FREE eBook

LEARNING google-analytics

Free unaffiliated eBook created from **Stack Overflow contributors.**

#google-analytics

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: google-analytics

It is an unofficial and free google-analytics ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official google-analytics.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with google-analytics

Remarks

Google Analytics and the Google Analytics developer platform allows you to collect, configure, and analyze your data to reach the right audience.

Libraries and SDKs for tracking

- Web Tracking (analytics.js): Measure user interaction with websites or web applications.
- Android: Measure user interaction with Android applications.
- iOS: Measure user interaction with iOS applications.
- Measurement Protocol: Measure user interaction in any environment with this low-level protocol.
- Unity: Track user interactions in Unity games.
- AMP HTML: Track user interactions in AMP pages.

APIs for reporting and configuration

- Core Reporting API: Query for dimensions and metrics to produce customized reports.
- Embed API: Easily create and embed dashboards on a 3rd party website in minutes.
- Multi-Channel Funnels Reporting API: Query the traffic source paths that lead to a user's goal conversion.
- Real Time Reporting API: Report on activity occurring on your property right now.
- Metadata API: Access the list of API dimensions and metrics and their attributes.
- Management API: View and manage accounts, properties, views, filters, uploads, permissions, etc.
- Provisioning API: Create Google Analytics accounts and enable Google Analytics for your customers at scale.

Examples

Adding analytics.js to your website

Add the following code (known as the "JavaScript tracking snippet") to your site's templates.

The code should be added before the closing tag, and the string **'UA-XXXXX-Y'** should be replaced with the property ID (also called the "tracking ID") of the Google Analytics property you wish to track.

```
<!-- 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 above code does four main things:

1. Creates a `<script>` element that starts asynchronously downloading the analytics.js JavaScript library from https://www.google-analytics.com/analytics.js
2. Initializes a global ga function (called the `ga()` command queue) that allows you to schedule commands to be run once the analytics.js library is loaded and ready to go.
3. Adds a command to the `ga()` command queue to create a new tracker object for the property specified via the `'UA-XXXXX-Y'` parameter.
4. Adds another command to the `ga()` command queue to send a pageview to Google Analytics for the current page.

Overview

Google Analytics is used to track user activity on your website or mobile application.

To set up google-analytics on a website you will need to get a snippet of JavaScript code from Google that you embed in the head of each page on your site that you want to track user activity.

Get the code snippet at www.google.com/analytics, and on the Admin tab select “Create new account” from the dropdown menu of the account column on the left.

Track pages called by AJAX and non-html content

To track so called "virtual pageviews", use the `ga('send')` method right after your asynchronous request:

Syntax: `ga('send', 'pageview', 'path to your virtual page');`

Example (Simple Link):

```
<a href="http://example.com/my.pdf"
   onClick="ga('send', 'pageview', '/virtual/my.pdf');">Download PDF</a>
```

Example (JQuery AJAX):

```
$.ajax({
    url: '/ajax-url/file.json',
    data: {page: 4},
    success: function(data) {
        ga('send', 'pageview', '/ajax-url/file.json');
        console.log("Got response",data);
    },
    dataType: 'json',
    method: 'GET'
});
```

Sources:

- How do I get Google Analytics to track pages called by AJAX?
- Tracking virtual pageviews - developers.google.com

Alternative async tracking snippet

While the JavaScript tracking snippet described above ensures the script will be loaded and executed asynchronously on all browsers, it has the disadvantage of not allowing modern browsers to preload the script.

The alternative async tracking snippet below adds support for preloading, which will provide a small performance boost on modern browsers, but can degrade to synchronous loading and execution on IE 9 and older mobile browsers that do not recognize the async script attribute. Only use this tracking snippet if your visitors primarily use modern browsers to access your site.

