 **Talend User Component tGoogleAnalyticsInput**

**Purpose**

This component addresses the needs of gathering Google Analytics data for a large number of profiles and fine-grained detail data.

The component uses the Core Reporting API 3.0 and the Authentication API OAuth 2.0 final.

To provide the ability to run in multiple iterations the component has special capabilities to avoid multiple logins through iterations. Usually automated processes should not use personal accounts. This requirement is addressed by using a service account, which are the only preferred way to login into Google Analytics for automated processes.

Please in case of problems check the checklist at the end of this document.

**Talend-Integration**

This component can be found in the palette under Business->Google

This component provides an output flow and several return values.

**Parameters**

Parameters to connect to Google Analytics (setup client)

|  |  |  |
| --- | --- | --- |
| **Property** | **Content** | **Data types** |
| Application Name | Not necessary, but recommended by Google.  Simple provide the name of your application gathering data. ***Required*** | String |
| Authentication Method | Choose the method to authenticate:  Service Account or Client-ID for native applications | String |

Properties to use the authentication method: Service ccount

|  |  |  |
| --- | --- | --- |
| **Property** | **Content** | **Data types** |
| Service Account Email | The email address of the service account. Google creates this address within the process of creating a service account. Only for service accounts! ***Required*** | String |
| Key File (p12) | The Service Account Login works with private key file for authentication. In the process of creating a service account you download this file. Only for service accounts ***Required*** | String |

Properties to use the authentication method: Client-ID for native applications

|  |  |  |
| --- | --- | --- |
| **Property** | **Content** | **Data types** |
| User Account Email | Email of the user account or the Client-ID | String |
| Client secret file (json) | This json file downloaded for the Client-ID | String |

The usage of the “Client-ID for native applications” expects on the first run an user interaction with the Google web page and after finishing the form to approve the access right you need to close the browser to let the component continue, otherwise the authentication process will not complete.

Parameters to define the query

|  |  |
| --- | --- |
| **Property** | **Content** |
| View-ID | Set here the View-ID (formally known as Profile-ID). It is a 10-digit number (fast growing number range!)  ***Required!*** |
| Start Date | All queries need always a time range (only date, not time).  The value must be a String with the pattern “yyyy-MM-dd”. ***Required!*** |
| End Date | Time range end. If you want gather data for one date, use start date as end date.  The value must be a String with the pattern “yyyy-MM-dd”. ***Required!*** |
| Dimensions | Dimensions are like group clauses. These dimensions will group the metric values.  See advise for notations below. Separate multiple dimensions with a comma.  ***Not required (since release 1.5)*** |
| Metrics | Things you want to measure. Separate multiple metrics with a comma.  See advise for notations below.  ***Required!*** |
| Filters | Contains all used filters as concatenated string.  See advise for notation below |
| Sorts | Contains all sort criteria as concatenated string. Separate multiple dimensions/metrics with a comma.  See advise for notation below |
| Segment | Segments are stored filters within Google Analytics. In case of using a service account you have to use dynamic segments because saved segments are always belongs to a personal account. If you depend on save segments you have to use as authentication method Client-ID for native applications. |
| Sampling Level | Google Analytics can collect the result based on sampled data. This attribute tells Google Analytics which kind of sampling should be used (in case of sampling is necessary because of the amount of data).  These are the possible values:  DEFAULT: It is a balance between Speed and precision  FASTER: use more sampled data but the result returns faster  HIGHER\_PRECISION: use less sampled data and it takes longer to get the result |
| Deliver Totals Data Set | The API provides a totals record. This can be used to calculate percentage values or check results. This data set will be delivered (as first row) if option is checked or will be omitted if option is not checked. Date values (e.g. ga:date) remains empty (null) in the totals record. |
| Normalize Output | If true the component normalizes the otherwise flat record into two normalized outputs (dimensions and metrics).  For every raw record with its columns for dimensions and metrics this option creates one record per raw-record and dimension / metric.  E.g. if the component in the flat mode would create 3 records with 4 dimensions and 2 metrics it will create for the dimensions-flow 3 x 4 records and for the metrics flow 3 x 2 records. |
| Exclude ga:date dimension and provide value as return value | Set this to exclude the ga:date dimension from the normalized output flow for dimension and instead set the ga:date value in the globalMap as return value (available while the flow runs). |

