year.
      c. If it does exist, It renews the expiry date automatically for another year from the last time the page has been refresh.
   3) Stores the URL for the user-tracking id.
      a. If it is not available, it can be not provided.
      b. Not shown in analytics.js but this part also needs to be on client site.

```
setTracking("UA-86314412-1",Cookie.Get("cid"),"");
```

2. Like in analytics.js, the project tracking id, the customer id (CID) and, optionally, the user id (UID) need to be provided.
   a. PID: Unique identifier for a project needed for Google for tracking. The code can't be shared across platform but it can be modified. For instance, for an app the code would look like UA-86314412-2.
   b. CID: Unique identifier for a specific browser or device. It is stored as a cookie. Google is able to track different clients assuming they always use the same device and browser.
   c. UID: Optionally, Google analytics is capable to track user cross platform. The way this is achieved is by assigning a user id to a user and it has to be managed by the organisation using an API. There are many ways on how user id can be identified and tracked. In the next section the user API will be explained.

```
sendPageView("Home");
```

3. Sending hit events is similar than in analytics. In this case the library is sending a "pageview" hit for the home page. What and how the library support hit functions will be explained later.

2.2. User ID API

As mentioned before, Google analytics can track user cross platforms. A custom User ID tracking needs to be implemented by the organisation. The library can resolve the problem by using 2 different methods:

2.2.1. Custom API

If wished the user id tracking control can be achieved using alternatives ways. At the end, the following function needs to be called and the UID might or might not be provided.

```
setTrackingUn("UA-86314412-1",Cookie.Get("cid"),"ABCD1234");
```

For a best practice:

- Call the function with a valid UID after a Login process.
- Do not store the UID value into a cookie.

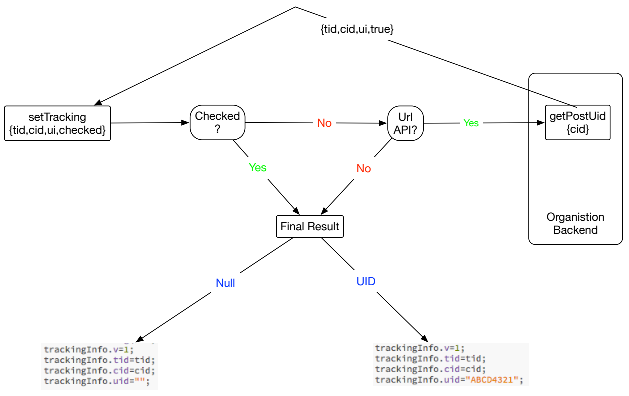
2.2.2. Eckoh_ga API addition

On the other hand, eckoh_ga offers a function for implementing with the following tools.

- A separate new table with `cid_ga` as varchar(24) as primary key and `uid_ga` as varchar(DEPEND).
- ManageGAUid.php:
- getPostUid function:
    - IN: post request with a field name `cid`;
    - OUT: JSON Object with a field "uid_ga" on the top level.
- addUidWithCid function: Checks if the pair value exists on the table. If not they got added. If exist but `uid` is different, it gets updated
    - IN: post request with a field name `cid` and a field name `uid`;
    - OUT: JSON Object with a field "uid_ga" and field value with the action performed on the top level.

So when the function below is called, that is what happens:

```
setTracking("UA-86314412-1",Cookie.Get("cid"),"");
```



```
trackingInfo.v=1;
trackingInfo.tid=tid;
trackingInfo.cid=cid;
trackingInfo.uid="";
```

```
trackingInfo.v=1;
trackingInfo.tid=tid;
trackingInfo.cid=cid;
trackingInfo.uid="ABCD4321";
```

2.3. Declaration methods

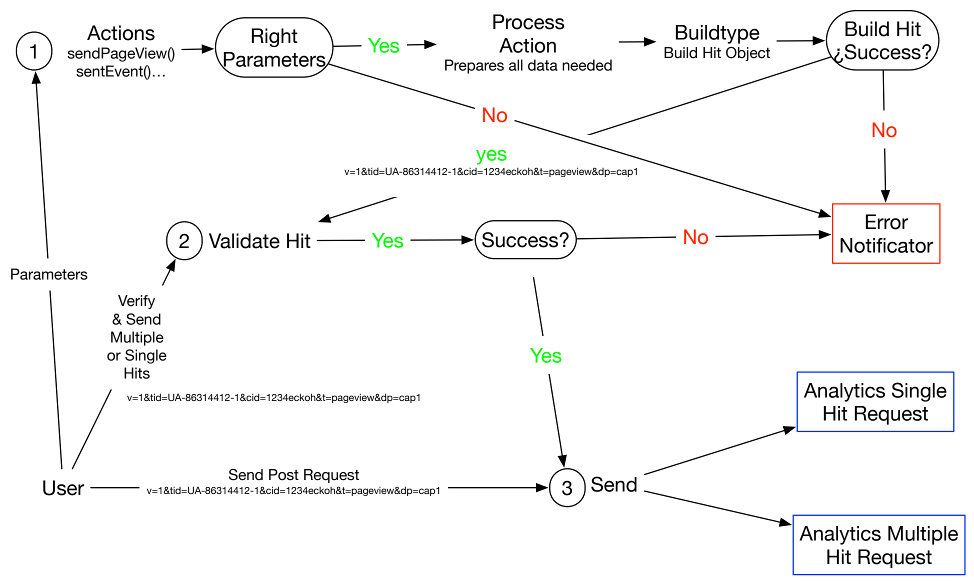
The library "eckoh_ga.js" can be called using different as multiple values can be sent. During the explanation all the variables are going to be covered.

