Auteur: L.doubinine
Date: 07.9.2012
Enregistre le: 25.10.2012

En:Supertracking

This document is not ready yet. Please be cautious about information you can fin here. Please contact those peoples if you have any question,notes, or insults :

Loïc Doubinine , Nicolas Bérard , MF2 (Meetic France 2)

- **English :en:Supertracking**
- **French : fr:Supertracking**

Please update French version if you make improvement to this document. If you can't translate it, just copy the English version, someone will probably translate it later.

The two document must be sync if we want to be able to use them

Sommaire

Intro

Supertracking is a project that aim to report user activity on websites of meetic-corp. Like Google analytic, it works with a small GIF image of 1px² added into the page dynamical. This picture is non view-able and does not interfere with the user experience.

What is Supertracking

Supertracking allow data to come from the client's browser to us. Several data are reported like those :

Nom	Database Coll name	Type	Constraints	Is it calculated ?	Description
pid	PAGE_ID	String --> int	Mandatory	No	Unique identifier of a page (member/index, mailbox/read)
aboid	ABOID	Number	Length 11	Yes (PHP)	Customer aboid if present
abto	ABOID_TO	Number	Length 11	No	Aboid of the user to who the interaction ti from/to (For events,mails,chat...)
abton	ABOID_TO_ONLINE	Number	0 or 1 or 2	No	Is the user wich is interacted with is online or not ? 0=no, 1=yes, 2=unavailable/hidden
pr	Product Id	Number	-/-	No	A number to identify the product : MatchMaking(2),Dating(1)
d-src	-/-	String	-/-	Yes (strack.js)	Screen resolution of the customer (ex: 1600x1200, 1280x1024)
cid	CLICK_ID	Number	-/-	No	Like the pageId, it is a unique identifier corresponding to an action somewhere on the site.
ppid	PREV_PAGE_ID	String --> int	Length 11	Yes (strack.js)	PageId of the previous page
v1-50	VAL1-50	String	-/-	No	Custom values where to put data. The meaning depend of the pid and/or cid of the track.
p_sid	PRODUCT_SESS_ID	String	-/-	Yes(PHP)	Session ID on the product (GUEST)
co	CODE_ORIGIN	Number	-/-	Yes (PHP)	Marketing code mixed with mail code if present (mtcmk or mail)
scc	SITE_CODE_CONN	String	-/-	Yes (PHP)	Product site identifier
nl	NOLOG_MEMBERID	String	-/-	Yes (PHP/Strack.js)	Keyade Id
ot	OFFER_TYPE	Number	-/-	No	-/-
tr_oid	TRACKING_OBJ_ID	Number	-/-	No	-/-

crc	-/-	MD5	-/-	Yes(PHP)	The CRC is a MD5 of the field "pid"+salt. It allow us to check the validity of the track. If the check result is ok, then the track will be added and reference table updated if required. If the result is KO or CRC is not given, then the track will be added only if no update of the REFERENCE table is required.

Additional data depends of the page viewed. One page can have a data into V1 (the page result), and another have something else (the number of people returned). Each pid and cid can define his own data inside v1-50.

Track something

In order to make this process easier, I will describe what to do to track something on one website of Meetic. Maybe the process will be se same for Affinity but, for now, I don't know.

Prerequisite

To track a single static page, an unique **pid** is required. The **tpl_id** is used and you should not have to anything to make the tracking work for a new page. For security reason, the new pid might have not been inserted into the Reference Table in database. If it's the case, just make a request to add it.

Ref. table : X_STRACK.REFERENCE

Even if the entry is created automatically, you must make a request to set description of all data returned

- val1 desc (string) : Description of the data stored in the val1 coll
- val2 desc (string) ...
- ..
- val50 desc

For **cid** (click_id) or a layer's **pid**, it's mandatory to make an add request because security will not allow insertion of new values in the ref. table. Please give those data:

- cid (string)
- val1 desc (string) : Description of the data stored in the val1 coll
- val2 desc (string) ...
- ..
- val50 desc

Please do not forget to define in the Ref. table description of data you track. Analytics department may have difficulties to understand the meaning of your data.Who knows what is 1 and 3 ?

Track a page

Tracking a new page is done by itself. Nothing to do by developers/integrator ... on the source code.

