```
{# themes/hellotheme/html/message.html.twig #}
{% extends ":"~template~":base.html.twig" %}

{% block content %}
<!DOCTYPE html>
<html>
    <head>
        {% if page is defined %}
        <title>{pagetitle}</title>
        <meta name="description" content="{pagemetadescription}">
        {% endif %}
        {{ outputHeadDeclarations() }}
    </head>
    <body>
        {{ outputScripts('bodyOpen') }}
        {% block content %}{% endblock %}
        {{ outputScripts('bodyClose') }}
    </body>
</html>
{% endblock %}
```

page.html.twig is exactly the same as email.html.twig except that it'll be used for landing pages instead. Thus, it can be more robust with the HTML document.

REST API

Mautic provides a REST API to manipulate leads and/or obtain information for various entities of Mautic.

☐ All Mautic API endpoints require an OAuth1a signtature or OAuth2 access token.

Error Handling

If an OAuth error is encountered, it'll be a JSON encoded array similar to:

```
{
  "error": "invalid_grant",
  "error_description": "The access token provided has expired."
}
```

If a system error encountered, it'll be a JSON encoded array similar to:

```
{
    "error": {
        "message": "You do not have access to the requested area/action.",
        "code": 403
    }
}
```

Mautic version check

In case your API service wants to support several Mautic versions with different features, you might need to check the version of Mautic you communicate with. Since Mautic 2.4.0 the version number is added to all API response headers. The header name is `Mautic-Version`. With Mautic PHP API library you can get the Mautic version like this:

```
// Make any API request:
$api = $this->getContext('contacts');
$response = $api->getList('', 0, 1);

// Get the version number from the response header:
$version = $api->getMauticVersion();
```

`$version` will be in a semantic versioning format: `[major].[minor].[patch]`. For example: `2.4.0`. If you'll try it on the latest GitHub version, the version will have `-dev` at the end. Like `2.5.1-dev`.

Authorization

Mautic uses OAuth or Basic Authentication (as of Mautic 2.3.0) for API authorization. It supports both OAuth 1a and OAuth 2; however, as of 1.1.2, the administrator of the Mautic instance must choose one or the other. Of course OAuth 2 is only recommended for servers secured behind SSL. Basic authentication must be enabled in Configuration -> API Settings.

The Mautic administrator should enable the API in the Configuration -> API Settings. This will add the "API Credentials" to the admin menu. A client/consumer ID and secret should then be generated which will be used in the following processes.

All authorization requests should be made to the specific Mautic instances URL, i.e. `https://your-mautic.com`.

> ☐ The OAuth processes can be a pain. If possible, it's best to use an OAuth library for the language being used. If PHP, it is recommended that Mautic's PHP library be used.

OAUTH 1A

```
<?php
use Mautic\Auth\ApiAuth;

// $initAuth->newAuth() will accept an array of OAuth settings
$settings = array(
    'baseUrl'      => 'https://your-mautic.com',
    'version'      => 'OAuth1a',
    'clientKey'    => '5ad6fa7asfs8fa7sdfa6sfas5fas6asdf8',
    'clientSecret' => 'adf8asf7sf54asf3as4f5sf6asfasf97dd',
    'callback'     => 'https://your-callback.com'
);

// Initiate the auth object
$initAuth = new ApiAuth();
$auth     = $initAuth->newAuth($settings);

// Initiate process for obtaining an access token; this will redirect the user to the authorize endpoint and/or set the tokens when the user is
redirected back after granting authorization

if ($auth->validateAccessToken()) {
    if ($auth->accessTokenUpdated()) {
        $accessTokenData = $auth->getAccessTokenData();

        //store access token data however you want
    }
}
```

The OAuth 1a method is recommended for servers that are not behind https. Note that OAuth 1a access tokens do not expire.

OAuth 1a can be a complicated method due to the need to generate a signature for the request. If anything is off with the signature, the request will not be validated.

AUTHORIZATION

STEP ONE - OBTAIN A REQUEST TOKEN

The first step is to obtain a request token that will be used when directing the user to the authorization page.

Make a POST to the request token endpoint `/oauth/v1/request_token`:

```
POST /oauth/v1/request_token
Authorization:
        OAuth oauth_callback="https%3A%2F%2Fyour-callback-uri.com",
              oauth_consumer_key="CONSUMER_KEY",
              oauth_nonce="UNIQUE_STRING",
              oauth_signature="GENERATED_REQUEST_SIGNATURE",
              oauth_signature_method="HMAC-SHA1",
              oauth_timestamp="1318467427",
              oauth_version="1.0"
```

(note that the header has been wrapped for legibility)

Review Generating the Authorization Header on the specifics of generating the OAuth header.

The response will be a query string:

```
oauth_token=REQUEST_TOKEN&oauth_token_secret=REQUEST_TOKEN_SECRET&oauth_expires_in=3600
```

Parse the string and use the parameters in the next step as indicated.

☐ These must be preserved in some kind of persistent storage as they will be used when obtaining the access token.

Note that the refresh token is only good for the number of seconds specified in `oauth_expires_in`.

STEP TWO - AUTHORIZATION

Now redirect the user to the authorization endpoint `oauth/v1/authorize` with the request token as part of the URL's query.

If the callback is something different than what is configured in Mautic, url encode it and include in the query as `oauth_callback`.

```
/oauth/v1/authorize?oauth_token=REQUEST_TOKEN&oauth_callback=https%3A%2F%2Fyour-callback-uri.com
```

The user will login and Mautic will redirect back to the either the consumer's configured callback or to the `oauth_callback` included in the query.

The callback will include `oauth_token` and `oauth_verifier` in the URL's query.

Compare the `oauth_token` in the query with that obtained in step two to ensure they are the same and prevent cross-site request forgery.

`oauth_verifier` will need to be part of the header generated in step three.

STEP THREE - OBTAIN AN ACCESS TOKEN

Generate the Authorization header and make a POST to the access token endpoint `/oauth/v1/access_token`.

When generating the header, the `oauth_token_secret` returned in step two should be used as the `TOKEN_SECRET` in the composite key.

`oauth_verifier` from step two should be part of the Authorization header generated.

```
POST /oauth/v1/access_token
Authorization:
        OAuth oauth_callback="https%3A%2F%2Fyour-callback-uri.com",
              oauth_consumer_key="CONSUMER_KEY",
              oauth_nonce="UNIQUE_STRING",
              oauth_signature="GENERATED_REQUEST_SIGNATURE",
              oauth_signature_method="HMAC-SHA1",
              oauth_timestamp="1318467427",
              oauth_verifier="OAUTH_VERIFIER_FROM_STEP_TWO"
              oauth_version="1.0"
```

(note that the header has been wrapped for legibility)

The response should include a query string with the access token:

```
oauth_token=ACCESS_TOKEN&oauth_token_secret=ACCESS_TOKEN_SECRET
```

☐ Mautic's OAuth 1a access tokens do not expire but the user can deauthorize them. If the access token is invalid, a `401` response will be returned.

The `oauth_token` can be included in the authorize header and the `oauth_token_secret` should be used as the `TOKEN_SECRET` in the composite key when signing API requests.