**Explanation for the Normalized Output**

The normalized output as made for scenarios in which the job will be configured with metrics and dimensions at runtime. In this use case it is not possible to declare the appropriated schema for the flat output. The normalization creates 2 read only output schemas:

Dimensions

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Meaning** |
| ROW\_NUM | int | The row number from the original flat result row. It identifies the records, which belongs to together. |
| DIMENSION\_NAME | String | Name of the dimension |
| DIMENSION\_VALUE | String | Value of the dimension |

Metrics

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Meaning** |
| ROW\_NUM | int | The row number from the original flat result row. It identifies the records, which belongs together. |
| METRIC\_NAME | String | Name of the metric |
| METRIC\_VALUE | Double | Value of the metric |

**Advice for filter and segment notation**

For dimensions, metrics, filters and sorts you have to use the notation from the Google Core API:

<https://developers.google.com/analytics/devguides/reporting/core/dimsmets>

Filters can be concatenated with OR or AND operator.

Separate filter expressions with a comma means OR

Separate filter expressions with a semicolon means AND

Filter comparison operators:

|  |  |
| --- | --- |
| **Operator** | **Meaning** |
| “==” | Exact match to include |
| “!=” | Exact match to exclude |
| “=~” | Regex match to include (only for strings) |
| “!~” | Regex match to exclude (only for strings) |
| “>=” | Greater or equals than (only for numbers) |
| “=@” | Contains string |
| “!@” | Does not contains string |
| “>” | Greater than (only for numbers) |
| “<=” | Lower or equals than (only for numbers) |
| “<” | Lower than (only for numbers) |

**Building Flat Schema for Component**

In the schema you need an amount of columns equals to the sum of the number of dimensions and metrics.

Columns in the schema must start at first with dimensions (if provided) and ends with metrics.

Schema column types must match to the data types of the dimensions and metrics. The schema column names can differ from the names of dimensions and metrics. Only the order and there types are important.

In Talend schema columns must follow the Java naming rules therefore avoid writing ga:xxx instead use the name without the ga: namespace prefix.

Important: For date dimensions (e.g. ga:date) you must specify the date pattern as “yyyyMMdd” if you want it as Date typed value.

**Advanced Option Parameters**

|  |  |
| --- | --- |
| **Property** | **Content** |
| Timeout in s | How long should the component wait for getting the first result and fetching all result with one internal iteration |
| Static Time Offset (to past) | Within the process of login, the component requests an access token and use the local time stamp (because these tokens will expire after a couple of seconds)  Google rejects all requests to access tokens when the request is in the future compared to the timestamp in Google servers. If you experience such kind of problems, this options let the requests appear to be more in the past (5-10s was recognized as good time shift) |
| Fetch Size | This is the amount of data the component fetches at once. The value is used to set the max\_rows attribute. max\_rows means not the absolute amount of data! The component manages setting the start index to get all data. To achieve this, the component iterates as long as the last result set are completely fetched. |
| Local Number Format | You can get numbers in various formats. Here you can define the locale in which format double or float values are should textual format by the API. |
| Reuse Client for Iterations | If you use this component in iterations it is strongly recommended to set this option. It saves time to authenticate unnecessary often and avoids problems because of max amount of connects per time range. |
| Distinct Name Extension | The client will be kept with an automatically created name:  Talend-Name-Component name + job name. In case this is not distinct enough, you can specify an additional extension to the name. |