1. Basic: only the location of the library is provided. At this point the library will manage the cookie as default with no user identification.
```
<script src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```
2. User identification URL manager: The URL for the user identification API is provided. It has to be on the implementation site.
```
<script data-url="USER_API_URL" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```
3. Custom _GA cookie: If at some point the cookie for storing the CID for Google analytics has been changed, it can be provided and the library will find and mange the cookie.
```
<script data-url="USER_API_URL" data-ga="_ga" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```
4. User tracking id: If a user has been tracked, this value can also be send.
```
<script data-url="USER_API_URL" data-ga="_ga" data-uid="ABCD1234" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```

These values can be mix and match according to the necessities. It is not important if these calls are not understood right now. At end of this document all of them are going to be covered.

2.4. Structure

The library is designed to be a multiple difficult step library with a common error tracker notification point.



1
Actions
sendPageView()
sentEvent()...
Right Parameters
Yes
Process Action
Prepares all data needed
Buildtype
Build Hit Object
Build Hit ¿Success?
No
No
yes
v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=cap1
2
Validate Hit
Yes
Success?
No
Error Notificator
Parameters
Verify & Send Multiple or Single Hits
v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=cap1
Yes
Analytics Single Hit Request
User
Send Post Request
v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=cap1
3
Send
Analytics Multiple Hit Request

As described, the process has 3 different starting points:

1. By calling a "hit function" with the right parameters, the library identifies multiple or single hit operations, it builds the request, checks if the

construction is correct and sends it to the appropriate backend. If any of these steps fail, a tracked error message will be generated.

2. If the correct format of the hit or hits is known, the library can check and send the request to the appropriate backend.

   ```
   validateTheHit(params,single)
   ```

   - The "params" field has to be the one that Google Analytics uses for POST request (see https://developers.google.com/analytics/devguides/collection/protocol/v1/devguide#enhancedecom).
   - At this point the library needs to know if it is a single or multiple hit request.
   - If the pre validation fails, a tracked error message will be generated.
   - If everything is correct, it sends the hit request to Google Analytics backend.

3. Also, if only the hit request wants to be sending without any kind of pre validation, the 3rd option can be used (Not Recommended).

   ```
   sendPostRequest(params,single);
   ```

   Once again we need to specify weather or not it's single or not.

3. Basic type of hits supported

3.1. Page tracking

Page tracking (also known as "pageview") is the simplest and most basic hit in Google analytics. The fields for a correct hit structure are:

```
v=1              // Version.
&tid=UA-XXXXX-Y  // Tracking ID / Property ID.
&cid=555         // Anonymous Client ID.

&t=pageview      // Pageview hit type.
&dh=mydemo.com   // Document hostname.
&dp=/home        // Page.
&dt=homepage     // Title.
```

Eckoh_GA.js:

```
sendPageView(pageViewString);
sendPageView(pageViewArray);
sendPageView(pageViewJson);
sendPageView(pageViewJsonArray);
```

There are 4 different ways for registering a "pageview" hit:

1. Page name given: tracks a page with a page name given.

   ```
   var pageViewString = "home";
   ```

2. List of page name: A multiple hit event can be send with a list of page name

   ```
   var pageViewArray = ["home", "login"];
   ```

3. Full-page tracking: For more accuracy of the tracking, the function can receive a JSON object with all the fields.

   ```
   var pageViewJson = {
       page_host : "utilities.com",
       page_name : "home",
       page_title : "Welcome"
   };
   ```

4. Full-page tracking list: List of JSON objects also available.

   ```
   var pageViewJsonArray = [
       {
           page_host : "utilities.com",
           page_name : "home",
           page_title : "Welcome"
       },
       {
           page_host : "utilities.com",
           page_name : "login",
           page_title : "Acces"
       }
   ];
   ```

Error tracking:

- Any other structure will cause a tracked exception error.
- If "page_name" is not provided, it will fail the pre validation step and a tracked exception error will be generated.

Output:

1.
   ```
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=home
   ```
2.
   ```
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=home
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=login
   ```
3.
   ```
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dh=utilities.com&dp=home&dt=Welcome
   ```
4.
   ```
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=event&dh=utilities.com&dp=home&dt=Welcome
   v=1&tid=UA-86314412-1&cid=1234eckoh&t=event&dh=utilities.com&dp=login&dt=Acces
   ```

3.2. Events tracking

The other kind of basic hit is "events". The fields for a correct hit structure are:

```
v=1                 // Version.
&tid=UA-XXXXX-Y     // Tracking ID / Property ID.
&cid=555            // Anonymous Client ID.

&t=event            // Event hit type
&ec=video           // Event Category. Required.
&ea=play            // Event Action. Required.
&el=holiday         // Event label.
&ev=300             // Event value.
```

Eckoh_GA.js

```
sendEvent(eventJson);
sendEvent(eventMultipleJson);
```

In this case multiple and single request are allowed with the following format.