GENERATING THE AUTHORIZATION HEADER

The OAuth header may include the following parameters:

Key	Description
oauth_callback	Optional, URL encoded callback to redirect the user to after authorization. If the callback URL is set in Mautic, this must match.
oauth_consumer_key	The consumer key obtained from Mautic's API Credentials
oauth_nonce	Uniquely generated string that should be used only once
oauth_signature	Signature generated from all applicable data for the request
oauth_signature_method	Method for creating the signature. Currently, only `HMAC-SHA1` is supported.
oauth_timestamp	Current unix timestamp
oauth_token	The access token
oauth_verifier	Returned after authorization and used when requesting an access token
oauth_version	Should always be `1.0`

☐ When making a GET or POST request with parameters, all parameters should be included in the header as well.

STEP ONE - BUILD THE BASE STRING

The base string is used to generate the oauth_signature.

The structure of the base string is as follows:

```
METHOD&URL_ENCODED_URI&NORMALIZED_PARAMETERS
```

`METHOD` should be the request method and should always be capitalized.

`URL_ENCODED_URI` should be the base URI the request is made to. It should not include any query parameters (query parameters should be part of `NORMALIZED_PARAMETERS`).

`NORMALIZED_PARAMETERS` should be a url encoded, alphabetically sorted query string of the above oauth parameters (except `oauth_signature`) plus the parameters of the request (query/post body).

Each key and each value of the parameters must be url encoded individually as well.

☐ PHP should use `rawurlencode()` instead of `urlencode()`.

Then each url encoded key/value pair should be concatenated into a single string with an ampersand (&) as the glue character.

For example, let’s say a request includes a query of `title=Mr&firstname=Joe&lastname=Smith`. The process would look something like the following (replacing `urlencode()` with whatever function is appropriate for the language being used). Of course realistically, natural sort and loop functions would be used.

```
urlencode(
    urlencode(firstname)=urlencode(Joe)
    &urlencode(lastname)=urlencode(smith)
    &urlencode(oauth_callback)=urlencode(https%3A%2F%2Fyour-callback-uri.com)
    &urlencode(oauth_consumer_key)=urlencode(kdjafs7fsdf86ads7a98a87df6ad9fsf98ad7f)
    &urlencode(oauth_nonce)=urlencode(ak877asdf6adf68asd9fas)
    &urlencode(oauth_signature_method)=urlencode(HMAC-SHA1)
    &urlencode(oauth_timestamp)=urlencode(1437604736)
    &urlencode(oauth_version)=urlencode(1.0)
    &urlencode(title)=urlencode(Mr)
)
```

☐ Multidimensional arrays should use `foo=baz&foo=bar` rather than `foo[bar]=baz&foo[baz]=bar` when generating the normalized parameters string.

☐ Notice that `oauth_callback`, if included, is DOUBLE url encoded.

Put all together, a base string might look like:

```
GET&http%3A%2F%2Fyour-mautic.com%2Fapi&firstname%3DJoe%26lastName%3DSmith%26oauth_callback%3Dhttps%253A%252F%252Fyour-
callback-
uri.com%26oauth_consumer_key%3Dkdjafs7fsdf86ads7a98a87df6ad9fsf98ad7f%26oauth_nonce%3Dak877asdf6adf68asd9fas%26oauth_si
gnature_method%3DHMAC-SHA1%26oauth_timestamp%3D1437604736%26oauth_version%3D1.0%26title%3DMr
```

STEP TWO - BUILD THE COMPOSITE KEY

The composite key is used to encrypt the base string. It is composed of the consumer secret and the token secret if available (post authorization) combined with an ampersand (&).

```
CLIENT_SECRET&TOKEN_SECRET
```

If the token secret is not available, i.e. during the request token process, then an ampersand (&) should still be used.

```
CLIENT_SECRET&
```

☐ `TOKEN_SECRET` Sources
For `/oauth/v1/request_token`, `TOKEN_SECRET` is empty.

For `/oauth/v1/access_token`, `TOKEN_SECRET` will be the `oauth_token_secret` returned by the request to `/oauth/v1/request_token`

For all other API endpoints, `TOKEN_SECRET` will be the `oauth_token_secret` returned by the request to `/oauth/v1/access_token`

STEP THREE - GENERATE THE SIGNATURE

Now, using the base string and the composite key, the signature can be generated using the appropriate HMAC function available to the language used. The signature generated should then be base64 encoded prior to being used in the Authorization header.

```
base64_encode(
    hmac_sha1(BASE_STRING, COMPOSITE_KEY)
)
```

The resulting string should then be used included in the header as `oauth_signature`.

SIGNING THE API REQUEST

To sign the API request, generate the Authorization header using the obtained access token.

```
POST /api/leads/new
Authorization:
        OAuth oauth_callback="https%3A%2F%2Fyour-callback-uri.com",
              oauth_consumer_key="CONSUMER_KEY",
              oauth_nonce="UNIQUE_STRING",
              oauth_signature="GENERATED_REQUEST_SIGNATURE",
              oauth_signature_method="HMAC-SHA1",
              oauth_timestamp="1318467427",
              oauth_token="ACCESS_TOKEN"
              oauth_version="1.0"

title=Mr&firstname=Joe&lastname=Smith
```

(note that the header has been wrapped for legibility)

OAUTH 2

```
<?php
use Mautic\Auth\ApiAuth;

// $initAuth->newAuth() will accept an array of OAuth settings
$settings = array(
    'baseUrl'      => 'https://your-mautic.com',
    'version'      => 'OAuth2',
    'clientKey'    => '5ad6fa7asfs8fa7sdfa6sfas5fas6asdf8',
    'clientSecret' => 'adf8asf7sf54asf3as4f5sf6asfasf97dd',
    'callback'     => 'https://your-callback.com'
);

// Initiate the auth object
$initAuth = new ApiAuth();
$auth     = $initAuth->newAuth($settings);
```

```
// Initiate process for obtaining an access token; this will redirect the user to the authorize endpoint and/or set the tokens when the user is
redirected back after granting authorization

if ($auth->validateAccessToken()) {
    if ($auth->accessTokenUpdated()) {
        $accessTokenData = $auth->getAccessTokenData();

        //store access token data however you want
    }
}
```

☐ Mautic supports only the authorization_code, refresh_token and client_credentials grant types.

AUTHORIZATION

STEP ONE - OBTAIN AUTHORIZATION CODE

Redirect the user to the authorize endpoint `oauth/v2/authorize`:

```
GET /oauth/v2/authorize?
    client_id=CLIENT_ID
    &grant_type=authorization_code
    &redirect_uri=https%3A%2F%2Fyour-redirect-uri.com%2Fcallback
    &response_type=code
    &state=UNIQUE_STATE_STRING
```

(note that the query has been wrapped for legibility)

☐ The state is optional but recommended to prevent CSRF attacks. It should be a uniquely generated string and stored locally in session, cookie, etc. to be compared with the returned value.

☐ Note that the redirect_uri should be URL encoded.

☐ As of 1.1.2, Mautic does not leverage OAuth2 scopes.

The user will be prompted to login. Once they do, Mautic will redirect back to the URL specified in redirect_uri with a code appended to the query.

