ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

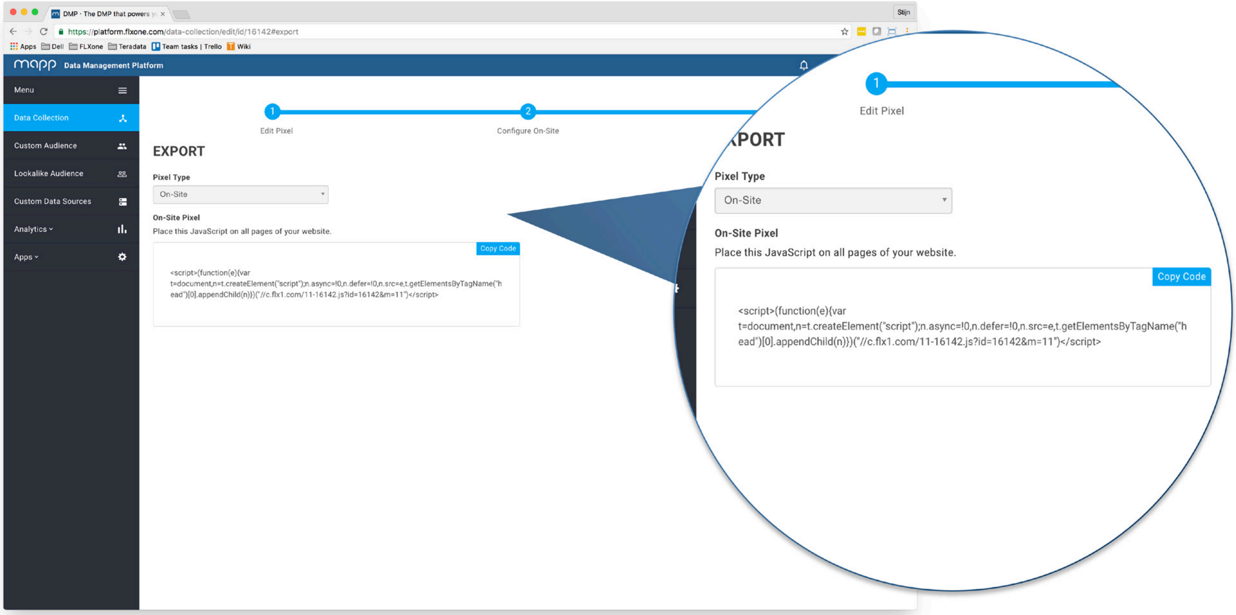
INTRODUCTION

Collecting data with the On-Site pixel(s) is fairly easy because it is just one line of JavaScript code and the implementation is one-off. The latter means that once the pixel is exported and implemented on the client's website it never has to be changed or replaced. Any configuration settings can be applied from within the DMP user interface and doesn't require a regeneration of the code. Moreover, after hitting the save button the On-Site pixel will automatically be redeployed within a couple of seconds to take all modifications into consideration. The script automatically collects pageviews and identifies valuable user intent patterns and it can serve as a container tag.

PIXEL IMPLEMENTATION

A step-by-step guide on how to implement the on-site pixel:

1. Go to *Data Collection* > Create page in your DMP account.
2. Provide the *Pixel Name* and enable the relevant *Modules*.
3. Click on the *Save and Export* button.
4. Select *On-Site* from the Pixel Type dropdown and click on *Copy Code.*
5. Place this code (Javascript) on your website. For an easy implementation we advise to put it in a centralized <head>*Pixel*</head> or <body>*Pixel*</body> section. An example of the code is displayed below.



EXPORT

Pixel Type

On-Site

On-Site Pixel

Place this JavaScript on all pages of your website.

```
<script>(function(e){var
t=document,n=t.createElement("script");n.async=!0,n.defer=!0,n.src=e,t.getElementsByTagName("h
ead")[0].appendChild(n)})("//c.flx1.com/11-16142.js?id=16142&m=11")</script>
```

Common Questions

What is the best way to forward the Pixel code to the person who is responsible for the actual implementation?
In case you need to send the pixel to an internal developer or to an external tech department we strongly recommend to copy/paste the pixel into a notepad file and add it as an attachment.