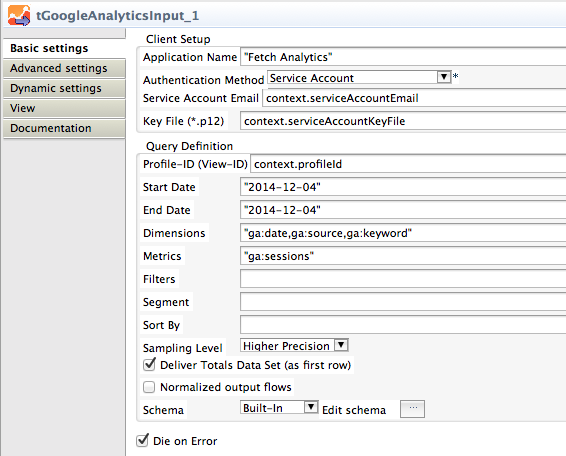
**Return values**

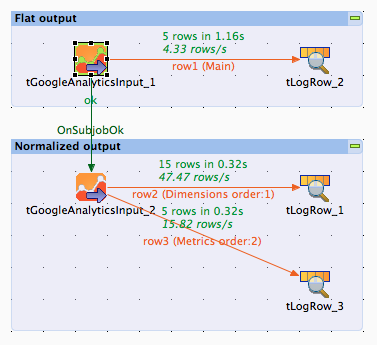
|  |  |
| --- | --- |
| **Return value** | **Content** |
| ERROR\_MESSAGE | Last error message |
| NB\_LINE | Number of delivered lines (only set if normalization is not used) |
| CONTAINS\_SAMPLED\_DATA | True if data are sampled, means not exactly calculated. This can happen if you query to many details. |
| SAMPLE\_SIZE | The amount of datasets used for the query |
| SAMPLE\_SPACE | The amount of available datasets for this query |
| TOTAL\_AFFECTED\_ROWS | Number of rows, which are collected by Google to calculate the result. |
| NB\_LINE\_DIMENSIONS | Number of normalized dimension records (only set if normalization is used) |
| NB\_LINE\_METRICS | Number of normalized metric records (only set if normalization is used) |
| CURRENT\_DATE | The value of the ga:date dimension (if present in the file) for every row. This value is only available in the “Normalized Flow” mode. Type: java.util.Date |
| ERROR\_CODE | The last error code as Integer. Default value is 0 at start. |

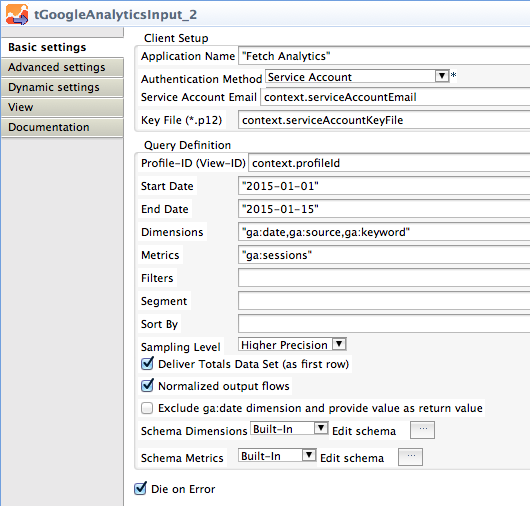
For more information about error codes please check this page:

<https://developers.google.com/analytics/devguides/reporting/core/v3/coreErrors>

**Scenario 1**

Using flat and normalized output in a test job.





The dimensions was set to: "ga:date,ga:source,ga:keyword", The metrics was set to: "ga:sessions"

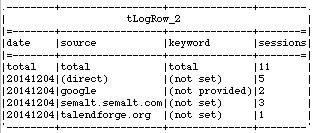
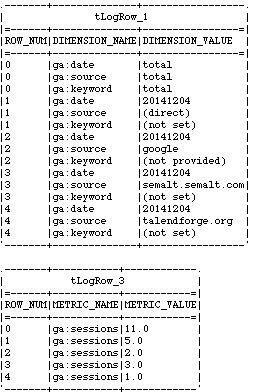
It is not necessary to know this configuration at runtime because the component recognizes the dimensions and metrics from the result set metadata.

The option “Exclude ga:date dimension and provide value as return value” allows you to run a report over a range of dates and save the values nevertheless normalized but treat the date in separate way. The value is in the return value CURRENT\_DATE of the component. This works only if you use normalized outputs.