It may look something like: `https://your-redirect-uri.com?code=UNIQUE_CODE_STRING&state=UNIQUE_STATE_STRING`

The state returned should be compared against the original to ensure nothing has been tampered with.

STEP TWO - REPLACE WITH AN ACCESS TOKEN

Obtain the value of the code from Step One then immediately POST it back to the access token endpoint `oauth/v2/token` with:

```
POST /oauth/v2/token

client_id=CLIENT_ID
    &client_secret=CLIENT_SECRET
    &grant_type=authorization_code
    &redirect_uri=https%3A%2F%2Fyour-redirect-uri.com%2Fcallback
    &code=UNIQUE_CODE_STRING
```

(note that the post body has been wrapped for legibility)

The response returned should be a JSON encoded string:

```
    {
        access_token: "ACCESS_TOKEN",
        expires_in: 3600,
        token_type: "bearer",
        scope: "",
        refresh_token: "REFRESH_TOKEN"
    }
```

This data should be stored in a secure location and used to authenticate API requests.

REFRESH TOKENS

The response's `expires_in` is the number of seconds the access token is good for and may differ based on what is configured in Mautic. The code handling the authorization process should generate an expiration timestamp based on that value. For example `<?php $expiration = time() + $response['expires_in']; ?>`. If the access token has expired, the refresh_token should be used to obtain a new access token.

By default, the refresh token is valid for 14 days. - If your application requests a new access token using the refresh token within 14 days, no user interaction is needed. Your application gets both a new access token and a new refresh token (which is valid for another 14 days after it's issued); - If your application does not request a new token using the refresh token within 14 days, user interaction is required in order to get new tokens.

The refresh token's expiration time is configurable through Mautic's Configuration.

☐ The application should monitor for a `400 Bad Response` response when requesting a new access token and redirect the user back through the authorization process.

To obtain a new access token, a POST should be made to the access token's endpoint `oauth/v2/token` using the `refresh_token` grant type.

```
POST /oauth/v2/token

client_id=CLIENT_ID
    &client_secret=CLIENT_SECRET
    &grant_type=refresh_token
    &refresh_token=REFRESH_TOKEN
    &redirect_uri=https%3A%2F%2Fyour-redirect-uri.com%2Fcallback
```

(note that the post body has been wrapped for legibility)

The response returned should be a JSON encoded string:

```
    {
        access_token: "NEW_ACCESS_TOKEN",
        expires_in: 3600,
        token_type: "bearer",
        scope: "",
        refresh_token: "REFRESH_TOKEN"
    }
```

CLIENT CREDENTIALS

The Client Credentials grant is used when applications request an access token to access their own resources, not on behalf of a user.

To obtain a new access token, a POST should be made to the access token's endpoint `oauth/v2/token` using the `client_credentials` grant type.

```
POST /oauth/v2/token

client_id=CLIENT_ID
    &client_secret=CLIENT_SECRET
    &grant_type=client_credentials
```

(note that the post body has been wrapped for legibility)

The response returned should be a JSON encoded string:

```
    {
        access_token: "NEW_ACCESS_TOKEN",
        expires_in: 3600,
        token_type: "bearer",
        scope: ""
    }
```

AUTHENTICATING THE API REQUEST

Authenticating the API request with OAuth2 is easy. Choose one of the following methods that is appropriate for the application's needs.

AUTHORIZATION HEADER

By using an authorization header, any request method can be authenticated.

However, note that this method requires that the server Mautic is installed on passes headers to PHP or has access to the `apache_request_headers()` function. `apache_request_headers()` is not available to PHP running under fcgi.

```
Authorization: Bearer ACCESS_TOKEN
```

METHOD SPECIFIC

The access token can also be appended to the query or included in the body of a POST.

```
GET https://your-mautic.com/api/leads?access_token=ACCESS_TOKEN
```

```
POST https://your-mautic.com/api/leads/new

firstname=John&lastname=Smith&access_token=ACCESS_TOKEN
```

BASIC AUTHENTICATION

As of Mautic 2.3.0, support for basic authentication can be enabled through Mautic's Configuration -> API Settings. As with OAuth2, it is only recommended to use basic authentication over HTTPS.

To authorize a request for basic authentication, set an `Authorization` header.

1. Combine the username and password of a Mautic user with a colon `:`. For example, `user:password`.
2. Base64 encode the string from above. `dXNlcjpwYXNzd29yZA==`.
3. Add an Authorization header to each API request as `Authorization: Basic dXNlcjpwYXNzd29yZA==`

API Rate limiter

You can configure rate limiter cache in local.php By default, filesystem is used as: `php api_rate_limiter_cache => [ 'type' => 'file_system', ],` You can configure memcached server: `php 'api_rate_limiter_cache' => [ 'memcached' => [ 'servers' => [ [ 'host' => 'localhost', 'port' => 11211 ] ] ] ],`

Or whatever cache you want described in Symfony cache documentation.

Libraries

PHP LIBRARY

Mautic provides a PHP library on Github. It is recommended that it be used in PHP projects. Other languages will need to use custom means and/or a 3rd party library to handle the OAuth/request processes.

INSTALL VIA COMPOSER

To install using composer, simply run `composer require mautic/api-library`.

INSTALL MANUALLY

Download the package from Github. Extract then include the following code in your project (of course change the file path if needed):

```
require_once __DIR__ . '/lib/Mautic/MauticApi.php';
```

☐ Refer to the README in the Github repository for further instructions on how to use the library or review the code examples included throughout this section.

Endpoints

☐ All responses are JSON encoded.

☐ Base API Endpoint:
`https://your-mautic.com/api`

Assets

Use this endpoint to obtain details on Mautic’s assets.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth = new ApiAuth();
$auth     = $initAuth->newAuth($settings);
$apiUrl   = "https://your-mautic.com";
$api      = new MauticApi();
```

```
$assetApi = $api->newApi("assets", $auth, $apiUrl);
```

GET ASSET

```php
<?php

//...
$asset = $assetApi->get($id);
```

Get an individual asset by ID.

HTTP REQUEST

`GET /assets/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Asset Properties

Name	Type	Description
id	int	ID of the asset
title	string	Title/name of the asset
description	string/null	Description of the asset
alias	string	Used to generate the URL for the asset
language	string	Locale of the asset
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the asset should be published
publishDown	datetime/null	Date/time the asset should be un published
dateAdded	datetime	Date/time asset was created
createdBy	int	ID of the user that created the asset
createdByUser	string	Name of the user that created the asset
dateModified	datetime/null	Date/time asset was last modified
modifiedBy	int	ID of the user that last modified the asset
modifiedByUser	string	Name of the user that last modified the asset
downloadCount	int	Total number of downloads
uniqueDownloadCount	int	Unique number of downloads
revision	int	Revision version
category	object/null	Object with the category details

Name	Type	Description
extension	string	Extension of the asset
mime	string	Mime type of the asset
size	int	Filesize of the asset in bytes
downloadUrl	string	Public download URL for the asset

LIST ASSETS

```php
<?php
// ...

$assets = $assetApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

HTTP REQUEST

`GET /assets`

Query Parameters

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.
publishedOnly	Only return currently published entities.
minimal	Return only array of entities without additional lists in it.