How to deal with the pixel organization when the client has multiple websites/subdomains?
We strongly recommend to create separate pixels per website/subdomain. From a practical point of view this will allow the client to benefit from efficiencies on DMP features since the Pixel ID is the main global identifier across the system. For example, restricted accounts can be created by permitting access to specific pixels and all reporting features contain filters on Pixel ID to easily gather stats. From a technical perspective it will gain efficiencies in terms of data processing because the Pixel ID is an integer whether filters on contextual attributes like Site Domain and Event URL are based on strings.

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

What is the difference between a JavaScript and Image pixel?
JavaScript pixels can execute code, images not. You should always use the JavaScript pixel in case you want to track interactions or fire additional external pixels (piggyback). Note that the configuration of DMP Custom Event and/or Custom Success Event requires the JavaScript on-site pixel. You should only use image pixels when there is no option to use JavaScript pixels or for purposes were JavaScript is disabled (e.g. emails).

Can the On-Site Pixel be placed into a container tag?
Yes, our pixel can be deployed through all tag managers. This includes but is not limited to Google Tag Manager, Qubit, and Tealium.

Does the DMP Pixel impact performance of the website?
No, all our pixels load asynchronously. Because of this our pixels do not affect the loading time of a website. The asynchronous loading time should be less than one second under normal conditions. You can always inspect the loading time in the console of your webbrowser.

Can we implement the pixel on mobile Apps and collect mobile specific attributes like Device ID?
Yes, this is possible even without the need of a Software Development Kit (SDK). In order to track Device IDs or any other mobile specific attributes the On-Site pixel should be implemented in a mobile app and the instructions below needs to be taken into account by an app developer. Given the example: http://go.flx1.com/px?id=1234&m=11&t=gif&sitid=2&lat=$latitude&lon=$longitude&devid_g=$GoogleAID&devid_a=$IDFA, the values for the mobile-specific parameters (respectively Latitude, Longitude, Google Ad ID and Apple Advertising Identifier) needs to be dynamically populated by using Android (java) and in Swift/Objective C. Please note the following:

- The parameter “devid_g” is only relevant for Android devices whether the parameter “devid_a” is only relevant for iOS devices.
- The parameters “id”, “m” are required and automatically generated when exporting a pixel.
- The “t” parameter indicates the pixel type and can be either an image pixel (gif) or a script tag (js).
- The “sitid” parameter indicates the platform and should always be “2” in case of a pixel that is implemented on a mobile app.

PIXEL CONFIGURATION

Within the DMP interface, a new or existing On-Site Pixel can be well-tailored in order to fire the relevant data events and collect the required data attributes according to the client’s use case(s).

On-Site Modules

An On-Site pixel module can be described as an optional and predefined module (script) that runs on top of the default code in order to track specific user interactions and/or capture meaningful data attributes. Once enabled, it will automatically be added to the On-Site pixel code which initiates the execution of it without the interference of a developer or without the need to change the implemented code. An overview of the various modules as well as the corresponding categories can be found below.

Category	Module	Description	Default
Interaction Tracking	Time on Page	Captures the time a user has been on any given webpage up to 3 minutes with the following intervals; 5, 10, 20, 30, 60, 90, 120, 180 seconds.	Yes
Interaction Tracking	Page Scroll	Captures the scroll depth on any given webpage up to 100% with the following intervals; 10, 25, 50, 75, 100 percent.	Yes
Interaction Tracking	Webform Interactions	Captures whether the user has interacted with any webform on any given webpage.	Yes
User Data	Webform Input	Captures and stores the value a user has entered in any webform on any given webpage after an inactivity of 3 milliseconds.	No
User Data	Browser Data	Captures and stores the screensize of the monitor and browser (dimensions: width and height) as well as some other attributes.	No
User Data	User Details	Captures and stores the user agent.	No