If you face a single TPL that is used for multiple pages, and each need a single pid, then got read Split one TPL_ID into multiple PID.

Track a layer

There is many ways to track a layer

The layer code is located in footer.htm

In this case, the layer is probably inside a "section" witch is displayed as needed. When the section showing the layer is displayed (every time), you should just add the script tag as follow :

```
<script>TK.display('pid=layer_reactivate_profile');</script>
```

If displaying the "section" is not enough, and JavaScript action is required (display:none) then look to the next way.

JavaScript tells the layer to be displayed or not

In this case, add the following code everywhere you have the code that actually displays the layer (One place only I suppose) :

```
TK.display('pid=d6_layer_cool');
```

Track a click

Two possibilities to track a click.

Track by "onClick"

Add inside **onClick** attribute the following code :

This method is not implemented yet

Simple track without JavaScript

Add **data-strack** attribute on html node the user clicks on.

```
<a href="http://google.fr" data-strack="e=click&cid=goGoogle&{{TK_QUERY_FULL}}">Google
```

If the node structure is complex, add the attribute on every node that can be clicked by the user (event if there is 1px border !!!)

```
<div class="BigCase" data-strack="e=click&cid=goGoogle&{{TK_QUERY_FULL}}">
   <a href="http://google.fr" data-strack="e=click&cid=goGoogle&{{TK_QUERY_FULL}}">Goo
</div>
```

Don't forget TK_QUERY_FULL and the "&".If forgot, the track will be rejected

Track an event

To track an event (flash,mail...), just add the following code where you generate the event call.

```
TK.event('eventName',aboIdTarget,AboidConnected?) ;
```

```
TK.event('eventName',12345678,1);
```

For more details on how to use this method, read [library window.TK (Supertracking.js)]

Track something else

To track something new, just create a new method inside the supertracking.js library like explained at [a method to the lib supertracking.js]

Test Tracking

To test that you are tracking the right thing and correctly, you need to check in database that the data is correctly inserted. See with Analitycs department to do that.

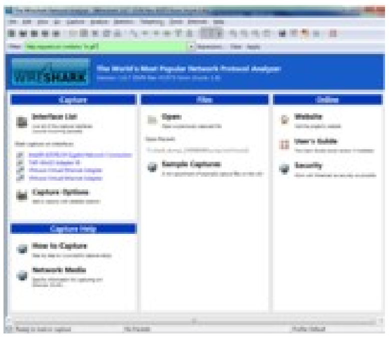
To only test what is sent by the website and do some debug during dev, you can use wireshark as described below.

Wireshark

Download and install Wireshark : [page on Wireshark website]

Do "Next" every times, even to install WinCap witch is required. Wireshark is free.

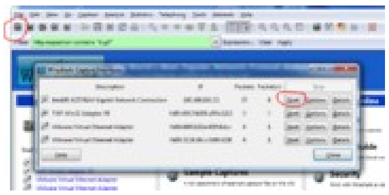
Run wireshark



Wireshark init
As you can see on the screenshot, copy the text below inside de "filter" field and hit "apply". The background of the field must be green. If not, the filter is not recognised

```
http.request.uri contains "tr.gif"
```

Start the recording



Wireshark start
click on the icon cercled, and hit start on the row with number that grow quickly.

Read results

After using the website, visiting some pages, sending some events, you can see results in the main window of WireShark.

Here an example :

Wireshark results
As you can see, the query-String sent to the tracking server can be read in two places. In the list, and in the details. All data are reported and you can check if yours are here as expected.

Lines are shown in real time but the website can keep a track for 5 seconds before sending it. So be patient and wait 10s max. If nothing shows up after 10s, something went wrong!

Integration

On meetic, supertracking was implemented and enabled. To do so, we had to work on php libs, js libs, templates, and webv6/pages.