(Off in this example)

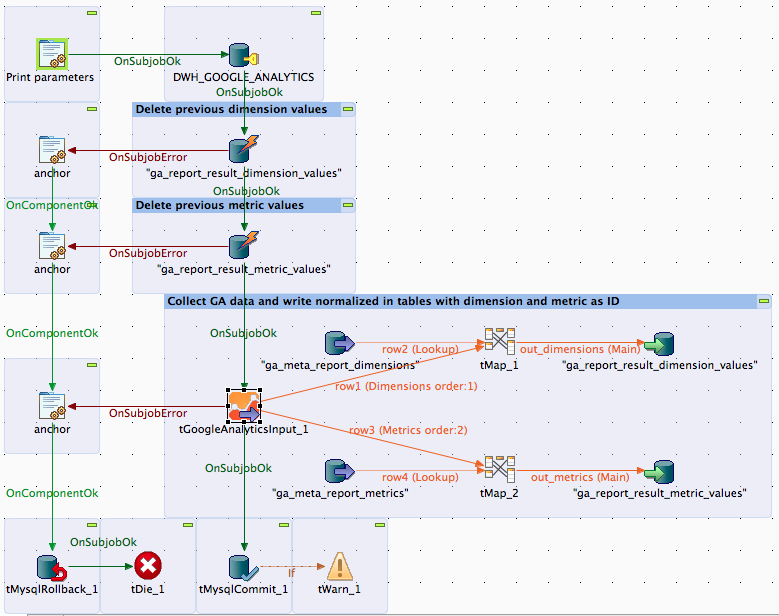
Here the outputs of the example test job above:

The output for the plain output The output for the normalized output

**Scenario 2**

Next a real live scenario for using the normalized output in conjunction with the usage of the meta-data (gathered with the component tGoogleAnalyticsManagement):



This job is designed to gather the data for one day and one report (a combination of a view, dimensions, metrics and filters very much like a custom report in the Google Analytics dashboard).

This job gets the view-ID, dimensions, metrics and filters as context variables and will be called, as much there are queries and dates to process.

The tMaps exchanges the dimension names and metric names with their numeric ids and adds a report-ID and the current date into the output flow for the database.

To get this job restart able everything is done within a transaction and the previous data for the report and the date will be deleted at first.

By the way, take note about the way to handle errors, this is very easy and avoid implementing the error handling multiple times. The anchors are tJava components without any code.

It is supposed to use gather the Analytics metadata to be sure you have access to all necessary data and to be able to build a star schema for the dimensions and metrics. Take a look at the component tGoogleAnalyticsManagement (today I would name it more like metadata but anyway).

**Configuration checklist**:

1. Is the email of the service account added to all relevant views (profiles)?
2. Is the system time of the host running the job synchronized with a NTP server?
3. Is the Google Analytics API enabled in the Google API Console?

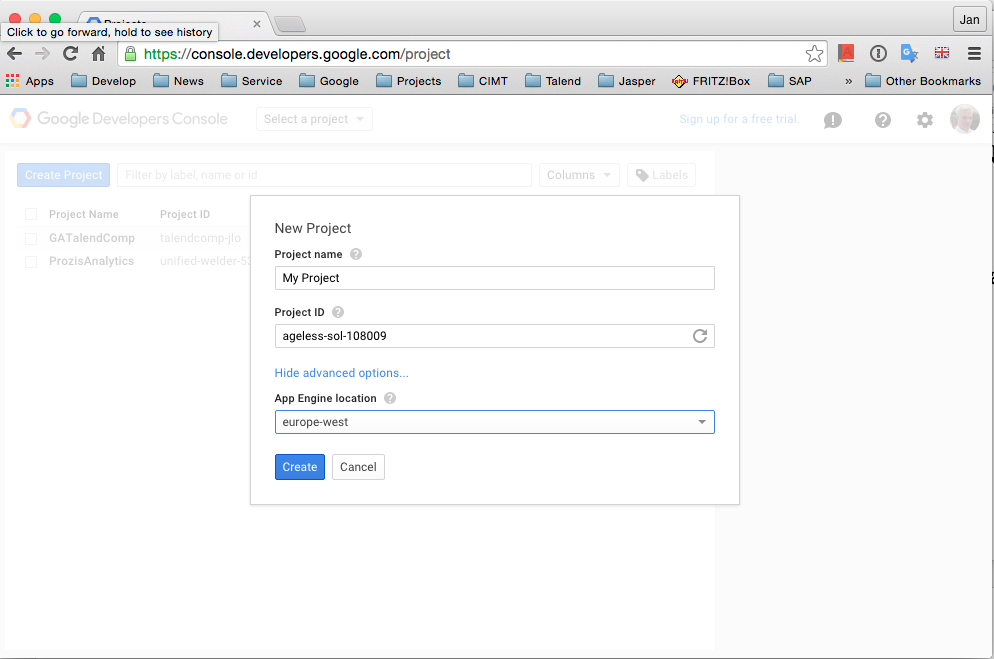
**Tip**:

Check your report at first in the Google Analytics API Explorer to get an idea if the data works for you.