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

Category	Module	Description	Default
User Data	Hashed Email	Captures and stores the email address entered by a user anywhere on the webpage (based on 'at-sign' identification). The address will be stored in 3 hashed formats; MD5, SHA1 and SHA256. **Common Identifier** The hashed email address could serve as a common identifier to unify CRM data with behavioural on-site data.	No
Website Data	Average Price on Page	Captures and stores the average price of products in a shopping cart. **How does it work?** We scan the visible elements on the page for one of the following strings: '€', '£', 'EUR', 'USD', 'GBP', '$'. We identify the string that occurs most and then scan the page again for that string followed by any number with or without a decimal. The decimal point can be both '.' and ','. Then we take the first 100 matches and strip and attempt to parse them as a number (so remove any formatting). Then if the number is higher then 0.01 and lower then 100000 we add it to the total price of the page. Then as a last step we determine the avarage by dividing the total by the number of prices we have summed.	No
Website Data	Clicked Elements	Captures and stores the click-interactions with elements that are identifiable on a webpage. **How does it work?** We track clicks on any element on the page and store the CSS path to this element (e.g. #contentArea > div.wrap:nth-child(1) > div.newsArticleColumn:nth-child(2)). Please note that we cannot track clicks inside iFrames so a click on an iframe will be tracked but it will not send the specific unless the iframe is on the same domain of the page.	No
Website Data	Common Words	Captures and stores the top 8 common words on a given webpage. The module applies auto-determination based on a general word count.	No
Website Data	Qubit Universal Variable	Captures and stores the ID of the Qubit Universal Variable retrieved from the Qubit pixel that should be implemented on the website as well.	No
Website Data	Google Analytics Ecommerce	Captures and stores the Order ID and Total Revenue value retrieved from the Google Analytics Ecommerce pixel that should be implemented on the website as well.	No
Website Data	URL Parameters	Captures and stores the key and value from any URL parameter that is part of the event URL. **How does it work?** At the time the On-Site Pixel fires on a certain wegpage of which the event_url includes URL parameters it will parse the part after the question mark and automatically stores the key and value of the detected attributes. Given the following example URL it will thus store 3 additional custom attributes (utm_source, utm_medium and utm_campaign): http://mapp.com/landingpage?utm_source=adwords&utm_medium=search&utm_campaign=brand_awareness.	No
Integrations		This reflects additional and conditional modules that are only relevant when certain integrations are activated.	No
*Please note that the Webform Input module will collect and store every value that have been entered in any webform on the given website. This may result in the collection of Personal Identifiable Information (PII), such as the full name, address, phone number or even all credit card details. Therefore, be careful with enabling this module and always ask the client for permission or check the legal implications first.			

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

CUSTOM (SUCCESS) EVENTS

Apart from the default Pixel settings and the possibility to enable or disable specific On-Site pixel modules you can also configure custom events that are only relevant for the given client. There are a variety of places on a website where you may want to measure specific user interactions that won't be tracked by default. The DMP allow you to capture click, gesture or behavior by visitors to your website by means of Custom Events and Custom Success Events that are configurable through the platform's web interface. With the 'Populate Custom Data' feature you can even collect the value of a specific element and store it as a custom attribute. It helps you to understand how users are interacting with your website and they can serve as input to define advanced Custom or Lookalike Audiences.

- Session Engagement:
  label users according to the number of page visits or total lenght of a particular browser session.

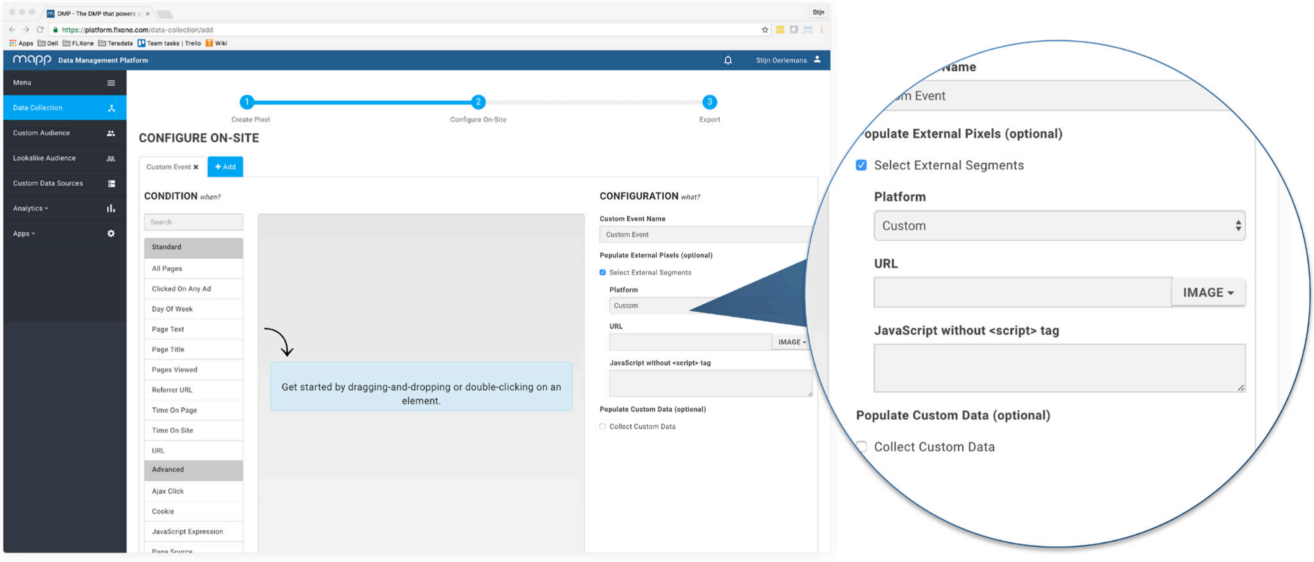
- User Preferences:
  label users that are searching for products of a particular brand or narrowing down the scope of hotel search results based on the stars.