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Properties

Same as Get Asset.

CREATE ASSET

```php
<?php

/**
 * Local asset example
 */
// Upload a local file first
$apiContextFiles = $this->getContext('files');
$apiContextFiles->setFolder('assets');
$fileRequest = array(
    'file' => dirname(__DIR__).'/'.'mauticlogo.png'
);
$response = $apiContextFiles->create($fileRequest);
```

```
$data = array(
    'title' => 'Mautic Logo sent as a API request',
    'storageLocation' => 'local',
    'file' => $response['file']['name']
);

$asset = $assetApi->create($data);


/**
 * Remote asset example
 */
$data = array(
    'title' => 'PDF sent as a API request',
    'storageLocation' => 'remote',
    'file' => 'https://www.mautic.org/media/logos/logo/Mautic_Logo_DB.pdf'
);

$asset = $assetApi->create($data);
```

Create a new asset. There are 2 options: local or remote asset.

HTTP REQUEST

`POST /assets/new`

Post Parameters

Name	Description
title	string
storageLocation	string
file	string

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Asset.

EDIT ASSET

```
<?php

$id   = 1;
$data = array(
    'type' => 'general',
);

// Create new a asset of ID 1 is not found?
$createIfNotFound = true;

$asset = $assetApi->edit($id, $data, $createIfNotFound);
```

Edit a new asset. Asset that this supports PUT or PATCH depending on the desired behavior.

PUT creates a asset if the given ID does not exist and clears all the asset information, adds the information from the request. **PATCH** fails if the asset with the given ID does not exist and updates the asset field values with the values form the request.

HTTP REQUEST

To edit a asset and return a 404 if the asset is not found:

`PATCH /assets/ID/edit`

To edit a asset and create a new one if the asset is not found:

`PUT /assets/ID/edit`

Post Parameters

Name	Description
title	string
storageLocation	string
file	string

RESPONSE

If `PUT`, the expected response code is `200` if the asset was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Asset.

DELETE ASSET

```
<?php

$asset = $assetApi->delete($id);
```

Delete a asset. In case of local storage location, the local file will be deleted as well.

HTTP REQUEST

`DELETE /assets/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Asset.

Campaigns

Use this endpoint to obtain details on Mautic’s campaigns.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth    = new ApiAuth();
$auth        = $initAuth->newAuth($settings);
$apiUrl      = "https://your-mautic.com";
```

```
$api         = new MauticApi();
$campaignApi = $api->newApi("campaigns", $auth, $apiUrl);
```

GET CAMPAIGN

```
<?php

//...
$campaign = $campaignApi->get($id);
```

Get an individual campaign by ID.

HTTP REQUEST

`GET /campaigns/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Campaign Properties

Name	Type	Description
id	int	ID of the campaign
name	string	Name of the campaign
description	string/null	Description of the campaign
alias	string	Used to generate the URL for the campaign
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the campaign should be published
publishDown	datetime/null	Date/time the campaign should be un published
dateAdded	datetime	Date/time campaign was created
createdBy	int	ID of the user that created the campaign
createdByUser	string	Name of the user that created the campaign
dateModified	datetime/null	Date/time campaign was last modified
modifiedBy	int	ID of the user that last modified the campaign
modifiedByUser	string	Name of the user that last modified the campaign
events	array	Array of Event entities for the campaign. See below.

Event Properties

Name	Type	Description
id	int	ID of the event

Name	Type	Description
name	string	Name of the event
description	string	Optional description for the event
type	string	Type of event
eventType	string	“action” or “decision”
order	int	Order in relation to the other events (used for levels)
properties	object	Configured properties for the event
triggerMode	string	“immediate”, “interval” or “date”
triggerDate	datetime/null	Date/time of when the event should trigger if triggerMode is “date”
triggerInterval	int/null	Interval for when the event should trigger
triggerIntervalUnit	string	Interval unit for when the event should trigger. Options are i = minutes, h = hours, d = days, m = months, y = years
children	array	Array of this event’s children ,
parent	object/null	This event’s parent
decisionPath	string/null	If the event is connected into an action, this will be “no” for the non-decision path or “yes” for the actively followed path.

LIST CAMPAIGNS

```php
<?php
// ...

$campaigns = $campaignApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

HTTP REQUEST

`GET /campaigns`

Query Parameters

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.
published	Only return currently published entities.
minimal	Return only array of entities without additional lists in it.

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Properties

Same as Get Campaign.

LIST CAMPAIGN CONTACTS

This endpoint is basically an alias for the stats endpoint with 'campaign_leads' table and campaign_id specified. Other parameters are the same as in the stats endpoint.

```
<?php
// ...

$response = $campaignApi->getContacts($campaignId, $start, $limit, $order, $where);
```

HTTP REQUEST

`GET /campaigns/ID/contacts`

Query Parameters

RESPONSE

`Expected Response Code: 200`

See JSON code example.

CREATE CAMPAIGN

```
<?php

$data = array(
    'name'        => 'Campaign A',
    'description' => 'This is my first campaign created via API.',
    'isPublished' => 1
);

$campaign = $campaignApi->create($data);
```

Create a new campaign. To see more advanced example with campaing events and so on, see the unit tests.

HTTP REQUEST

`POST /campaigns/new`

Post Parameters

Name	Description
name	Campaign name is the only required field
alias	string
description	A description of the campaign.
isPublished	A value of 0 or 1

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Campaign.

CLONE A CAMPAIGN

```
<?php

$camnpaignId = 12;

$campaign = $campaignApi->cloneCampaign($campaignId);
```

Clone an existing campaign. To see more advanced example with campaign events and so on, see the unit tests.

HTTP REQUEST

`POST /campaigns/clone/CAMPAIGN_ID`

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Campaign.

EDIT CAMPAIGN

```
<?php

$id   = 1;
$data = array(
    'name'        => 'New campaign name',
    'isPublished' => 0
);

// Create new a campaign of ID 1 is not found?
$createIfNotFound = true;

$campaign = $campaignApi->edit($id, $data, $createIfNotFound);
```

Edit a new campaign. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a campaign if the given ID does not exist and clears all the campaign information, adds the information from the request. **PATCH** fails if the campaign with the given ID does not exist and updates the campaign field values with the values form the request.

HTTP REQUEST

To edit a campaign and return a 404 if the campaign is not found:

`PATCH /campaigns/ID/edit`

To edit a campaign and create a new one if the campaign is not found:

`PUT /campaigns/ID/edit`

Post Parameters

Name	Description
name	Campaign name is the only required field
alias	Name alias generated automatically if not set

Name	Description
description	A description of the campaign.
isPublished	A value of 0 or 1

RESPONSE

If `PUT`, the expected response code is `200` if the campaign was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Campaign.

DELETE CAMPAIGN

```
<?php

$campaign = $campaignApi->delete($id);
```

Delete a campaign.

HTTP REQUEST

`DELETE /campaigns/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Campaign.

ADD CONTACT TO A CAMPAIGN

```
<?php

//...
$response = $campaignApi->addContact($campaignId, $contactId);
if (!isset($response['success'])) {
    // handle error
}
```

Manually add a contact to a specific campaign.