```
    <!-- Google Analytics -->
<script>
window.ga=window.ga||function(){(ga.q=ga.q||[]).push(arguments)};ga.l=+new Date;
ga('create', 'UA-XXXXX-Y', 'auto');
ga('send', 'pageview');
</script>
<script async src='https://www.google-analytics.com/analytics.js'></script>
<!-- End Google Analytics -->
```

Using Plugins

Plugins are scripts that enhance the functionality of analytics.js to aid in measuring user interaction. Plugins are typically specific to a set of features that may not be required by all Google Analytics users, such as ecommerce or cross-domain tracking, and are therefore not included in analytics.js by default.

This guide explains how to require and use analytics.js plugins.

The require command takes the name of a plugin and registers it for use with the `ga()` command queue. If the plugin accepts configuration options, those options can be passed as the final argument to the require command.

The following is the full require command's signature:

```
ga('[trackerName.]require', pluginName, [pluginOptions]);
```

For example, here is how you would require the Enhanced Ecommerce plugin for use with the default tracker:

```
ga('require', 'ec');
```

And here is how you would require the Display Features plugin for a tracker named "myTracker" and pass a configuration option that overrides the default cookie name value:

```
ga('myTracker.require', 'displayfeatures', {
  cookieName: 'display_features_cookie'
});
```

What data does the tracking snippet capture?

When you add either of these tracking snippets to your website, you send a pageview for each page your users visit. Google Analytics processes this data and can infer a great deal of information including:

The total time a user spends on your site. The time a user spends on each page and in what order those pages were visited. What internal links were clicked (based on the URL of the next pageview). In addition, the IP address, user agent string, and initial page inspection analytics.js does when creating a new tracker is used to determine things like the following:

The geographic location of the user. What browser and operating system are being used. Screen size and whether Flash or Java is installed. The referring site.

Getting on board with Google Analytics

1. **Getting a GA Account**: If you don't have an Analytics account, create one. If you do have an Analytics account, sign in. Both options are available at google.com/analytics
2. **Setting up a property in your Analytics account:** A property represents your website or app where the data gets aggregated.
3. **Create a view:** Views let you create filtered perspectives of your data. When you create a property one view is created by default. You can create multiple views based on the requirement and filter the reports based on the reporting structure.
4. **Embed the Analytics in your website**: Go to property > tracking info and get the tracking code which looks like below