- Website Interactions:
  label users that are downloading content, forwarding an article via email or social media or commenting on a blogpost.

- Ecommerce Details:
  label users according to their stage in the buying process of an ecommerce cycle. It also allow you to capture the total amount of their shopping cart.

- Account Details:
  label users according to their demographics available from their account profile. It also allow you to capture the information a user provides in a specific input form during the sign-up or subscription process (e.g. Client ID) or their current account balance.

Populate External Pixels
When creating a custom event or custom success event it is also possible to populate external pixels as an optional configuration setting. All type of pixels (e.g. image url, script, etc.) are supported and can be piggybacked onto the corresponding custom event or custom success event. The term piggybacking usually refers to the situation where the firing of pixel A causes the firing of pixel B. This second firing can either be via a redirect or a server-side firing. Piggyback pixels may be used for tracking custom success events in secondary systems like a DSP for instance.



CONFIGURE ON-SITE

CONDITION *when?*

Get started by dragging-and-dropping or double-clicking on an element.

CONFIGURATION *what?*

Custom Event Name: Custom Event

Populate External Pixels (optional)

☑ Select External Segments

Platform: Custom

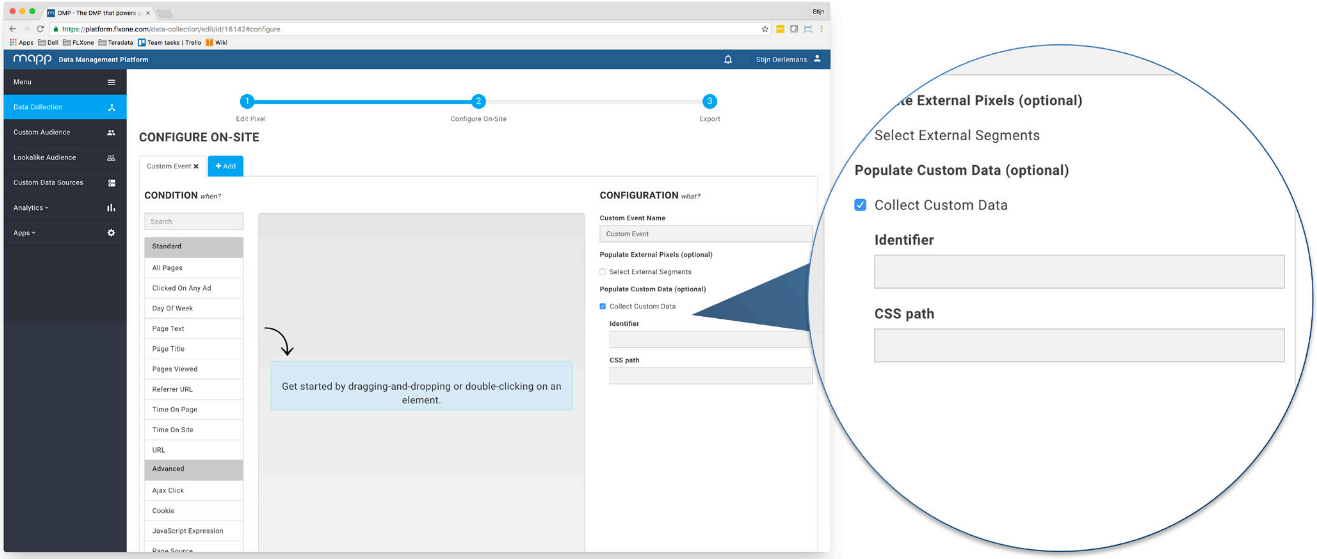
URL:

JavaScript without <script> tag:

Populate Custom Data (optional)

☐ Collect Custom Data

Populate Custom Data
Because every website has its own content and structure it is important to customize the DMP On-Site Pixel according to your client needs. The Custom Data feature enables you to capture relevant elements from a specific website/webpage that are available from the source code or even from user generated content and send it to the DMP in real-time. This enables you to store highly relevant data attributes that can be used to create advanced Custom Audiences.



DMP - The DMP that powers y… ×
https://platform.flxone.com/data-collection/edit/id/16142#configure
Apps | Dell | FLXone | Teradata | Team tasks | Trello | Wiki

mapp Data Management Platform — Stijn Oerlemans

Menu: Data Collection, Custom Audience, Lookalike Audience, Custom Data Sources, Analytics, Apps

1 Edit Pixel — 2 Configure On-Site — 3 Export

CONFIGURE ON-SITE

Custom Event ✖ | + Add

CONDITION *when?*

Search

Standard: All Pages, Clicked On Any Ad, Day Of Week, Page Text, Page Title, Pages Viewed, Referrer URL, Time On Page, Time On Site, URL

Advanced: Ajax Click, Cookie, JavaScript Expression, Page Source

Get started by dragging-and-dropping or double-clicking on an element.

CONFIGURATION *what?*

Custom Event Name: Custom Event

Populate External Pixels (optional)
□ Select External Segments

Populate Custom Data (optional)
☑ Collect Custom Data

Identifier

CSS path

Common Questions

Do I always need to configure a Custom Event with an ‘All Pages’ condition in order to collect data?
No, this is not required. Once the On-Site pixel is correctly implemented it will fire on each and every page independent of any custom events or custom success events configured. Therefore, many data attributes will already be collected by default. Consider the custom events and custom success events as labels to easily group your visitors according to certain specific behaviour. Please note that every successive pixel request originating from a custom events or custom success event will be counted as an additional data event.