HTTP REQUEST

`POST /campaigns/CAMPAIGN_ID/contact/CONTACT_ID/add`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

REMOVE CONTACT FROM A CAMPAIGN

```
<?php

//...
$response = $listApi->removeContact($campaignId, $contactId);
if (!isset($response['success'])) {
    // handle error
}
```

Manually remove a contact from a specific campaign.

HTTP REQUEST

`POST /campaigns/CAMPAIGN_ID/contact/CONTACT_ID/remove`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Categories

Use this endpoint to obtain details on Mautic's categories or to manipulate category memberships.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth    = new ApiAuth();
$auth        = $initAuth->newAuth($settings);
$apiUrl      = "https://your-mautic.com";
$api         = new MauticApi();
$categoryApi = $api->newApi("categories", $auth, $apiUrl);
```

GET CATEGORY

```
<?php

//...
$category = $categoryApi->get($id);
```

Get an individual category by ID.

HTTP REQUEST

`GET /categories/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Category Properties

Name	Type	Description
id	int	ID of the category
isPublished	boolean	Whether the category is published
dateAdded	datetime	Date/time category was created
createdBy	int	ID of the user that created the category

Name	Type	Description
createdByUser	string	Name of the user that created the category
dateModified	datetime/null	Date/time category was last modified
modifiedBy	int	ID of the user that last modified the category
modifiedByUser	string	Name of the user that last modified the category
title	string	The category title
alias	string	The category alias
description	string	The category description
color	string	The category color
bundle	string	The bundle where the category will be available

LIST CONTACT CATEGORIES

```
<?php

//...
$categories = $categoryApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

Returns a list of contact categories available to the user. This list is not filterable.

HTTP REQUEST

`GET /categories`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Category Properties

Name	Type	Description
id	int	ID of the category
isPublished	boolean	Whether the category is published
dateAdded	datetime	Date/time category was created
createdBy	int	ID of the user that created the category
createdByUser	string	Name of the user that created the category
dateModified	datetime/null	Date/time category was last modified
modifiedBy	int	ID of the user that last modified the category
modifiedByUser	string	Name of the user that last modified the category
title	string	The category title

Name	Type	Description
alias	string	The category alias
description	string	The category description
color	string	The category color
bundle	string	The bundle where the category will be available

CREATE CATEGORY

```php
<?php

$data = array(
    'categoryname' => 'test',
    'categoryemail' => 'test@category.com',
    'categorycity' => 'Raleigh',
);

$category = $categoryApi->create($data);
```

Create a new category.

HTTP REQUEST

`POST /categories/new`

Post Parameters

Name	Description
title	string
bundle	string

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Category.

EDIT CATEGORY

```php
<?php

$id   = 1;
$data = array(
    'title' => 'test',
    'bundle' => 'asset'
);

// Create new a category of ID 1 is not found?
$createIfNotFound = true;

$category = $categoryApi->edit($id, $data, $createIfNotFound);
```

Edit a new category. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a category if the given ID does not exist and clears all the category information, adds the information from the request. **PATCH** fails if the category with the given ID does not exist and updates the category field values with the values form the request.

HTTP REQUEST

To edit a category and return a 404 if the category is not found:

`PATCH /categories/ID/edit`

To edit a category and create a new one if the category is not found:

`PUT /categories/ID/edit`

Post Parameters

Name	Description
title	string
bundle	string

RESPONSE

If `PUT`, the expected response code is `200` if the category was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Category.

DELETE CATEGORY

```
<?php

$category = $categoryApi->delete($id);
```

Delete a category.

HTTP REQUEST

`DELETE /categories/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Category.

ASSIGN A CATEGORY

To assign a category to an entity simply set `category = [ID]` to the payload. For example this is how a category 123 can be asssigned to a new Asset:

```
$data = array(
    'title' => 'PDF sent as a API request',
    'storageLocation' => 'remote',
    'file' => 'https://www.mautic.org/media/logos/logo/Mautic_Logo_DB.pdf'
    'category' => 123
);

$asset = $assetApi->create($data);
```

The category must exist in the Mautic instance and the entity must support categories,

Companies

Use this endpoint to obtain details on Mautic's companies or to manipulate contact-company memberships.

```php
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth   = new ApiAuth();
$auth       = $initAuth->newAuth($settings);
$apiUrl     = "https://your-mautic.com";
$api        = new MauticApi();
$companyApi = $api->newApi("companies", $auth, $apiUrl);
```

GET COMPANY

```php
<?php

//...
$company = $companyApi->get($id);
```

Get an individual company by ID.

HTTP REQUEST

`GET /companies/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Company Properties

Name	Type	Description
id	int	ID of the company
isPublished	boolean	Whether the company is published
dateAdded	datetime	Date/time company was created
createdBy	int	ID of the user that created the company
createdByUser	string	Name of the user that created the company
dateModified	datetime/null	Date/time company was last modified
modifiedBy	int	ID of the user that last modified the company
modifiedByUser	string	Name of the user that last modified the company
fields	array	Custom fields for the company

LIST CONTACT COMPANIES

```php
<?php
```

```php
//...
$companies = $companyApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

Returns a list of contact companies available to the user. This list is not filterable.

HTTP REQUEST

`GET /companies`

** Query Parameters **

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Company Properties

Name	Type	Description
id	int	ID of the company
isPublished	boolean	Whether the company is published
dateAdded	datetime	Date/time company was created
createdBy	int	ID of the user that created the company
createdByUser	string	Name of the user that created the company
dateModified	datetime/null	Date/time company was last modified
modifiedBy	int	ID of the user that last modified the company
modifiedByUser	string	Name of the user that last modified the company
fields	array	Custom fields for the company

CREATE COMPANY

```php
<?php

$data = array(
    'companyname' => 'test',
    'companyemail' => 'test@company.com',
    'companycity' => 'Raleigh',
    'overwriteWithBlank' => true
);
```

```
$company = $companyApi->create($data);
```

Create a new company.

HTTP REQUEST

`POST /companies/new`

Post Parameters

Name	Description
companyname	Company name is the only required field. Other company fields can be sent with a value
isPublished	A value of 0 or 1
overwriteWithBlank	If true, then empty values are set to fields. Otherwise empty values are skipped

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Company.

EDIT COMPANY

```
<?php

$id   = 1;
$data = array(
    'companyname' => 'test',
    'companyemail' => 'test@company.com',
    'companycity' => 'Raleigh',
);

// Create new a company of ID 1 is not found?
$createIfNotFound = true;

$company = $companyApi->edit($id, $data, $createIfNotFound);
```

Edit a new company. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a company if the given ID does not exist and clears all the company information, adds the information from the request. **PATCH** fails if the company with the given ID does not exist and updates the company field values with the values form the request.

HTTP REQUEST

To edit a company and return a 404 if the company is not found:

`PATCH /companies/ID/edit`

To edit a company and create a new one if the company is not found:

`PUT /companies/ID/edit`

Post Parameters

Name	Description
companyname	Company name is the only required field. Other company fields can be sent with a value

Name	Description
isPublished	A value of 0 or 1
overwriteWithBlank	If true, then empty values are set to fields. Otherwise empty values are skipped

RESPONSE

If `PUT`, the expected response code is `200` if the company was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Company.

DELETE COMPANY

```
<?php

$company = $companyApi->delete($id);
```

Delete a company.

HTTP REQUEST

`DELETE /companies/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Company.

ADD CONTACT TO A COMPANY

```
<?php

//...
$response = $companyApi->addContact($companyId, $contactId);
if (!isset($response['success'])) {
    // handle error
}
```

Manually add a contact to a specific company.

HTTP REQUEST

`POST /companies/COMPANY_ID/contact/CONTACT_ID/add`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

REMOVE CONTACT FROM A COMPANY

```
<?php

//...
$response = $companyApi->removeContact($contactId, $companyId);
if (empty($response['success'])) {
    // handle error
}
```

Manually remove a contact to a specific company.

HTTP REQUEST

`POST /companies/COMPANY_ID/contact/CONTACT_ID/remove`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Contacts

Use this endpoint to manipulate and obtain details on Mautic’s contacts.