1. Single hit event:

```
var eventJson = {
    event_category : "Test",
    event_action : "The Action",
    event_label : "Action for testing",
    event_value : 1
}
```

2. Multiple hit events:

```
var eventMultipleJson = [
    {
        event_category : "Test",
        event_action : "The Action",
        event_label : "Action for testing",
        event_value : 1
    },
    {
        event_category : "Test",
        event_action : "The Second Action",
        event_label : "Another action for testing",
        event_value : 1
    }
];
```

Error tracking:

- Any other structure will cause a tracked exception error.
- If "event_category" and "event_action" are not provided, it will fail the pre validation step and a tracked exception error will be generated.
- The other fields can be not provided (Optional).

Output:

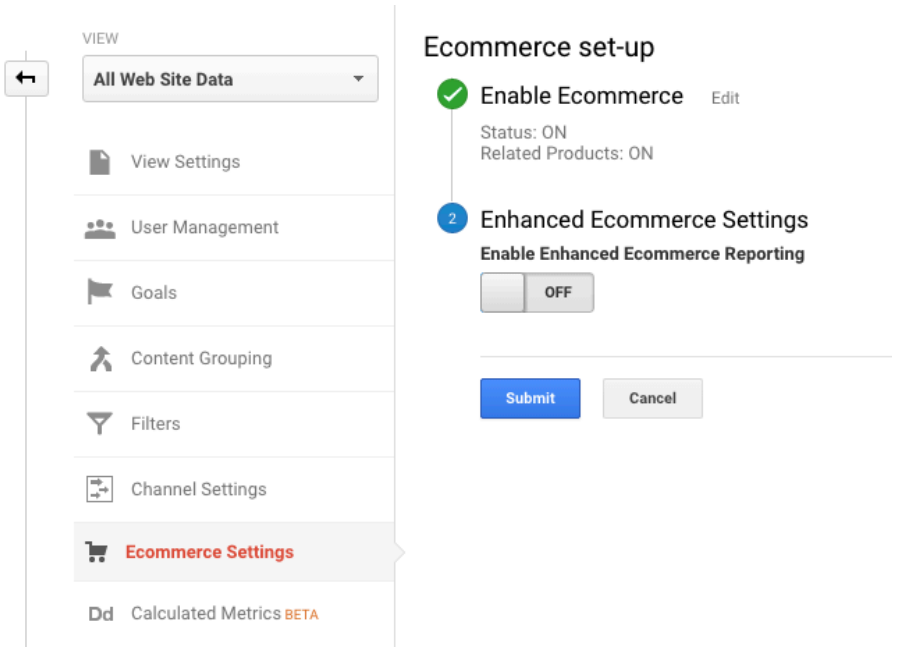
-
  ```
  v=1&tid=UA-86314412-1&cid=1234eckoh&t=event&ec=Test&ea=The Action&el=Action for testing&ev=1
  ```

-
  ```
  v=1&tid=UA-86314412-1&cid=1234eckoh&t=event&ec=Test&ea=The Action&el=Action for testing&ev=1
  v=1&tid=UA-86314412-1&cid=1234eckoh&t=event&ec=Test&ea=The Second Action&el=Another action for testing&ev=1
  ```

4. Ecommerce Tracking

Ecommerce tracking allows organisation to track all the monetary transactions that go through the web site and mobile apps.

Before using ecommerce in the project, it needs to be enabled by going into your project view setting.



VIEW
All Web Site Data

View Settings
User Management
Goals
Content Grouping
Filters
Channel Settings
Ecommerce Settings
Calculated Metrics BETA

Ecommerce set-up

Enable Ecommerce Edit

Status: ON
Related Products: ON

Enhanced Ecommerce Settings

Enable Enhanced Ecommerce Reporting

OFF

Submit Cancel

4.1. Purchases

Google Analytics allows tracking all the purchases that the web site and the app are processing. A purchase transaction can include different items and it is sent using a "pageview" hit.

Here there are the fields needed to be send within the link:

```
v=1                                     // Version.
&tid=UA-XXXXX-Y                         // Tracking ID / Property ID.
&cid=555                                // Anonymous Client ID.
&t=pageview                             // Pageview hit type.
&dh=mydemo.com                          // Document hostname.
&dp=/receipt                            // Page.
&dt=Receipt%20Page                      // Title.

&ti=T12345                              // Transaction ID. Required.
&ta=Google%20Store%20-%20Online         // Affiliation.
&tr=37.39                               // Revenue.
&tt=2.85                                // Tax.
&ts=5.34                                // Shipping.
&tcc=SUMMER2013                         // Transaction coupon.

&pa=purchase                            // Product action (purchase). Required.
&pr1id=P12345                           // Product 1 ID. Either ID or name must be set.
&pr1nm=Android%20Warhol%20T-Shirt       // Product 1 name. Either ID or name must be set.
&pr1ca=Apparel                          // Product 1 category.
&pr1br=Google                           // Product 1 brand.
&pr1va=Black                            // Product 1 variant.
&pr1ps=1                                // Product 1 position.
```