**How to create a service account**

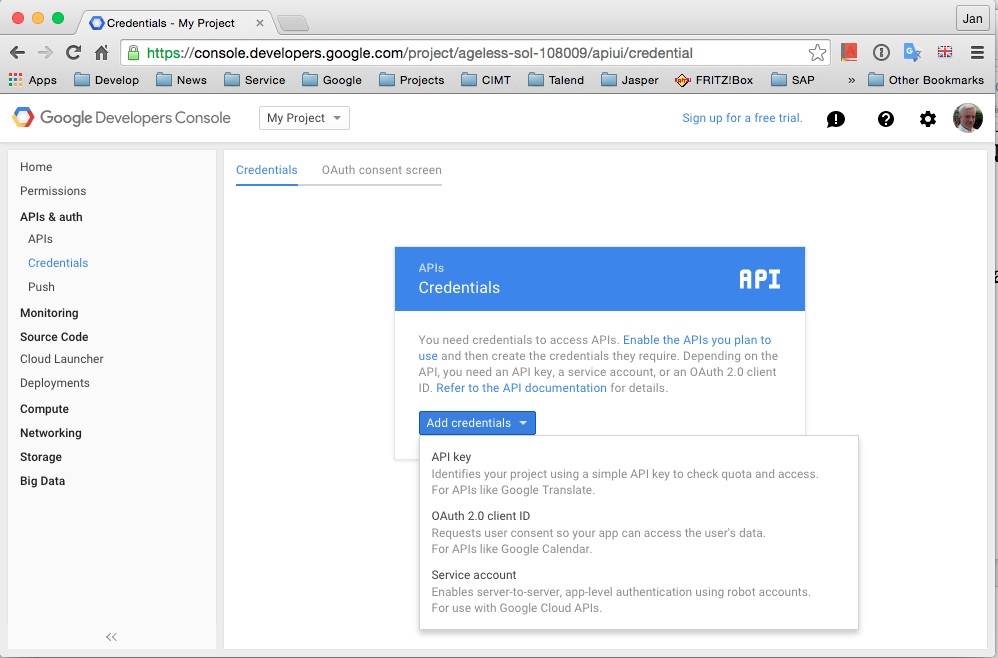
Service accounts have the advantage to not being personal - a big issue in automated processes. The service account use a key file to authorize and therefore do not need a password.

If you do not already have one, create a so called project in the Google Developer Console.