The Tag

For supertracking to be available on any pages, a Tag must be added. This tag is a JavaScript file call that can fallback to a flat image node of the GIF in case there is no JavaScript enabled. The dating tag code :

```
<script id="strackTag" type="text/javascript" data-strack="{{section:s_TK_list}}o.{{TK
opts.url='{{TK_IMG_URL}}';
opts.hooks=[
['eventTrigger',window.TK.hookStrackAddppid],
['eventTrigger',window.TK.hookStrackAddTkQueryFull],
['event',window.TK.hookStrackAddppid],
['event',window.TK.hookStrackAddTkQueryFull],
['pageData',window.TK.hookStrackPageData],
['storeData',window.TK.hookStrackStoreData]
] ;
"
        src="{{TK_JS_URL}}?v={{GLOBAL_V_CSS_VERSION}}" ></script>
<script type="text/javascript">
    window.TK.TK_QUERY = '{{TK_QUERY}}' ;
    window.TK.TK_QUERY_FULL = '{{TK_QUERY_FULL}}' ;
    window.TK.ready = true ;
    window.TK.TK_TPL_ID = '{{TK_TPL_ID}}' ;
</script>
<noscript>
        <img src="{{TK_IMG_URL}}?{{TK_QUERY_FOOTER}}" width="1" height="1" border="0"
</noscript>
```

Wich gives this after Yats processing :

```
<script id="strackTag" type="text/javascript" data-strack="o.aboid = '40053785';o.pr =
opts.url='http://mdhahbi.tk.ilius.net.dev/tr.gif';
opts.hooks=[
['eventTrigger',window.TK.hookStrackAddppid],
['eventTrigger',window.TK.hookStrackAddTkQueryFull],
['event',window.TK.hookStrackAddppid],
['event',window.TK.hookStrackAddTkQueryFull],
['pageData',window.TK.hookStrackPageData],
['storeData',window.TK.hookStrackStoreData]
] ;
"
        src="http://ldoubinine.stda.ilius.net.dev/js/core/main/supertracking/strack.js
<script type="text/javascript">
    window.TK.TK_QUERY = 'pr=1&co=618063&scc=PO&aboid=40053785&p_sid=01__3_8277272a8ef
    window.TK.TK_QUERY_FULL = 'pr=1&co=618063&scc=PO&aboid=40053785&p_sid=01__3_827727
    window.TK.ready = true ;
    window.TK.TK_TPL_ID = '/index' ;
</script>
<noscript>
        <img src="http://tk.ilius.net.dev/tr.gif?aboid=40053785&pr=1&co=618063&scc=PO&
</noscript>
```

The Tag has many Yats properties. They are managed by the supertracking php lib.

- {{TK_NAME}} et {{TK_VALUES}} are arrays witch contains list of parameters given to supertracking. All standard parameters of "one page track" are listed. In the example above data was :

```
o.{{TK_NAME}} = '{{TK_VALUES}}';
o.aboid = '40053785';
o.pr = '1';
o.co = '618063';
o.scc = 'PO';
o.pid = '/index';
o.p_sid = '01__3_8277272a8ef1cc2bcce4ea7aebd19890';
o.e = 'display;
```

- {{TK_IMG_URL}} is the GIF url like http://tk.ilius.net/tr.gif.

* {{TK_JS_URL}} is the URL of strack.js file. For now, the file is hosted on Meetic, but in a near future, the file should be hosted by **tk.ilius.net**. Meetic and Affinity would have the same URL for the exact same file.
* {{TK_QUERY}} & {{TK_QUERY_FULL}} are Query-string (name=value&name=value...) with all parameters listed inside arrays TK_NAMES and TK_VALUES. Only TK_QUERY_FULL have the parameter named **pid**. TK_QUERY does not have it at all.
* {{TK_TPL_ID}} is the pid of the current page
* {{TK_QUERY_FOOTER}}is the same as {{TK_QUERY_FULL}} with **e=display** added. Not very useful ,maybe it should be removed now. (Check the way ppid is calculated before)

JavaScript of the tag

The tag calls JavaScript function that are in Meetic Supertracking library.This library is specific to meetic, and should be duplicated on each product (Affinity...) and customized. The namespace is **window.TK**. "Hooks" can be different from one product to another. On Meetic it manages **ppid**.

The file supertracking.js wich contains window.TK must be loaded before the Tag.

Php library

The Php library can be found in those files

* Meetic

  1. libs/product/supertracking/supertracking.php
  2. libs/display/supertracking/supertracking.php

Methods below are included :

superTracking_isTrackingEnabled()

Return TRUE if supertracking is enabled according to the two way of activating/deactivating it.