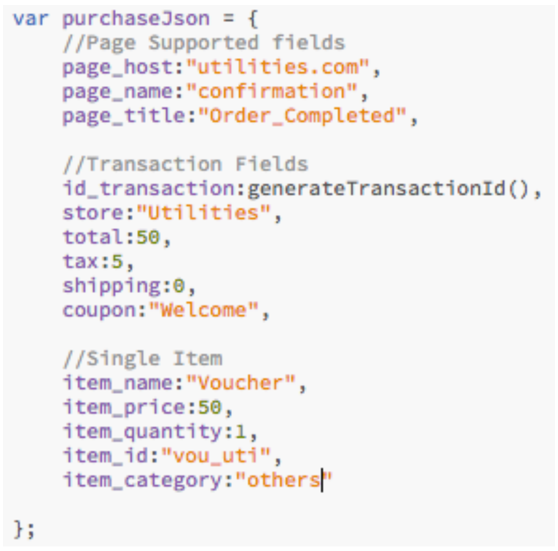
Eckoh_GA.js

In this case <u>only single</u> requests are allowed.

```
addTransaction(purchaseJson);
```

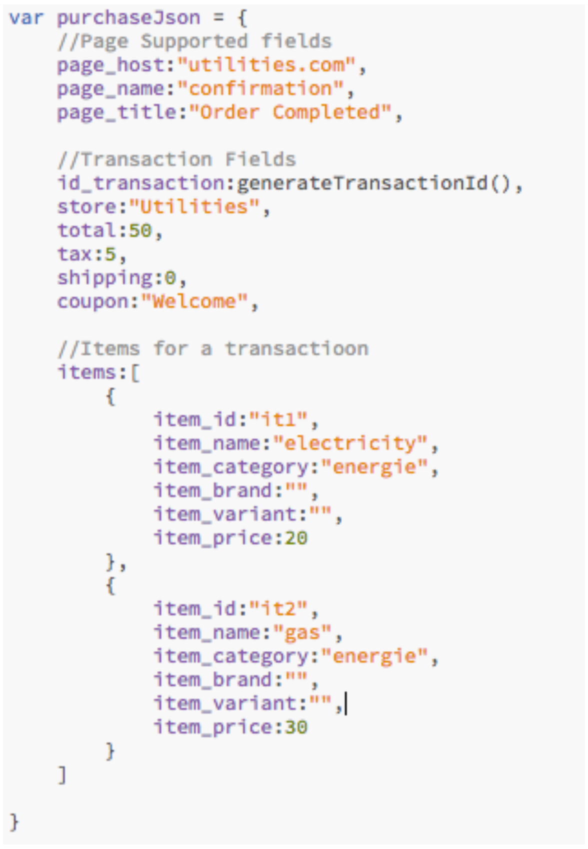
There are 2 ways for registering a purchase:

- <u>Single transaction with a value:</u> In this case the library implements the classic way for Google analytics ecommerce.



```
var purchaseJson = {
    //Page Supported fields
    page_host:"utilities.com",
    page_name:"confirmation",
    page_title:"Order_Completed",

    //Transaction Fields
    id_transaction:generateTransactionId(),
    store:"Utilities",
    total:50,
    tax:5,
    shipping:0,
    coupon:"Welcome",

    //Single Item
    item_name:"Voucher",
    item_price:50,
    item_quantity:1,
    item_id:"vou_uti",
    item_category:"others"

};
```

- <u>Transaction with multiple products:</u> By using the enhanced Google analytics ecommerce, multiple items can be declared within the same transaction and the same hit.



```
var purchaseJson = {
    //Page Supported fields
    page_host:"utilities.com",
    page_name:"confirmation",
    page_title:"Order Completed",

    //Transaction Fields
    id_transaction:generateTransactionId(),
    store:"Utilities",
    total:50,
    tax:5,
    shipping:0,
    coupon:"Welcome",

    //Items for a transactioon
    items:[
        {
            item_id:"it1",
            item_name:"electricity",
            item_category:"energie",
            item_brand:"",
            item_variant:"",
            item_price:20
        },
        {
            item_id:"it2",
            item_name:"gas",
            item_category:"energie",
            item_brand:"",
            item_variant:"",
            item_price:30
        }
    ]

}
```

Error tracking:

- Any other structure will cause a tracked exception error.
- If “item_category”, “id_transaction” and “page_name” are not provided, it will fail the pre validation step and a tracked exception error will be generated.
- The other fields can be not provided (Optional).
- If quantity is not provided, by default it will send 1.

Output:

-
  ```
  v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=confirmation&dh=utilities.com&dt=Order_Completed&ti=
  fe941f&ta=Utilities&tr=50&tt=5&ts=0&tcc=Welcome&pa=purchase&in=Voucher&ip=25&iq=2&ic=vou_uti&iv=others
  ```

-
  ```
  v=1&tid=UA-86314412-1&cid=1234eckoh&t=pageview&dp=confirmation&dh=utilities.com&dt=Order_Complete
  d&ti=e3e5f7&ta=Utilities&tr=50&tt=5&ts=0&tcc=Welcome&pa=purchase&pr1id=it1&pr1nm=electricity&pr1ca=e
  nergie&pr1br=&pr1va=&pr1pr=20&pr2id=it2&pr2nm=gas&pr2ca=energie&pr2br=&pr2va=&pr2pr=30
  ```

4.2. Refunds

In the other hand, refunds can also be tracked. The library can manage either a full refund for a transaction or a single or multiple items from a transaction.