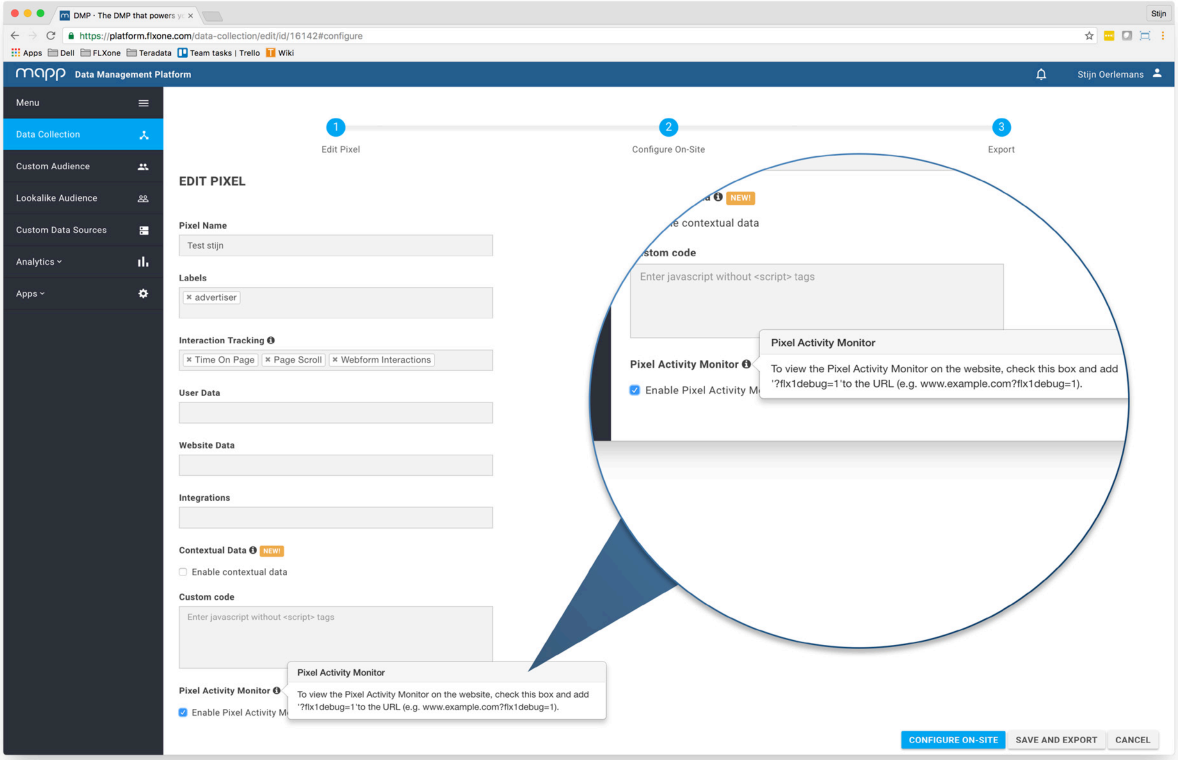
What is the difference between a Custom Event and a Custom Success Event?
Custom Events (segments) and Custom Success Events (conversions) are industry-standard labels for targetable entities, which is just terminology. The only difference from a technical perspective is that you can apply deduplication settings on a Custom Success Event level within the Mapp DMP. For the rest they both represent a conglomeration of unique User IDs (cookies) that will be added to the entity when the on-site user interactions meets the condition(s) that have been configured for this Custom Event or Custom Success Event. Typically a Custom Success Event serves as a recognizable label of valuable user interactions in an e-commerce environment such as a purchase event, a subscribe event, or a download event.

PIXEL ACTIVITY MONITOR

The Pixel Activity Monitor can be enabled as an On-Site pixel module and serves as a tool to verify the DMP pixel is active on the website. It will pop-up as a floating and draggable element in your browser and shows all data events that are initiated by the implemented pixel script. This also includes interaction events like ‘Time on Page’, ‘Page Scroll’, and ‘Form Interactions’ that are originating from the default pixel modules. Please note that this module is not available by default.

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type



EDIT PIXEL

Pixel Name: Test stijn

Labels: advertiser

Interaction Tracking: Time On Page, Page Scroll, Webform Interactions

User Data

Website Data

Integrations

Contextual Data NEW!

□ Enable contextual data

Custom code: Enter javascript without <script> tags

Pixel Activity Monitor

☑ Enable Pixel Activity M

Pixel Activity Monitor

To view the Pixel Activity Monitor on the website, check this box and add '?flx1debug=1'to the URL (e.g. www.example.com?flx1debug=1).

A step-by-step guide on how to enable the Pixel Activity Monitor:

1. Reach out to your *DMP Account Manager* to activate this module..
2. Go to *Data Collection* and create a new Pixel or choose an existing one.
3. Tick the box to enable the Pixel Activity Monitor for this Pixel and click on *Save and Export*.
4. Browse to the corresponding website and add http://www.sitedomain.com*?flx1debug=1* to the URL.

Demo
Note that the tool can also be used for demo purposes. An extension like Greasemonkey (Firefox) can be used to automatically run the Pixel script on a given website.

Common Questions

As a developer responsible for the pixel implementation, are there any other methods to verify the pixel implementation?
Yes, Firebug or comparable tools can be used to verify the generic DMP pixel loads and the successful measurement of the enabled interaction tracking modules. It can also serve as quick check to make sure the successive segment and conversion events are implemented properly. First navigate to the website where the On-Site Pixel is active.

Tool	Method
Firebug	Validate that the pixel is loaded by looking into the Net panel and search for 'customer_id-pixel_id.js' being called from the 'c .flx1.com' domain. Tip: use the relevant pixel_id as a filter value.
Chrome	Validate that the pixel is loaded by looking into the Network panel of the Developer Console and search for 'customer_id-pixel_id.js' being called from the 'c.flx1.com' domain. Tip: use the relevant pixel_id as a filter value.