```php
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth   = new ApiAuth();
$auth       = $initAuth->newAuth($settings);
$apiUrl     = "https://your-mautic.com";
$api        = new MauticApi();
$contactApi = $api->newApi("contacts", $auth, $apiUrl);
```

GET CONTACT

```php
<?php

//...
$contact = $contactApi->get($id);
```

Get an individual contact by ID.

HTTP REQUEST

`GET /contacts/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

** Contact Properties **

Name	Type	Description
id	int	ID of the contact
isPublished	Boolean	True if the contact is published
dateAdded	datetime	Date/time contact was created
createdBy	int	ID of the user that created the contact

Name	Type	Description
createdByUser	string	Name of the user that created the contact
dateModified	datetime/null	Date/time contact was last modified
modifiedBy	int	ID of the user that last modified the contact
modifiedByUser	string	Name of the user that last modified the contact
owner	object	User object that owns the contact.
points	int	Contact’s current number of points
lastActive	datetime/null	Date/time for when the contact was last recorded as active
dateIdentified	datetime/null	Date/time when the contact identified themselves
color	string	Hex value given to contact from Point Trigger definitions based on the number of points the contact has been awarded
ipAddresses	array	Array of IPs currently associated with this contact
fields	array	Array of all contact fields with data grouped by field group. See JSON code example for format. This array includes an “all” key that includes an single level array of fieldAlias => contactValue pairs.
tags	array	Array of tags associated with this contact. See JSON code example for format.
utmtags	array	Array of UTM Tags associated with this contact. See JSON code example for format.
doNotContact	array	Array of Do Not Contact objects. See JSON code example for format.

LIST CONTACTS

```
<?php

//...
$contacts = $contactApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

Get a list of contacts.

HTTP REQUEST

`GET /contacts`

** Query Parameters **

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response. However, all properties in the response that are written in camelCase need to be changed a bit. Before every capital add an underscore `_` and then

Name	Description
	change the capital letters to non-capital letters. So `dateIdentified` becomes `date_identified`, `modifiedByUser` becomes `modified_by_user` etc.
orderByDir	Sort direction: asc or desc.
publishedOnly	Only return currently published entities.
minimal	Return only array of entities without additional lists in it.
where	An array of advanced where conditions
order	An array of advanced order statements

ADVANCED FILTERING

In some cases you may want to filter by specific value(s). Use URL params like this:

In PHP: `php $where = [ [ 'col' => 'phone', 'expr' => 'in', 'val' => '444444444,888888888', ] ];` This design allows to add multiple conditions in the same request.

If you are not using PHP, here is URL-encoded example of the above where array: `GET https://[your_mauitc_domain]/api/contacts?where%5B0%5D%5Bcol%5D=phone&where%5B0%5D%5Bexpr%5D=in&where%5B0%5D%5Bval%5D=444444444,888888888`

List of available expressions

RESPONSE

`Expected Response Code: 200`

See JSON code example.

** Properties **

Same as Get Contact.

CREATE CONTACT

```
<?php

$data = array(
    'firstname' => 'Jim',
    'lastname'  => 'Contact',
    'email'     => 'jim@his-site.com',
    'ipAddress' => $_SERVER['REMOTE_ADDR'],
    'overwriteWithBlank' => true,
);

$contact = $contactApi->create($data);
```

Create a new contact.

HTTP REQUEST

`POST /contacts/new`

** Post Parameters **

Name	Description
*	Any contact field alias can be posted as a parameter. For example, firstname, lastname, email, etc.

Name	Description
ipAddress	IP address to associate with the contact
lastActive	Date/time in UTC; preferablly in the format of Y-m-d H:m:i but if that format fails, the string will be sent through PHP's strtotime then formatted
owner	ID of a Mautic user to assign this contact to
overwriteWithBlank	If true, then empty values are set to fields. Otherwise empty values are skipped

RESPONSE

`Expected Response Code: 201`

** Properties **

Same as Get Contact.

CREATE BATCH CONTACT

```
<?php

$data = array(
    array(
    'firstname' => 'Jim',
    'lastname'  => 'Contact',
    'email'     => 'jim@his-site.com',
    'ipAddress' => $_SERVER['REMOTE_ADDR']
    ),
    array(
        'firstname' => 'John',
    'lastname'  => 'Doe',
    'email'     => 'john@his-site.com',
    'ipAddress' => $_SERVER['REMOTE_ADDR']
    )
);
$contact = $contactApi->createBatch($data);
```

Create a batch of new contacts.

HTTP REQUEST

`POST /contacts/batch/new`

** Post Parameters **

Name	Description
*	Any contact field alias can be posted as a parameter. For example, firstname, lastname, email, etc.
ipAddress	IP address to associate with the contact
lastActive	Date/time in UTC; preferablly in the format of Y-m-d H:m:i but if that format fails, the string will be sent through PHP's strtotime then formatted
owner	ID of a Mautic user to assign this contact to

RESPONSE

`Expected Response Code: 201`

** Properties **

Array of contacts. Record is the same as Get Contact.

EDIT CONTACT

```php
<?php

$id   = 1;
$data = array(
    'email'     => 'jim-new-address@his-site.com',
    'ipAddress' => $_SERVER['REMOTE_ADDR'],
);

// Create new a contact of ID 1 is not found?
$createIfNotFound = true;

$contact = $contactApi->edit($id, $data, $createIfNotFound);
```

Edit a new contact. Note that this supports PUT or PATCH depending on the desired behavior.

** PUT ** creates a contact if the given ID does not exist and clears all the contact information, adds the information from the request. **PATCH** fails if the contact with the given ID does not exist and updates the contact field values with the values form the request.

HTTP REQUEST

To edit a contact and return a 404 if the contact is not found:

`PATCH /contacts/ID/edit`

To edit a contact and create a new one if the contact is not found:

`PUT /contacts/ID/edit`

** Post Parameters **

Name	Description
*	Any contact field alias can be posted as a parameter. For example, firstname, lastname, email, etc.
ipAddress	IP address to associate with the contact
lastActive	Date/time in UTC; preferably in the format of Y-m-d H:m:i but if that format fails, the string will be sent through PHP's strtotime then formatted
owner	ID of a Mautic user to assign this contact to
overwriteWithBlank	If true, then empty values are set to fields. Otherwise empty values are skipped

RESPONSE

If `PUT`, the expected response code is `200` if the contact was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

** Properties **

Same as Get Contact.