Here there are the fields needed to be send within the link:

```
v=1                                    // Version.
&tid=UA-XXXXX-Y                        // Tracking ID / Property ID.
&cid=555                               // Anonymous Client ID.
&t=event                               // Event hit type.
&ec=Ecommerce                          // Event Category. Required.
&ea=Refund                             // Event Action. Required.
&ni=1                                  // Non-interaction parameter.

&ti=T12345                             // Transaction ID. Required.
&pa=refund                             // Product action (refund). Required.
```

Item parameters:

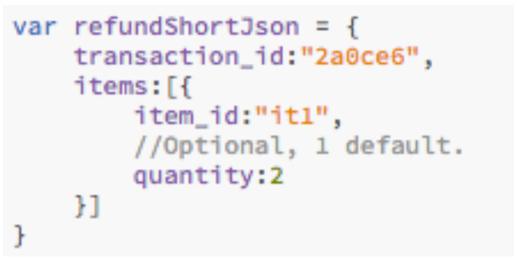
```
&pr1id=P12345                          // Product 1 ID. Required.
&pr1qt=1                               // Product 1 quantity. Required.
```

Eckoh_GA.js

```
refundTransaction(refundString);
refundTransaction(refundArray);
refundTransaction(refundJson);
```

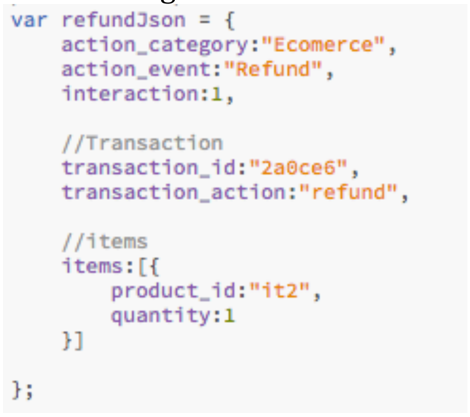
There are 3 different ways for making a refund. Arrays of multiple refund transactions for different items are not supported.

1. Full transaction (Single): By giving the transaction id, the library completes the rest of the fields.
   ```
   var refundString = "2a0ce6";
   ```
2. Array of transactions (Multiple): A list of transactions might be also deleted at the same time.
   ```
   var refundArray = ["2a0ce6","2a0ce7"];
   ```
3. Specific items for a transaction (Single Only): Refunds part of the transaction:
   a. *Short*: Only the fields needed.



```
var refundShortJson = {
    transaction_id:"2a0ce6",
    items:[{
        item_id:"it1",
        //Optional, 1 default.
        quantity:2
    }]
}
```

b. *Completed:* All the fields given.



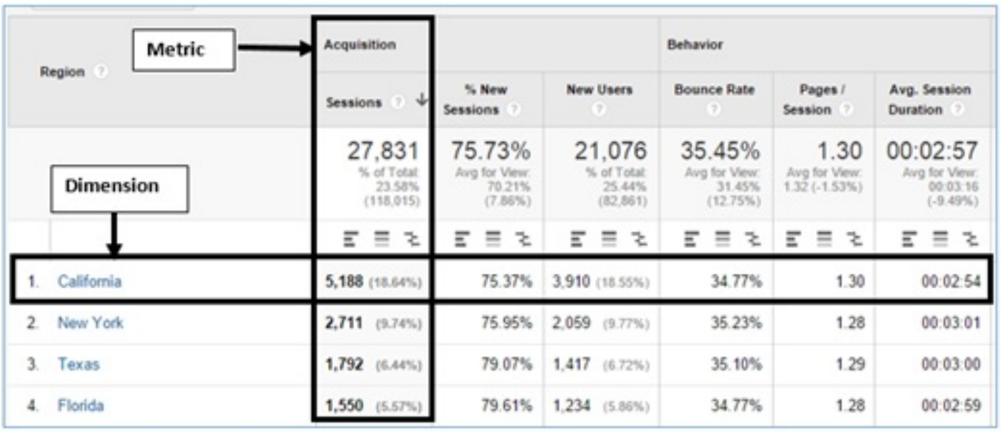
```
var refundJson = {
    action_category:"Ecomerce",
    action_event:"Refund",
    interaction:1,

    //Transaction
    transaction_id:"2a0ce6",
    transaction_action:"refund",

    //items
    items:[{
        product_id:"it2",
        quantity:1
    }]

};
```

Error tracking:

- Any other structure will cause a tracked exception error.
- If "transaction_id" is not provided, it will fail the pre validation step and a tracked exception error will be generated.
- The other fields can be not provided (Optional).
- If quantity is not provided, the library will assume it's only one item.