```
<!-- 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 -->
```

5. The string '**UA-XXXXX-Y**' should be replaced with the property ID (also called the "tracking ID") of the Google Analytics property you wish to track.

With these simple steps, your website will be ready to send the pageviews to GA.

Read Getting started with google-analytics online: https://riptutorial.com/google-analytics/topic/1246/getting-started-with-google-analytics

Chapter 2: Create and manage goals

Examples

Smart Goals

In addition to the goal types described above, Analytics provides an alternative conversion tracking method called Smart Goals. Smart Goals are specifically designed to help AdWords advertisers who may not have enough conversions to use the AdWords optimization tools, such as automated bidding. When you have Smart Goals enabled, Analytics automatically evaluates your website or app visits and assigns each a score, with the "best" visits being translated into Smart Goals.

Read Create and manage goals online: https://riptutorial.com/google-analytics/topic/6142/create-and-manage-goals

Chapter 3: Data processing latency

Examples

What is Data processing latency?

When a hit is sent to Google Analytics the data must be processed. Processing latency is 24-48 hours. This means that it can take time before you will see data under standard reports (Not real-time) any data that you do see may not be correct as it has probably not completed processing.

Standard accounts that send more than 200,000 sessions per day to Analytics will result in the reports being refreshed only once a day. This can delay updates to reports and metrics for up to two days. To restore intra-day processing, reduce the number of sessions your account sends to < 200,000 per day. For Analytics 360 accounts, this limit is extended to 2 billion hits per month.

Read Data processing latency online: https://riptutorial.com/google-analytics/topic/6413/data-processing-latency

Chapter 4: Displaying Google Analytics data in your Website

Examples

Displaying Google Analytics data in your Website

This document explains how to get Google Access tokens and use them to get Google Analytics data to be displayed in our websites.

Example: A live example is available in

> https://newtonjoshua.com

note: Use the same gmail account for all the below steps.

STEP 1: Set Up Google Analytics

Follow the below steps to set up Google Analytics in your website

1. Sign in to your Analytics account.
2. Select the Admin tab.
3. Select an account from the drop-down menu in the ACCOUNT column.
4. Select a property from the drop-down menu in the PROPERTY column.
5. Under PROPERTY, click Tracking Info > Tracking Code.
6. To collect data, you must copy and paste the Analytics tracking code into the source code on every web page you wish to track.
7. Once you have the Javascript tracking code snippet for your property, copy the snippet exactly without editing it.
8. Paste your tracking code snippet (unaltered, in its entirety) before the closing tag on every web page on your site you wish to track.
9. Once you have successfully installed Analytics tracking, it may take up to 24 hours for data such as traffic referral information, user characteristics, and browsing information to appear in your reports

refer,

1. https://support.google.com/analytics/answer/1008080?hl=en
2. https://analytics.google.com

STEP 2: Get Tokens

Google Project:

To Create a Google Cloud Platform project, open the Google Developers Console (https://console.developers.google.com) and click *Create Project.*

Enable OAuth 2.0 API access:

Your app will need to access user data and contact other Google services on your behalf. Use OAuth 2.0 to grant your app API access.

To enable that, you need a Client ID:

1. Open the Google API Console Credentials page (https://console.developers.google.com/apis/credentials).
2. From the project drop-down, select your project.
3. Select *Create credential*s and choose *OAuth client ID*.
4. Under Application type, select *Web application*, enter a Name and
5. set the Restrictions by entering *JavaScript origins*, ***Redirect URIs*** to point the website where you are planning to display the data, then click *Create*.

5. Make note of the OAuth 2.0 ***client_id*** and ***client_secret***. You will need them to configure the UI.

Get Authorization code:

Enter in browser,

> https://accounts.google.com/o/oauth2/auth?scope=https://www.googleapis.com/auth/analytics.reado
> {{ client_id}}&redirect_uri={{redirect_uri }}
> &approval_prompt=force&access_type=offline

You will get redirected to

> {{redirect_uri }}?code=={{***authorization_code***}}#

Get Refresh Token:

Send a POST request, possibly via a REST console to

> https://www.googleapis.com/oauth2/v3/token?code={{authorization_code}} &client_id=
> *{{client_id}}*&client_secret=*{{client_secret}}* &redirect_uri=*{{redirect_uri }}*
> &grant_type=authorization_code

You will get a JSON response with

> {"refresh_token": ***refresh_token***}

You can use the refresh toke to get access token to access to Google APIs

Get Access Token:

Send a POST request to,

https://www.googleapis.com/oauth2/v3/token?client_id=*{{client_id}}* &client_secret= *{{client_id}}* &grant_type=refresh_token&refresh_token=*{{refresh_token}}*

You will get a JSON with access_token in the response.

{access_token: ***{{access_token}}***}

Example:

```
 var access_token = '';
function getAccessToken(){
    $.post('https://www.googleapis.com/oauth2/v3/token', {
            client_id: {{client_id}},
            client_secret: {{client_secret}},
            grant_type: 'refresh_token',
            refresh_token: {{refresh_token}}
        }, function (data, status) {
            if (status === 'success') {
                access_token = data.access_token;
                // Do something eith the access_token
            } else {
                console.error(status);
            }
        });
}
```

Check Token validity:

Send a POST request to,

https://www.googleapis.com/oauth2/v1/tokeninfo?access_token={{access_token}}

Example:

```
function checkValidity() {
    $.post('https://www.googleapis.com/oauth2/v1/tokeninfo', {
            access_token:{{access_token}}
        }).done(function (data, status) {
            if (status === 'success') {
                console.debug(data.expires_in);
                var check = false;
                check = data.hasOwnProperty('expires_in');
                if (check) {
                    // Token is valid
                }
                if (!check) {
                    getAccessToken();
                }
            } else {
                console.debug(status);
            }