1. config/tracking.conf.inc => Constant SUPER_TRACKING_ENABLE set to TRUE. If FALSE, Supertracking is globally disabled for all website.
2. product/site/.../config => ['SUPERTRACKING']['ENABLED'] set to TRUE. If FALSE (by default) then Supertracking is not enabled in web Sites targeted by the configuration file.

superTracking_assignValue($_name,$value)

Add a property to arrays {{TK_NAME}} and {{TK_VALUES}}
Use it everywhere it is useful.

There is one restriction very important : It's forbidden to set property named as V1 to v50 outside webv6 folder. You can't set v1-50 inside lib folders.

This warning is very important because if you do not follow this restriction, conflict on V1-50 will occur.

DO NOT SET V1-50 INSIDE LIB FOLDER UNLESS YOU KNOW WHAT YOU'RE DOING

superTracking_setYatsProperty()

It is the function to execute on every pages in order to initilize supertracking. This function MUST be exectued juste before the render of the page. On meetic it's done inside MPP.inc in **appli_echo_result**.

Split one TPL_ID into multiple PID

Each standard page have one template associated with it but sometimes, multiple functions use the same TPL_ID and need to have a separated pid. The easy way to see that is to watch the BAL (internals mails of Meetic website) pages. Each one have a particular function (Read, Write,Reply...) that share the same tpl_id.

To be able to have multiple pid, two things are mandatory :

- Write the php code that is able to make the distinction between each pages/functions
- Declare the fonction into the extensionConfirguration file of Supertracking

In product/supertracking/extensionsConf.inc are set all extensions. The function **superTracking_getExtensionConf** return the array that tell all cases. Change it to add to your TPL_ID you want to modify. Put the name of the function that do the job, and the file path to include it correctly when needed.

The function must meet the pattern below :

```
function superTrackingTemplateExtension_nameOfTheFunction(&$_tplId){
   $_tplId += '/'.$_GET['param'] ; //This is only an example !
}
```

For now, parameters are references, and you must write directly into the variable. The returned value is not used at all.

If you want to see some examples, check the configuration file and look at some functions like those of the BAL part.

Configuration

To tweak Supertracking there is some configurations to know.

Environment Configuration

In the file 'config/tracking.conf.inc'

- SUPER_TRACKING_ENABLE => Enable/disable Supertracking globaly.
- SUPER_TRACKING_JS_URL => URL of the **strack.js** file.

  The URL can be absolute (http://tk.ilius.net/strack.js) or relative (core/main/supertracking/strack.js). The relative case mean hosting the file by the product like it's done by Meetic today.

- SUPER_TRACKING_IMG_URL => URL of the GIF image. As SUPER_TRACKING_JS_URL the URL can be absolute or relative.

Site Configuration

In the file config.inc and private.inc, it is possible to configure this :

- ['SUPERTRACKING']['ENABLED'] => Enable/Disable supertracking for website targeted by the configuration file. By default it's false.

Library Configuration

The Php library need to be updated in case you implement supertracking on a new product (Affinity ?).

- lib/product/supertracking/supertracking.inc => STRACK_PRODUCT_ID => Change it by the ProductId required.It's 1 for Meetic and *2 for Affinity*[réf. nécessaire]

JavaScript Tracker (Strack.js)

This part should be common for meetic/Affinity. By common, I mead exactly the same file at the same URL for each product.

When Strack.js is loaded, it allow us to track anything on the page.

data-strack

To track a click, adding the correct attribute to the node is the only thing required.

```
<img src="url" data-strack="e=click&cid=cross_selling_top_right&{{TK_QUERY_FULL}}" />
```

The attribute content must be the complete Query-String with all data to send to supertracking servers. Please Use {{TK_QUERY_FULL}} and don't forget the "**&**" character.

A click tracked almost always report a **cid** value.

Due to some limitation, a click track of a complex node structure need us to add the attributes to all nodes and sub-nodes.

```
<div class="thumb" data-strack="e=click&cid=cross_selling_top_right&{{TK_QUERY_FULL}}"
  <span class="title" data-strack="e=click&cid=cross_selling_top_right&{{TK_QUERY_FULL
  <div class="pic" data-strack="e=click&cid=cross_selling_top_right&{{TK_QUERY_FULL}}"
    <img src="url" data-strack="e=click&cid=cross_selling_top_right&{{TK_QUERY_FULL}}"
  </div>
</div>
```

This limitation is due to the way click are connected in JavaScript on the page. Event Delegation is used instead of the **live** method of J Query. This choice is for performance reason, they are drastically not the same.