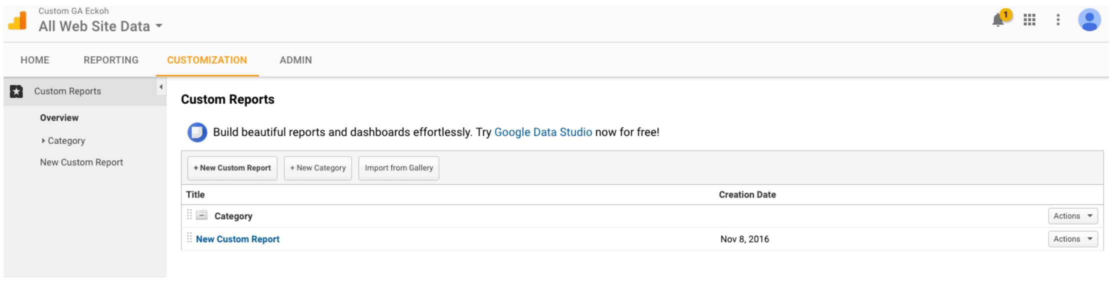
5. Custom demographics and metrics (Advanced)

Google analytics expands what websites and apps can track by offering custom dimension and metrics[2]. Don't confuse with custom variables from the previous versions.



Region	Acquisition			Behavior		
	Sessions	% New Sessions	New Users	Bounce Rate	Pages / Session	Avg. Session Duration
	27,831 % of Total: 23.58% (118,015)	75.73% Avg for View: 70.21% (7.86%)	21,076 % of Total: 25.44% (82,861)	35.45% Avg for View: 31.45% (12.75%)	1.30 Avg for View: 1.32 (-1.53%)	00:02:57 Avg for View: 00:03:16 (-9.49%)
1. California	5,188 (18.64%)	75.37%	3,910 (18.55%)	34.77%	1.30	00:02:54
2. New York	2,711 (9.74%)	75.95%	2,059 (9.77%)	35.23%	1.28	00:03:01
3. Texas	1,792 (6.44%)	79.07%	1,417 (6.72%)	35.10%	1.29	00:03:00
4. Florida	1,550 (5.57%)	79.61%	1,234 (5.86%)	34.77%	1.28	00:02:59

A part from that, Google analytics also offers the possibility for creating custom reports that are helpful for booth custom dimension/metrics and default dimension/metrics reporting.



Custom Reports

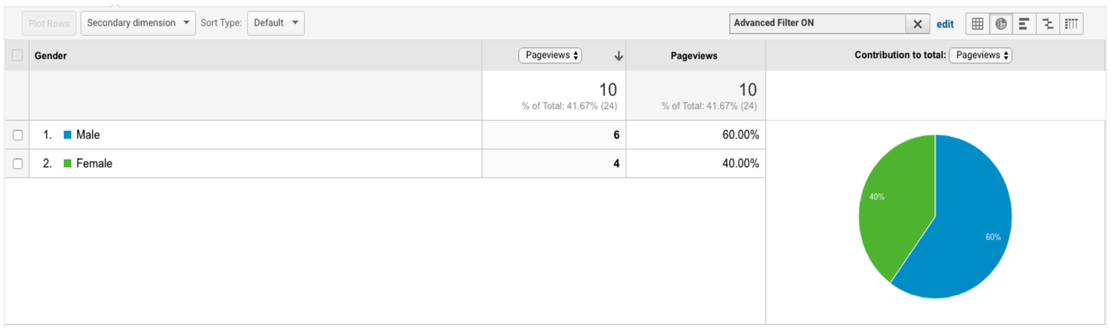
Build beautiful reports and dashboards effortlessly. Try Google Data Studio now for free!

Title	Creation Date
Category	
New Custom Report	Nov 8, 2016

2https://www.optimizesmart.com/complete-guide-to-dimensions-and-metrics-in-google-analytics/

5.1. Custom demographics

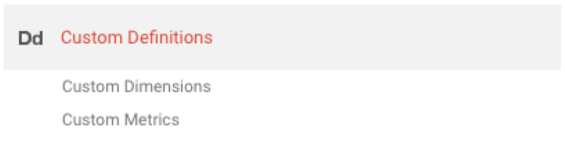
Although Google analytics says that demographics can be tracked, neither analytics.js nor ga.js (old Google Analytics) are dealing with that feature. For sending demographics data, websites needs to implement "double click cookies". These kinds of cookies are third party cookies used for advertisements and, at certain level they depend on **analytics.js**. Is for that reason that if demographics are going to be implemented, they need to be implemented using custom metrics and, in fact, that is the method that most developers suggest to do.



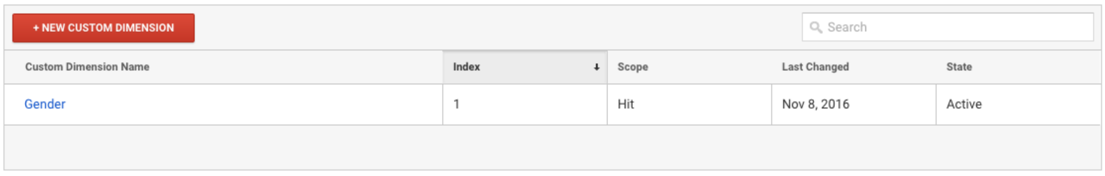
Gender	Pageviews	Pageviews
	10 % of Total: 41.67% (24)	10 % of Total: 41.67% (24)
1. 🟦 Male	6	60.00%
2. 🟩 Female	4	40.00%

There are many ways for tracking demographics using custom dimensions (also it can be achieved using metrics). The following example shows how it can be achieved but it is more than advisable to read the document provided in the last page footer.