        })
        .fail(function (data) {
            console.error(data);
            getAccessToken();
        });
}
```

Step 3: Fetch Data

Embed API:

The GA Embed API is a JavaScript library that allows you to easily create and embed your GA dashboard on your website in a matter of minutes.

refer https://developers.google.com/analytics/devguides/reporting/embed/v1/getting-started

Query Explorer: visit Embed API Query Explorer and authorize

> https://ga-dev-tools.appspot.com/query-explorer/

Select the view for wich you want to fetch the data.

Select the required metrics and dimensions.

Example:

Get Country Data (I want to know the number of users accessing my website from each country)

To get that data, select the metrics as 'users' and the dimensions as 'country'

Click on *Run Query*

You will find the analytics data for the query displayed in a table.

Copy the ***API Query URI***. And add access_token=*{{access_token}}* to the uri

Example:

> https://www.googleapis.com/analytics/v3/data/ga?ids={{ids}}&start-date=2015-07-01&end-date=today&metrics=ga%3Ausers&dimensions=ga%3A*country* &access_token=*{{access_token}}*

Send POST request to the URIs to get the data in your browser.

Example:

```
function gaGetCountry() {
    $.get('https://www.googleapis.com/analytics/v3/data/ga?' +
        'ids={{ids}}' +
        'start-date=2015-07-01&' +
        'end-date=today&' +
        'metrics=ga%3Ausers&' +
        'dimensions=ga%3Acountry&' +
        'sort=ga%3Ausers&' +
        'filters=ga%3Ausers%3E10&' +
        'max-results=50' +
        '&access_token=' + {{access_token}},
        function (data, status) {
            if (status === 'success') {

                // Display the Data
                drawRegionsMap(data.rows);

            } else {
                console.debug(status);
            }

        });
}
```

Step 4: Display Data

Now we have gathered the data. Finally we have to diaplay them in our website.

"*Display live data on your site*" is the title of Google Charts. And that is what we are going to do.

refer https://developers.google.com/chart/

The following example will draw a GeoChart in the div with id='countryChart'

```
  //Draw country Chart
  function drawRegionsMap(data) {

         var head = data[0];
         head[0] = 'Country';
         head[1] = 'Users';
         for (var i = 1; i < data.length; i++) {
             var d = data[i];
             d[1] = Number(d[1]);
         }

         var chartData = google.visualization.arrayToDataTable(data);
         var options = {
             title: 'My Website is viewed from,',
             domain: '{{Country Code eg: IN for India}}',
             tooltip: {
                 textStyle: {
                     color: 'navy'
                 },
                 showColorCode: true
             },
             legend: {
                 textStyle: {
                     color: 'navy',
                     fontSize: 12
                 }
             },
             colorAxis: {
                 colors: ['#00FFFF', '#0000FF']
             }
         };

         var chart = new
```

```
google.visualization.GeoChart(document.getElementById('countryChart'));