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

DATA ATTRIBUTES

An overview of what attributes (data) is being collected by default.

Key	Value	Info	API Identifier
Timestamp	2016-06-09 04:47:13.000		flx_timestamp
Date	06/09/2016		flx_date
Hour	12		flx_hour
User ID	3899573f-ff0e-4a80-4791-b96e060aa7ef	Mapp Cookie ID	flx_uuid
Pixel ID	11223		flx_pixel_id
Event Type	2		flx_event_type
Interaction Type*	4		flx_interaction_type
Interaction Value*	20		flx_interaction_value
Custom (Success) Event ID*	1450		flx_segment_dmp, flx_conversion_dmp
User IP	80.113.22.2		flx_user_ip
User IP Truncated	80.113.22.0		flx_user_ip_truncated
Geo Country	NLD		flx_geo_country
Geo State	NB		flx_geo_region
Geo City	Eindhoven		flx_geo_city
Geo Latitude	52.500000		flx_geo_lat
Geo Longitude	5.750000		flx_geo_long
Browser	1		flx_browser
Browser Version	51.0.2704.84		flx_browser_version
Browser Language	en		flx_browser_language
Browser Language Country	uk		flx_browser_language_country
Operating System	3		flx_operating_system
Operating System Version	3.11		flx_operating_system_version
Device Type	3		flx_device_type
Device Brand	apple		flx_device_brand
Referer URL	http://www.site-domain.com/electronics		flx_referer_url
Event Referer URL	http://www.site-domain.com/home		flx_event_referer_url
Event URL	http://www.site-domain.com/product-details-123		flx_event_url
Site Domain	site-domain.com		flx_site_domain
External Pixel ID**	56789		flx_external_id
Customer User ID	77fac0a6765bcw-24d817a6e24803419de64b00c2a		flx_customer_id
Server Side Fingerprint	85be813dbc9bd0f81e76e-a6eee37389074f76036		flx_server_side_fingerprint
External Data	{"Key":"value}	See External Data	flx_external_data

ON-SITE PIXEL

This entry provides detailed information on the On-Site Pixel type

The following attributes will only be collected when the relevant On-Site module is enabled.

Module	Key	Value
Browser Data	Screen Resolution Height	1080
	Screen Resolution Width	1920
	Browser Resolution Height	421
	Browser Resolution Width	1920
	Page Height	421
	Browser Date	2016-06-09 13:41:15
	Browser Timezone Offset	-120
User Details	User Agent	Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.84 Safari/537.36

External Data

Module	Key	JSON Example
Webform Input	*name**	{“billing_company”:”Mapp”}
Hashed Email	email_md5	{“email_md5”:”9a93efa79aa9f5d35e14bc55a3e16dc4”}
	email_sha1	{“email_sha1”:”ccb9adde72603318101eee57f919ed4debc4a924”}
	email_sha256	{“email_sha256”:”cc779c04191c2e736d89e45c11339c8382832bcaf70383f7df94e3d08ba7a6d9”}
Common Words	common_words	{“common_words”:[“sport”,”football”,”club”,”league”,”soccer”]}
Average Price on Page	avg_price	{“avg_price”:”33.17”}
Clicked Elements	click	{“click”:”<LI className=uk-button sport active> Foot”}
URL Parameters	*name***	{“utm_source”:”adwords”,”utm_medium”:”search”,”utm_campaign”:”brand_awareness”}
* It automatically assigns the name of the input field as key for this (new) DMP attribute. ** It automatically assigns the key of the URL parameter as key for this (new) DMP attribute.		

Common Questions

What is the difference between the referer_url and the event_referer_url?

The referer_url is present in the ‘Request Header’ section of the browser console and represents the referer of the pixel itself. In case a user enters a page on which the pixel is active and being loaded for the first time this referer_url is always similar to the actual event_url. When you then visit the ‘next page’ it will be the url of the first page visit where the pixel was loaded. Consider this as the internal pixel referer_url. The event_referer_url is present in the ‘Query String Parameters’ section of the browser console. This is the referer_url as represent in the ‘Query String Parameters’ section. As an example you could imagine that a user has entered the page on which the pixel is active as a result of clicking on a google search entry and can therefore be seen as the cross-domain referer_url.