> Note: In order to remove tag from contact add minus `-` before it. For example: `tags: ['one', '-two']` - sending this in request body will add tag `one` and remove tag `two` from contact.

EDIT BATCH CONTACT

```php
<?php
```

```
$data = array(
    array(
        'id'        => 1,
        'firstname' => 'Jim',
        'lastname'  => 'Contact',
        'email'     => 'jim@his-site.com',
        'ipAddress' => $_SERVER['REMOTE_ADDR']
    ),
    array(
        'id'        => 1,
        'firstname' => 'John',
        'lastname'  => 'Doe',
        'email'     => 'john@his-site.com',
        'ipAddress' => $_SERVER['REMOTE_ADDR']
    )
);

$contact = $contactApi->editBatch($data);
```

Edit several contacts in one request. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a contact if the given ID does not exist and clears all the contact information, adds the information from the request. **PATCH** fails if the contact with the given ID does not exist and updates the contact field values with the values form the request.

HTTP REQUEST

To edit a contact and return a 404 if the contact is not found:

`PATCH /contacts/batch/edit`

To edit a contact and create a new one if the contact is not found:

`PUT /contacts/batch/edit`

Post Parameters

Name	Description
*	Any contact field alias can be posted as a parameter. For example, firstname, lastname, email, etc.
ipAddress	IP address to associate with the contact
lastActive	Date/time in UTC; preferably in the format of Y-m-d H:m:i but if that format fails, the string will be sent through PHP's strtotime then formatted
owner	ID of a Mautic user to assign this contact to
overwriteWithBlank	If true, then empty values are set to fields. Otherwise empty values are skipped

RESPONSE

If `PUT`, the expected response code is `200` if the contact was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Contacts array. Record same as Get Contact.

> Note: In order to remove tag from contact add minus `-` before it. For example: `tags: ['one', '-two']` - sending this in request body will add tag `one` and remove tag `two` from contact.

DELETE CONTACT

```
<?php
```

```
$contact = $contactApi->delete($id);
```

Delete a contact.

HTTP REQUEST

`DELETE /contacts/ID/delete`

RESPONSE

`Expected Response Code: 200`

** Properties **

Same as Get Contact.

DELETE BATCH CONTACT

```
<?php
$data = array(1, 2);
$contact = $contactApi->deleteBatch($data);
```

Delete contacts.

HTTP REQUEST

`DELETE /contacts/batch/delete`

If you are not using PHP, here is a URL example:

`DELETE https://[your_mautic_domain]/api/contacts/batch/delete?ids=1,2`

RESPONSE

`Expected Response Code: 200`

Properties

Contacts array. Record same as Get Contact.

ADD DO NOT CONTACT

```
<?php

$contactApi->addDnc($contactId, $channel, $reason, $channelId, $comments);
```

Add a contact to DNC list

HTTP REQUEST

To add Do Not Contact entry to a contact:

`POST /contacts/ID/dnc/CHANNEL/add`

Data Parameters (optional)

Name	Description
channel	Channel of DNC. For example ‘email’, 'sms’… Default is email.
reason	Int value of the reason. Use Contacts constants: Contacts::UNSUBSCRIBED (1), Contacts::BOUNCED (2), Contacts::MANUAL (3). Default is Manual
channelId	ID of the entity which was the reason for unsubscription

Name	Description
comments	A text describing details of DNC entry

RESPONSE

Same as Get Contact.

REMOVE FROM DO NOT CONTACT

```
<?php
$contactApi->removeDnc($contactId, $channel);
```

Remove a contact from DNC list

HTTP REQUEST

To remove Do Not Contact entry from a contact:

`POST /contacts/ID/dnc/CHANNEL/remove`

Data Parameters (optional)

Name	Description
channel	Channel of DNC. For example 'email', 'sms'… Default is email.

RESPONSE

Same as Get Contact.

ADD UTM TAGS

```
<?php

$data = array(
    'utm_campaign' => 'apicampaign',
    'utm_source'   => 'fb',
    'utm_medium'   => 'social',
    'utm_content'  => 'fbad',
    'utm_term'     => 'mautic api',
    'useragent'    => 'Mozilla/5.0 (Windows NT 10.0; WOW64; rv:50.0) Gecko/20100101 Firefox/50.0',
    'url'          => '/product/fbad01/',
    'referer'      => 'https://google.com/q=mautic+api',
    'query'        => ['cid'=>'abc','cond'=>'new'], // or as string with "cid=abc&cond=new"
    'remotehost'   => 'example.com',
    'lastActive'   => '2017-01-17T00:30:08+00:00'
 );
$contactApi->addUtm($contactId, $data);
```

Add UTM tags to a contact

HTTP REQUEST

To add UTM tag entry to a contact:

`POST /contacts/ID/utm/add`

UTM Parameters (required)

While the parameter array is required, each utm tag entry is optional.

Name	Description
utm_campaign	The UTM Campaign parameter
utm_source	The UTM Source parameter
utm_medium	The UTM Medium parameter
utm_content	The UTM Content parameter
utm_term	The UTM Term parameter
useragent	The browsers UserAgent. If provided a new Device entry will be created if necessary.
url	The page url
referer	The URL of the referer,
query	Any extra query parameters you wish to include. Array or http query string
remotehost	The Host name
lastActive	The date that the action occured. Contacts lastActive date will be updated if included. Date format required `2017-01-17T00:30:08+00:00`.

RESPONSE

Same as Get Contact with the added UTM Tags.

REMOVE UTM TAGS FROM A CONTACT

```
<?php
$contactApi->removeUtm($contactId, $utmId);
```

Remove a set of UTM Tags from a contact

HTTP REQUEST

To remove UTM Tags from a contact:

`POST /contacts/ID/utm/UTMID/remove`

Data Parameters

None required.

RESPONSE

Same as Get Contact without the removed UTM Tags.

ADD POINTS

```
<?php

$data = array(
     'eventName' => 'Score via api',
     'actionName' => 'Adding',
 );
$contactApi->addPoints($contactId, $pointDelta, $data);
```

Add contact points

HTTP REQUEST

To add points to a contact and return a 404 if the lead is not found:

`POST /contacts/ID/points/plus/POINTS`

Data Parameters (optional)

Name	Description
eventName	Name of the event
actionName	Name of the action

RESPONSE

`Expected Response Code: 200` `json { "success": true }`

SUBTRACT POINTS

```php
<?php

$data = array(
     'eventname' => 'Score via api',
     'actionname' => 'Subtracting',
 );
$contactApi->subtractPoints($contactId, $pointDelta, $data);
```

Subtract contact points

HTTP REQUEST

To subtract points from a contact and return a 404 if the contact is not found:

`POST /contacts/ID/points/minus/POINTS`

Data Parameters (optional)

Name	Description
eventname	Name of the event
actionname	Name of the action

RESPONSE

`Expected Response Code: 200` `json { "success": true }`

LIST AVAILABLE OWNERS

```php
<?php

$owners = $contactApi->getOwners();
```

Get a list of owners that can be used to assign contacts to when creating/editing.