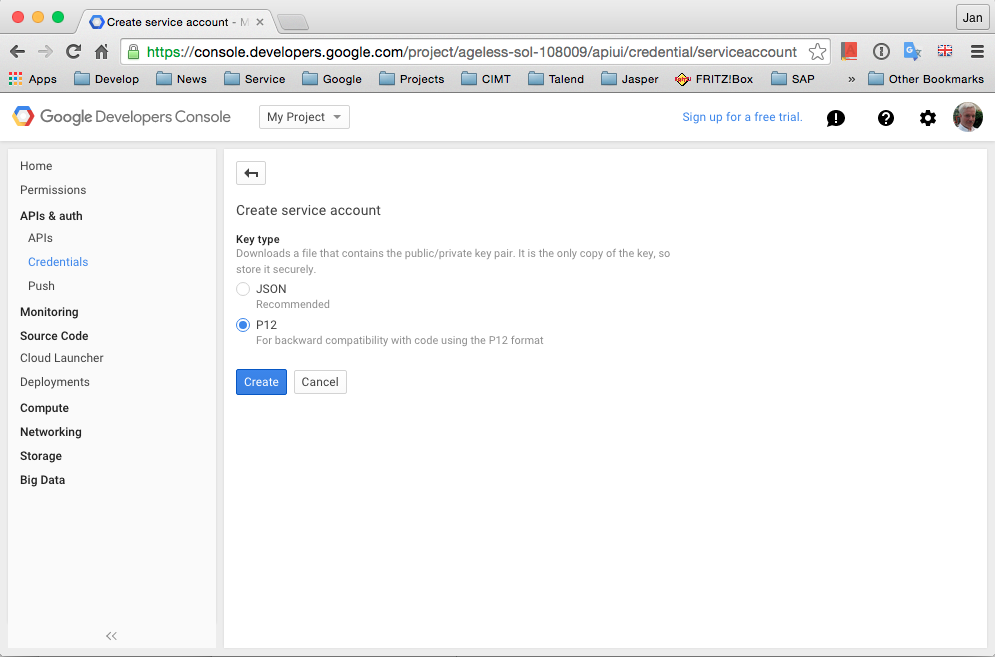


In the first place you do not see this options except the project name. Click at advanced options to expand the dialog.

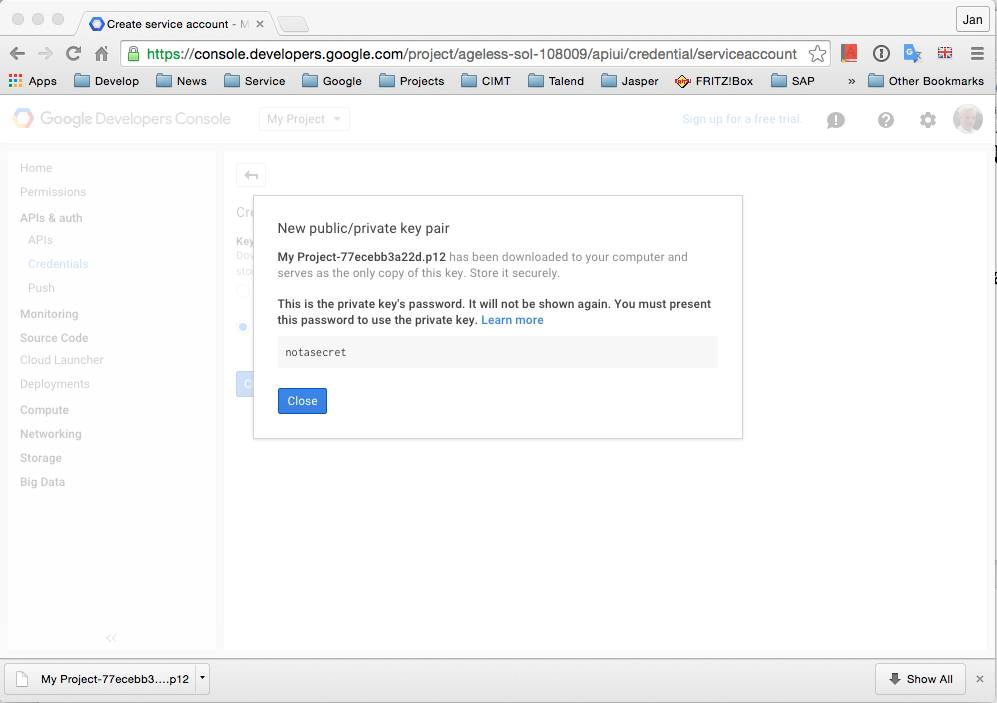
Now create new credentials ... we choose here the last option "Service account".



The current releases of the components works with the p12-format for the secured credentials.