1. Under **property** click into **custom definitions**.

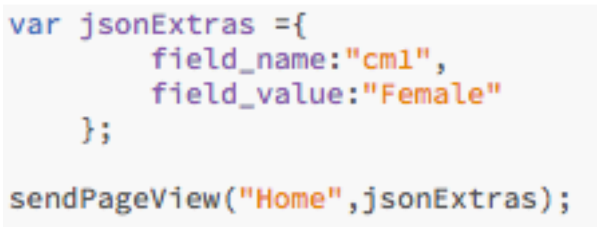


2. Create a new custom dimension



Custom Dimension Name	Index	Scope	Last Changed	State
Gender	1	Hit	Nov 8, 2016	Active

3. Add the extras (in JSON format) when calling a hit function. The field "**cd1**" stands for custom dimension at index one.



```
var jsonExtras ={
        field_name:"cm1",
        field_value:"Female"
    };

sendPageView("Home",jsonExtras);
```

How the demographics variables are got and stored through all the customer journey is up to the developer.

6. Auto fill values

When a hit reaches Google backend, Google gets some extra information such as location, browser, and operating system among others. However, on the other hand, when "analytics.js" is used, it sends some other information for a better accuracy. Eckoh_ga.js is a smart library and tries to get the some extra information without the developer get worry about.

The information that library, by now, gets is:

- Language: If it's not provided in the header, English is the default option.
- Screen resolution: Important for mobile devices that running web browsers.
- Data source: Indicates the data source of the hit. Hits sent from a web will have the value of "web" and the ones sent from an app will have the value "app".
- Document encryption: Indicates the type of encoding.
- Screen colours: Indicates the bit resolution.

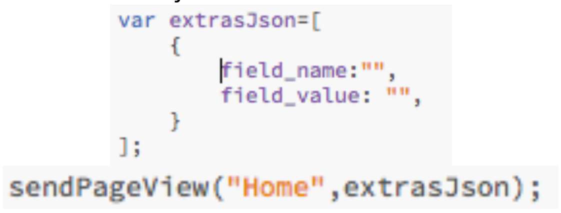
7. Extras fields

Some automatic tracked information and ecommerce transactions can be either overwritten or extended. Also it applies to demographics and custom metrics and dimensions. It only will work for the global hit (If multiples, in each hit) and not for the items on the hit.

Because there are a lot of possible values and different combinations, the extra fields need to be declared by the field name that Google Analytics uses[3] and followed with the value using one of the following structures.

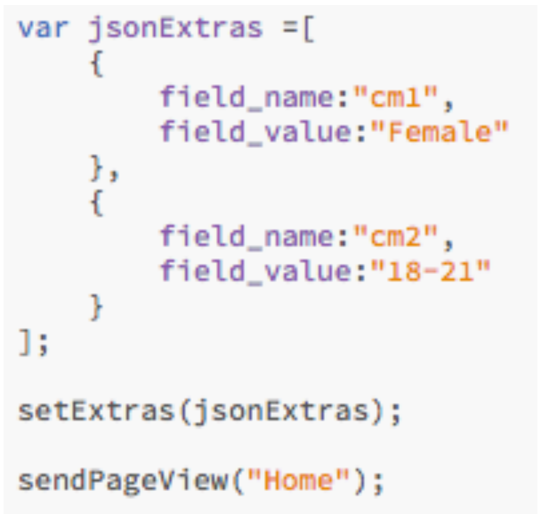
There are 3 different ways for sending extra data with a hit:

1. JSON object with the function call method. That is probably the easiest way because the JSON object won’t be restored after it is been used.



```
var extrasJson=[
    {
        field_name:"",
        field_value: "",
    }
];
sendPageView("Home",extrasJson);
```

2. By calling a function that initialises or adds the different extras. It can be call more than once before a hit function and it will automatically restored after the hit has been sent.



```
var jsonExtras =[
    {
        field_name:"cm1",
        field_value:"Female"
    },
    {
        field_name:"cm2",
        field_value:"18-21"
    }
];

setExtras(jsonExtras);

sendPageView("Home");
```

3. A mixture of both is possible. Very useful for sending different extras with a fixed extras.

Error tracking:

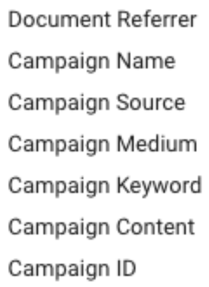
- Any other structure will cause a tracked exception error.

[3] https://developers.google.com/analytics/devguides/collection/protocol/v1/parameters#user

7.1. Campaigns

Some times organisations want to know what make users go to the site. By doing that they can decide where to spend more on promotion.

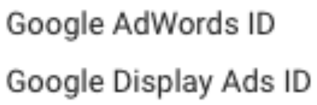
These fields are the ones that Google uses for tracking all the data about campaigns.