Track js

Strack is a low-level library. It mean that it should not be used directly on the website as-is. Use Supertracking.js, window.TK or simply TK.

Hooks

Strack allow us to define Hooks. This is to do some data manipulation directly inside the lib without change it. Many level of the lib process can be tweaked by this way. On Meetic, Hooks are used to add automaticaly the ppid to the tracks sent to the servers.

To add a Hook, it must be registered inside the Tag. Every declaration is done in two part

- Hook's name (String)
- Callback Funciton (ref Function)

```
opts.hooks=[
['eventTrigger',window.TK.hookStrackAddppid],
['event',window.TK.hookStrackAddppid],
['event',window.TK.hookStrackAddTkQueryFull],
['storeData',window.TK.hookStrackStoreData]
]
```

It is possible to add many callback to the same hook, execution will follow the registration order. In the above example **hookStrackAddppid** then **hookStrackAddTkQueryFull** for the hook named **event**

Hook's names are free (look source code of Strack.js) but are usually internal functions names. Hooks are executed at the beginning of the function.

To make a Hook function, follow this two rules:

- The only parameter is a string (the Query string of the track)
- The function return a query-string. At least the same as the input if noting is done inside the function.

Hook Example : **hookStrackAddppid**

```
obj.hookStrackAddppid = function (params) {
      params = params + "&ppid=" + getLastPid();
      return params;
};
```

JavaScript library window.TK (Supertracking.js)

The Library 'window.TK' allows to centralize common operation in order to make tracking life easier in JavaScript.

The file is located in staticv6/js/core/main/supertracking/supertracking.js on Meetic

The file supertracking.js must be loaded at the very beginning or at least before the first call of one of it's methods

Inside a Browser, in JavaScript, window.TK can be shortcut-ed as TK. The global namespace of an HTML page is always window.

```
window.TK.event('flash'); <==> TK.event('flash');
```

The library contains common methods for common use.

- **TK.display(queryString)** and **TK.displayFull(queryString)** => Track a display like the tag do it by itself. Very usefull for layer displayed with JavaScript.

  TK.TK_QUERY et TK.TK_QUERY_FULL are added automatically according to the function name.

```
TK.display('pid=mapage/souspage/sousouspage');
```

- **TK.trackFieldError(field)** => Track an error in signup forms. (Should this method be more generic?)
- **TK.trackUploadPhoto(photoId)** => Track a photo upload (Same note as above)
- **TK.event(eventId[,abto[,abton]])** => Track an event (Flash,yes,fav...)

  eventId is a string representing the event (event/flash,event/yes...)
  abto and **abton** are not mandatory.

  - **abton** => Number/String

    - 1=connected
    - 0=disconnected
    - 2=not available/hidden
  - **abto** => **aboid** of the user who receive the event.A flash is always between two aboid !

```
TK.event('flash',12345678,0);
TK.event('flash','{{member_aboid}}','{{section:s_list_member_online autohide="yes" par
```

Add a method to the lib

To add a method to the lib, first look how existing functions work, they are very simple. On thing is mandatory/required/not forgettable/ the use of **onReady**. This function just wait for the low level lib **Strack.js** to be available, then do the track.

OnReady is also the function that stop supertracking calls when supertracking is not enable.If **window.TK.ready** is not set to true (or not set at all), then no supertracking will occur. Even in this cass, methods of TK are allowed and usable, they just don't do the job.

Usage

The lib must be loaded at the very beginning. This allow us to call any methods any time in **JavaScript**, **onClick**, **<Script> tag**. Tracking will be done when all is ready.

Color meaning

- Constants : CONSTANTE
- File Path : Chemin/Vers/Le/Fichier.ext
- Environments configuration : NB_PROFILE_COMPAT
- Site configuration : ['GLOBAL']['NG_RESULTS']
- Yats property : {{Ma_Propriete_Yats}}