        chart.draw(chartData, options);
}
```

Refer https://newtonjoshua.com to view the above exple in action.

Read Displaying Google Analytics data in your Website online: https://riptutorial.com/google-analytics/topic/7430/displaying-google-analytics-data-in-your-website

Chapter 5: Event Tracking

Introduction

Below is the example of how to hard code the google analytics implementation in the website. To track the shopping cart actions let induce the tracking snippet. It look as below. You can track the events in google analytics open source tool

Syntax

- `ga('send', 'event', [eventCategory], [eventAction], [eventLabel], [eventValue], [fieldsObject]);`

Parameters

Field Name	Description
eventCategory	Typically the object that was interacted with (e.g. 'Video')
eventAction	The type of interaction (e.g. 'play')
eventLabel	Useful for categorizing events (e.g. 'Fall Campaign')
eventValue	A numeric value associated with the event (e.g. 42)

Examples

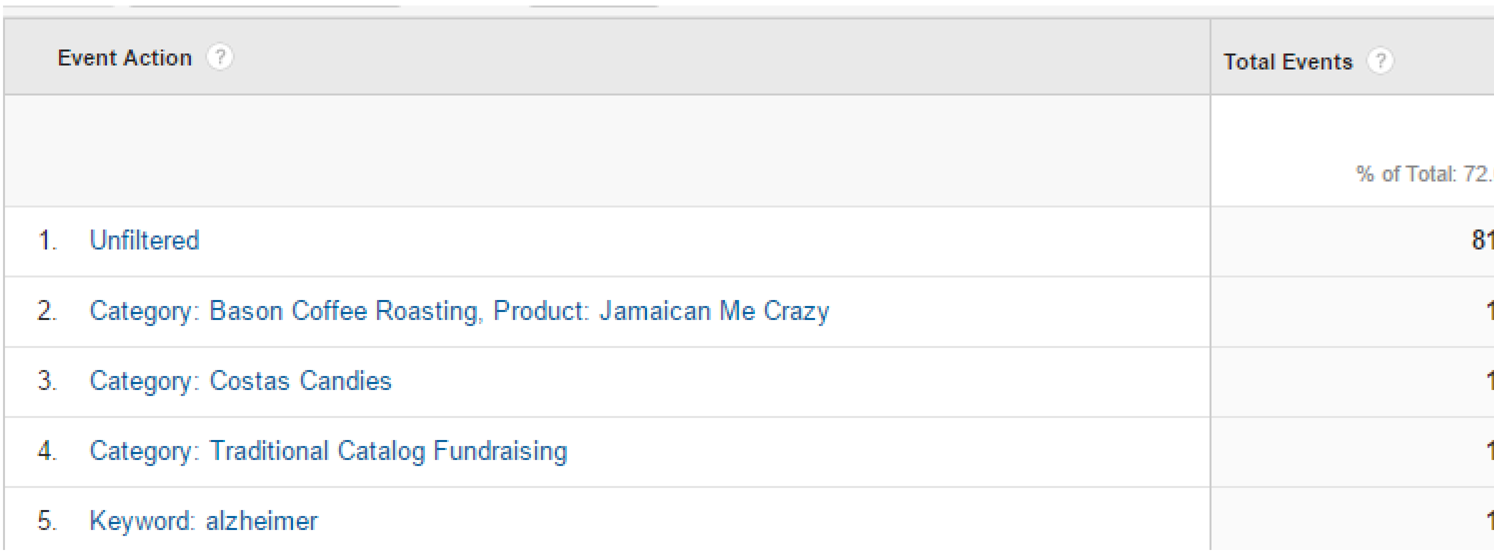
Tracking searches within your site

Insert the following function call within your JavaScript when searching within your website to track how visitors are using your internal search features.