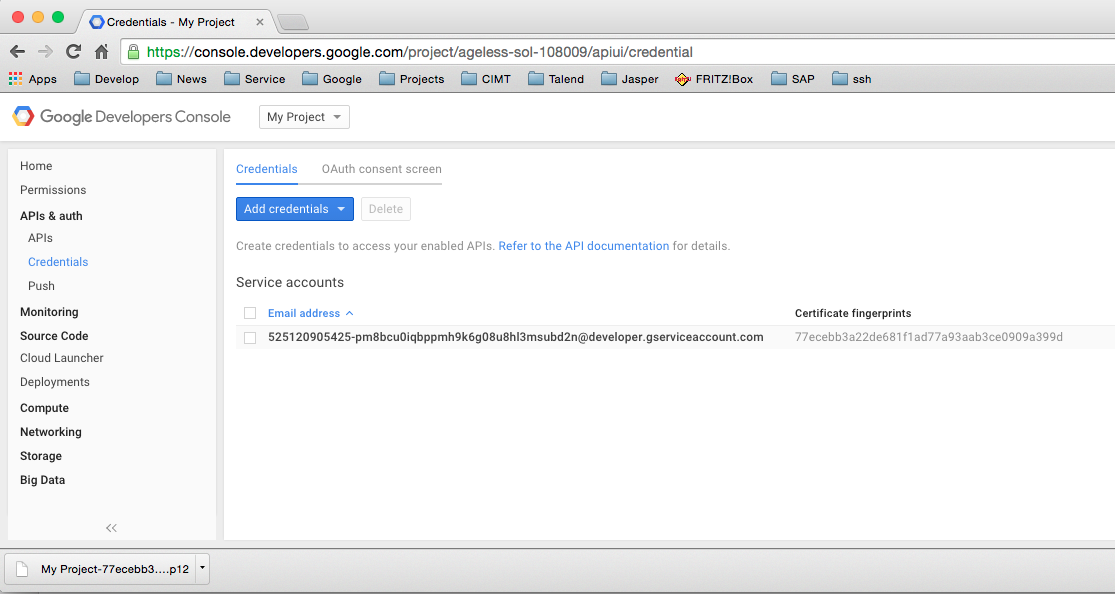


After the click on Create you will ask for saving the p12-file. Save it on a clear save place. You can move the file at any place you want. Later the components need to have read access to it.



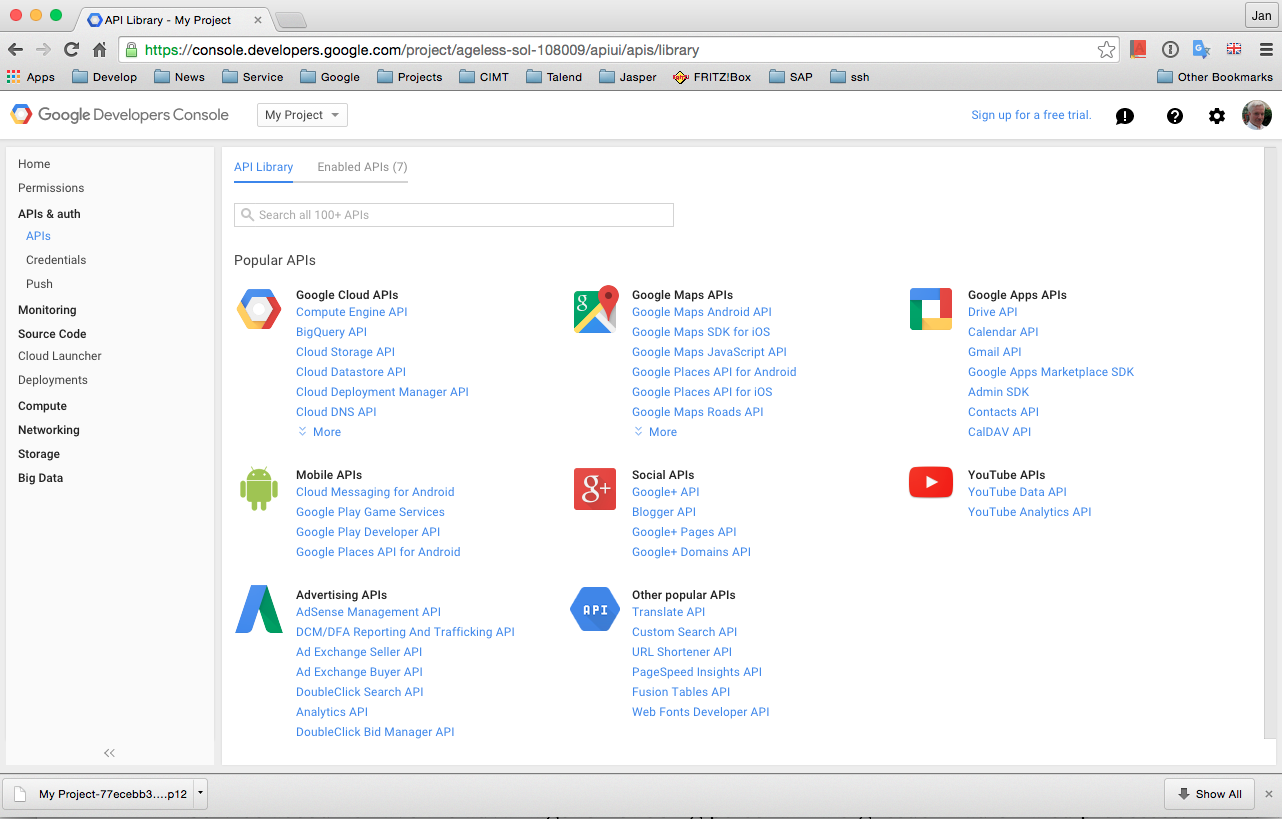
Once you have suceeded downloading the file, you see this sucess message.