Document Referrer
Campaign Name
Campaign Source
Campaign Medium
Campaign Keyword
Campaign Content
Campaign ID

7.2. AdWords

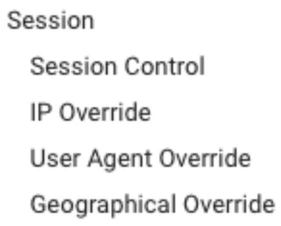
With the same principle, adverts campaigns can also be tracked.



Google AdWords ID
Google Display Ads ID

7.3. Session

Also some session data can be overwritten.



Session
Session Control
IP Override
User Agent Override
Geographical Override

8. Taking over from another analytics

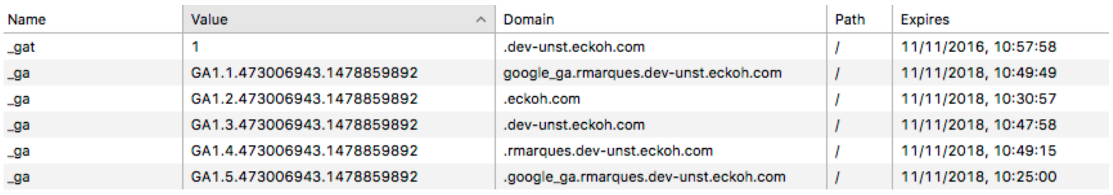
One of the main concerns at the moment of the library proposal was that most of the websites were already implementing analytics. So, on the event of writing a new module (in this case a payment module), Eckoh not only can offer a secure and efficient implementation of Google analytics but also can take over from a current Google analytics implementation.

8.1. Cookie security level and CID

As explained before, Google analytics uses a cookie stored on the browser for identifying different visitors. These cookies are different for a browser and device so a single user can get multiple identities.

Furthermore, these cookies, as any others, can have different security level. By default, Google analytics cookies (A.K.A. "_ga"[4]) has an "auto" cookie security configuration that finds the best option and, typically, it is domain associated (G.A1.2).

On the event that Google analytics is hosted in "http://google_ga.rmarques.dev-unst.eckoh.com" these are the different security levels.



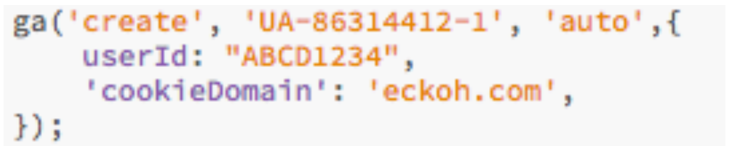
Name	Value	Domain	Path	Expires
_gat	1	.dev-unst.eckoh.com	/	11/11/2016, 10:57:58
_ga	GA1.1.473006943.1478859892	google_ga.rmarques.dev-unst.eckoh.com	/	11/11/2018, 10:49:49
_ga	GA1.2.473006943.1478859892	.eckoh.com	/	11/11/2018, 10:30:57
_ga	GA1.3.473006943.1478859892	.dev-unst.eckoh.com	/	11/11/2018, 10:47:58
_ga	GA1.4.473006943.1478859892	.rmarques.dev-unst.eckoh.com	/	11/11/2018, 10:49:15
_ga	GA1.5.473006943.1478859892	.google_ga.rmarques.dev-unst.eckoh.com	/	11/11/2018, 10:25:00

As shown:

- G.AX.X.: Shows the level security.
- CID: A random number formed using 2 other numbers separated with a coma.

Although 5 different variables are displayed, the CID number is the same in all the cases.

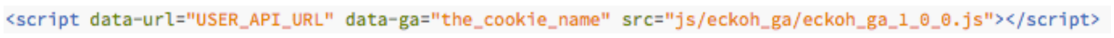
Before, accessing to the page that is using "eckoh_ga.js", security level can be changed according the needs.



```
ga('create', 'UA-86314412-1', 'auto',{
    userId: "ABCD1234",
    'cookieDomain': 'eckoh.com',
});
```

4 https://developers.google.com/analytics/devguides/collection/analyticsjs/cookies-user-id

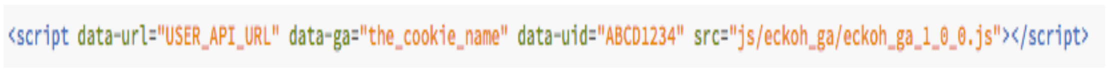
If the cookie name has been changed (option also available), it needs to be specified when the library is declared.



```
<script data-url="USER_API_URL" data-ga="the_cookie_name" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```

8.2. Taking over of UID

How user tracking is implemented has also been seen but, if the organisation has already made the effort of building a custom UID tracking, the library also can take over the user tracking. To do so, the user id needs to be given when declaring the library.



```
<script data-url="USER_API_URL" data-ga="the_cookie_name" data-uid="ABCD1234" src="js/eckoh_ga/eckoh_ga_1_0_0.js"></script>
```

9. Next steps

- Ecommerce impressions
- More Managing functions

10. Recommended links

- https://developers.google.com/analytics/devguides/collection/protocol/v1/parameters#user
- https://ga-dev-tools.appspot.com/hit-builder/
- https://developers.google.com/analytics/devguides/collection/protocol/v1/devguide#enhancedecom
- https://www.optimizesmart.com/complete-guide-to-dimensions-and-metrics-in-google-analytics/