In this example, the Event Action `filters` is a comma-delimited list of search filter name/value pairs, and the Event Label `orderedBy` is a string describing the user-determined sort order.

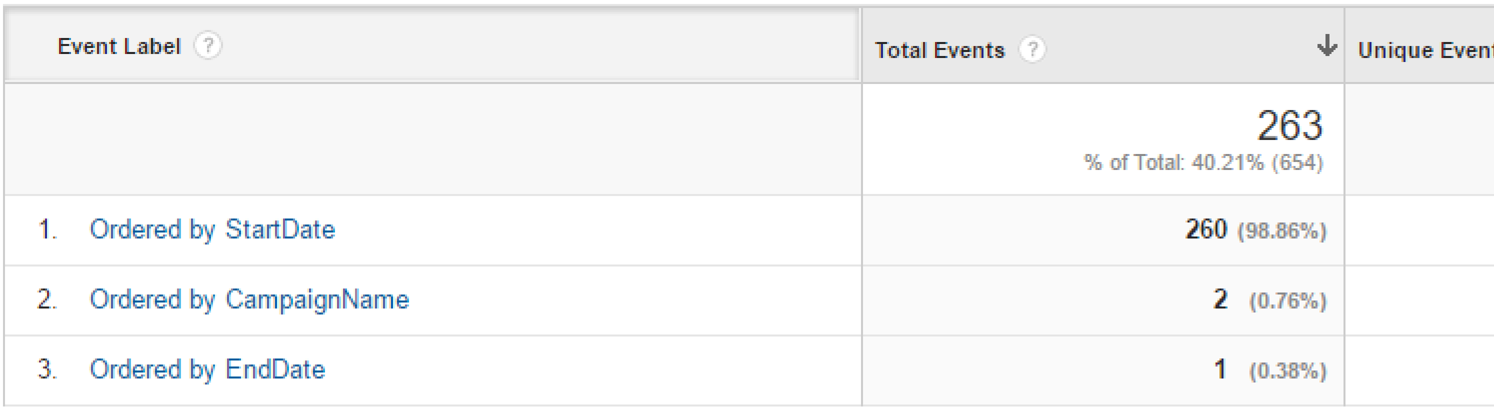
```
ga('send', 'event', 'Product Search', filters, orderedBy);
```

These events can then be viewed within Analytics under Behavior > Events:



Event Action	Total Events
	% of Total: 72.
1. Unfiltered	81
2. Category: Bason Coffee Roasting, Product: Jamaican Me Crazy	1
3. Category: Costas Candies	1
4. Category: Traditional Catalog Fundraising	1
5. Keyword: alzheimer	1

Likewise, the searches can be viewed by Label to see how users are sorting their data:



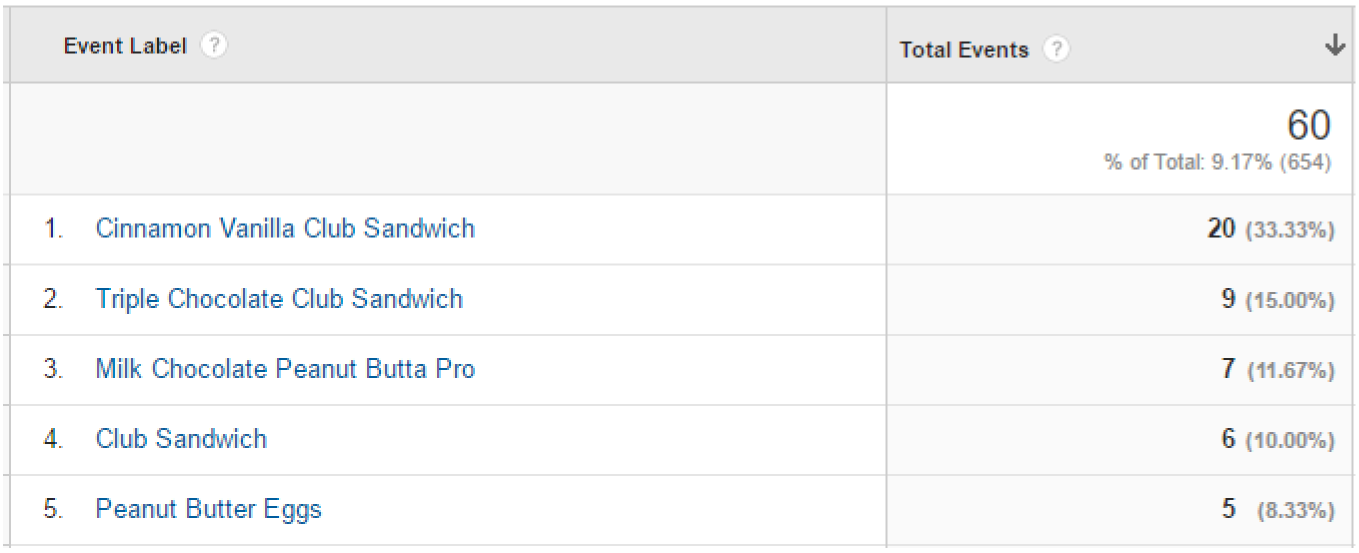
Event Label	Total Events	Unique Event
	263 % of Total: 40.21% (654)	
1. Ordered by StartDate	260 (98.86%)	
2. Ordered by CampaignName	2 (0.76%)	
3. Ordered by EndDate	1 (0.38%)	

Track shopping cart actions

Adding a product to a shopping cart (Label `item.name` references the name property of the product added):