Yo do not need the pass phrase because it is always the same and the component code knows this pass phrase and applies it internally.

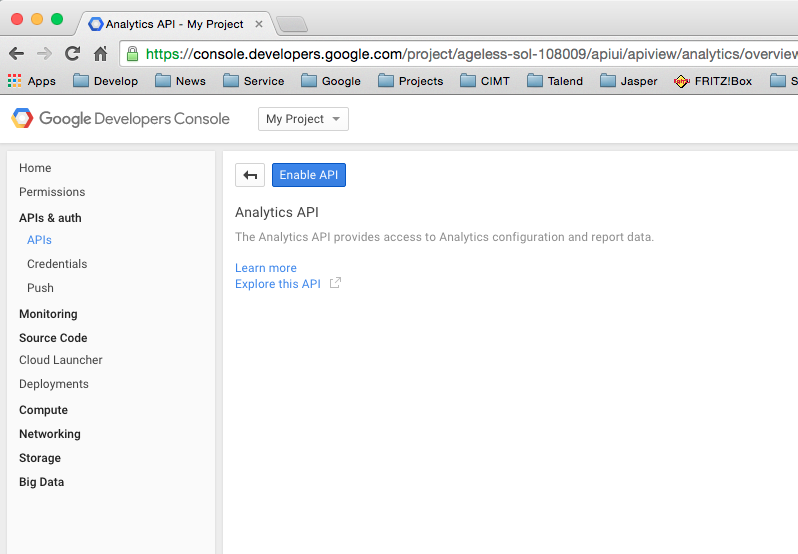


Now you can see the service account email address. This address must be added as user to the views, web properties or accounts (the authorisation will be applied to the lower level elements in the order account->web property->view).

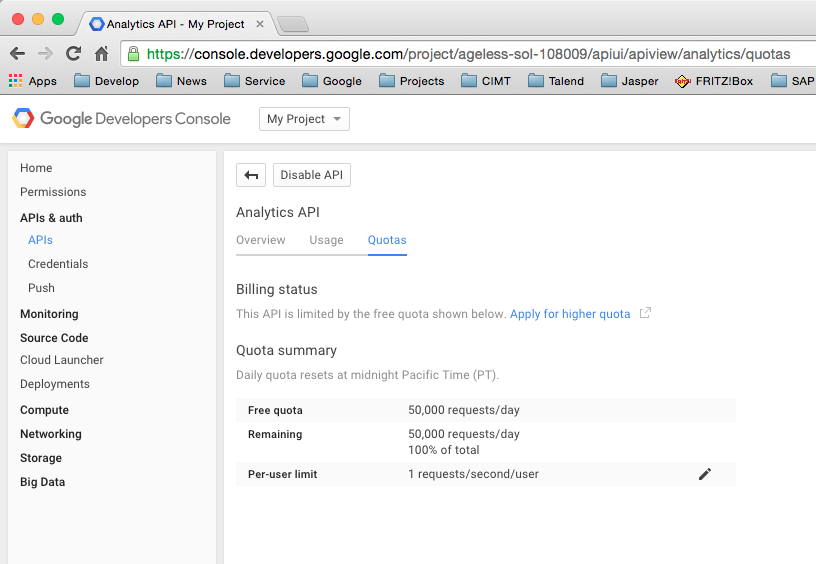
Now we have to enable the necessary APIs for the servive account needed.



It depends on the component what API you have to enable. For tGoogleAnalytics\* components you have to anable the Analytics API. In the detail section of an API you can also see the quotas (limitation of usage) you have. Here you can also order an enhancement if necessary.



This way you enable the Analytics API.



Here you can see the quotas.