```
ga('send', 'event', 'Cart', 'Add', product.name);
```

This lets you see what people are adding to the shopping cart, even if they never complete the order, allowing more insight into where users are abandoning their session:



Event Label	Total Events
	60 % of Total: 9.17% (654)
1. Cinnamon Vanilla Club Sandwich	20 (33.33%)
2. Triple Chocolate Club Sandwich	9 (15.00%)
3. Milk Chocolate Peanut Butta Pro	7 (11.67%)
4. Club Sandwich	6 (10.00%)
5. Peanut Butter Eggs	5 (8.33%)

Removing an item from a shopping cart:

```
ga('send', 'event', 'Shopping', 'Removed', product.name);
```

Emptying a shopping cart:

```
ga('send', 'event', 'Cart', 'Emptied', 'empty');
```

Read Event Tracking online: https://riptutorial.com/google-analytics/topic/6521/event-tracking

Chapter 6: Logging JavaScript errors within Google Analytics

Remarks

The example above has two tracking events, Event Tracking and Exception Tracking.

Event Tracking will allow you to see JS errors in real-time. Under `Real Time -> Events` sections.

Unfortunately, your error messages will be limited by 500 Bytes, so you will not be able to understand a problem properly, however you will know that something is going wrong.

Exception Tracking will give you more detailed report, with full error message and browser information.

You can generate Exception Tracking report with Custom Reports.

Examples

Following code will submit all JavaScript errors into 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','//www.google-analytics.com/analytics.js','_watchdog');

_watchdog('create', 'UA-xxxxxxx-x', 'auto');

window.onerror = function(msg) {
  _watchdog('send', 'exception', { 'exDescription': msg });

  _watchdog('send', 'event', {
    eventCategory: 'javascript',
    eventAction: 'error',
    eventLabel: msg,
    transport: 'beacon'
  });
}
</script>
```

Read Logging JavaScript errors within Google Analytics online: https://riptutorial.com/google-analytics/topic/6317/logging-javascript-errors-within-google-analytics

Chapter 7: Sampling

Parameters

Parameter	Details
sampleRate	Float number describing the amount of users in percent to be tracked. Default = 100.

Remarks

> Sampling in Analytics is the practice of selecting a subset of data from your traffic and reporting on the trends available in that sample set. Sampling is widely used in statistical analysis because analyzing a subset of data gives similar results to analyzing all of the data. In addition, sampling speeds up processing for reports when the volume of data is so large as to slow down report queries.

In layman's terms this means that when there is a large amount of data we can take a chunk of that data and analyze based upon that sample. When looking at large data sets it can often be faster to analyse upon a sample rather then the full data set. However one must always take into account that the results will not be 100% the same as if you had done the analysis upon the full data set.

Google Analytics handle's sampling is as follows:

> Analytics inspects the number of sessions for the specified date range at the property level. If the number of sessions in the property over the given date range exceeds 500k sessions (100M for Analytics 360)1, Analytics will employ a sampling algorithm which uses a sample set proportional to the distribution of sessions by day for the selected date range. Thus, the session sampling rate varies for every query depending on the number of sessions included in the selected date range for the given property.

Additional information can be found here: How sampling works

Examples

Sample Rate

```
ga('create', 'UA-XXXX-Y', {'sampleRate': 5});
```

Optional. This may only be set in the create method.

Specifies what percentage of users should be tracked. This defaults to 100 (no users are sampled out) but large sites may need to use a lower sample rate to stay within Google Analytics processing limits.

Read Sampling online: https://riptutorial.com/google-analytics/topic/2805/sampling

Credits

S. No	Chapters	Contributors
1	Getting started with google-analytics	Community, GlabbichRulz, James, James Steele, Matt, Nick, P. Alexandru, RamenChef, Tiem Song, Tushar
2	Create and manage goals	Beofett, P. Alexandru
3	Data processing latency	DalmTo
4	Displaying Google Analytics data in your Website	Newton Joshua
5	Event Tracking	Beofett, Priya
6	Logging JavaScript errors within Google Analytics	mrded
7	Sampling	acalb, DalmTo, Jensd