HTTP REQUEST

`GET /contacts/list/owners`

RESPONSE

`Expected Response Code: 200`

** Owner Properties **

Name	Type	Description
id	int	ID of the Mautic user
firstName	string	First name of the Mautic user
lastName	string	Last name of the Mautic user

LIST AVAILABLE FIELDS

```
<?php

$fields = $contactApi->getFieldList();
```

Get a list of available contact fields including custom ones.

HTTP REQUEST

`GET /contacts/list/fields`

RESPONSE

`Expected Response Code: 200`

** Field Properties **

Name	Type	Description
id	int	ID of the field
label	string	Field label
alias	string	Field alias used as the column name in the database
type	string	Type of field. I.e. text, lookup, etc
group	string	Group the field belongs to
order	int	Field order

LIST CONTACT NOTES

```
<?php

$notes = $contactApi->getContactNotes($id, $searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

Get a list of notes for a specific contact.

HTTP REQUEST

`GET /contacts/ID/notes`

** Query Parameters **

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.

Name	Description
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.

RESPONSE

`Expected response code: 200`

** Note Properties **

Name	Type	Description
id	int	ID of the note
text	string	Body of the note
type	string	Type of note. Options are “general”, “email”, “call”, “meeting”
dateTime	datetime	Date/time string of when the note was created.

GET SEGMENT MEMBERSHIPS

```php
<?php

$segments = $contactApi->getContactSegments($id);
```

Get a list of contact segments the contact is a member of.

HTTP REQUEST

`GET /contacts/ID/segments`

RESPONSE

`Expected response code: 200`

** List Properties **

Name	Type	Description
id	int	ID of the list
name	string	Name of the list
alias	string	Alias of the list used with search commands.
dateAdded	datetime	Date/time string for when the contact was added to the list
manuallyAdded	bool	True if the contact was manually added to the list versus being added by a filter
manuallyRemoved	bool	True if the contact was manually removed from the list even though the list’s filter is a match

CHANGE LIST MEMBERSHIPS

See Segements.

GET CAMPAIGN MEMBERSHIPS

```php
<?php

$campaigns = $contactApi->getContactCampaigns($id);
```

Get a list of campaigns the contact is a member of.

HTTP REQUEST

`GET /contacts/ID/campaigns`

RESPONSE

`Expected response code: 200`

** List Properties **

Name	Type	Description
id	int	ID of the campaign
name	string	Name of the campaign
dateAdded	datetime	Date/time string for when the contact was added to the campaign
manuallyAdded	bool	True if the contact was manually added to the campaign versus being added by a contact list
manuallyRemoved	bool	True if the contact was manually removed from the campaign when the contact's list is assigned to the campaign
listMembership	array	Array of contact list IDs this contact belongs to that is also associated with this campaign

CHANGE CAMPAIGN MEMBERSHIPS

See Campaigns.

GET CONTACT'S EVENTS

```php
<?php

$events = $contactApi->getEvents($id, $search, $includeEvents, $excludeEvents, $orderBy, $orderByDir, $page);
```

Warning: Deprecated. Use `getActivityForContact` instead.

** Query Parameters **

Name	Description
id	Contact ID
filters[search]	String or search command to filter events by.
filters[includeEvents][]	Array of event types to include.
filters[excludeEvents][]	Array of event types to exclude.
order	Array of Column and Direction [COLUMN, DIRECTION].
page	What page number to load

Get a list of contact events the contact created.

HTTP REQUEST

`GET /contacts/ID/events`

Warning: Deprecated. Use `GET /contacts/ID/activity` instead.

RESPONSE

`Expected response code: 200`

** List Properties **

Name	Type	Description
events	array	List of events
event	string	ID of the event type
icon	string	Icon class from FontAwesome
eventType	string	Human name of the event
eventPriority	string	Priority of the event
timestamp	timestamp	Date and time when the event was created
featured	bool	Flag whether the event is featured
filters	array	Filters used in the query
order	array	Ordering used in the query
types	array	Array of available event types
total	int	Total number of events in the request
page	int	Current page number
limit	int	Limit of events per page
maxPages	int	How many pages of events are there

GET ACTIVITY EVENTS FOR SPECIFIC CONTACT

```
<?php

$events = $contactApi->getActivityForContact($id, $search, $includeEvents, $excludeEvents, $orderBy, $orderByDir, $page, $dateFrom, $dateTo);
```

** Query Parameters **

Name	Description
id	Contact ID
filters[search]	String or search command to filter events by.
filters[includeEvents][]	Array of event types to include.

Name	Description
filters[excludeEvents][]	Array of event types to exclude.
filters[dateFrom]	Date from filter. Must be type of `\DateTime` for the PHP API libary and in format `Y-m-d H:i:s` for HTTP param
filters[dateTo]	Date to filter. Must be type of `\DateTime` for the PHP API libary and in format `Y-m-d H:i:s` for HTTP param
order	Array of Column and Direction [COLUMN, DIRECTION].
page	What page number to load
limit	Limit of events per page

Get a list of contact events the contact had created.

HTTP REQUEST

`GET /contacts/ID/activity`

RESPONSE

`Expected response code: 200`

** List Properties **

Name	Type	Description
events	array	List of events
event	string	ID of the event type
icon	string	Icon class from FontAwesome
eventType	string	Human name of the event
eventPriority	string	Priority of the event
timestamp	timestamp	Date and time when the event was created
featured	bool	Flag whether the event is featured
filters	array	Filters used in the query
order	array	Ordering used in the query
types	array	Array of available event types
total	int	Total number of events in the request
page	int	Current page number
limit	int	Limit of events per page
maxPages	int	How many pages of events are there

GET ACTIVITY EVENTS FOR ALL CONTACTS

```php
<?php

$events = $contactApi->getActivity($search, $includeEvents, $excludeEvents, $orderBy, $orderByDir, $page, $dateFrom, $dateTo);
```

** Query Parameters **

Name	Description
filters[search]	String or search command to filter events by.
filters[includeEvents][]	Array of event types to include.
filters[excludeEvents][]	Array of event types to exclude.
filters[dateFrom]	Date from filter. Must be type of `\DateTime` for the PHP API libary and in format `Y-m-d H:i:s` for HTTP param
filters[dateTo]	Date to filter. Must be type of `\DateTime` for the PHP API libary and in format `Y-m-d H:i:s` for HTTP param
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.
page	What page number to load

HTTP REQUEST

`GET /contacts/activity`

RESPONSE

`Expected response code: 200`

** List Properties **

Name	Type	Description
events	array	List of events
event	string	ID of the event type
icon	string	Icon class from FontAwesome
eventType	string	Human name of the event
eventPriority	string	Priority of the event
contactId	ID of the contact who created the event	
timestamp	timestamp	Date and time when the event was created
featured	bool	Flag whether the event is featured
filters	array	Filters used in the query
order	array	Ordering used in the query

Name	Type	Description
types	array	Array of available event types
total	int	Total number of events in the request
page	int	Current page number
limit	int	Limit of events per page
maxPages	int	How many pages of events are there

GET CONTACT'S COMPANIES

~~~
<?php

$companies = $contactApi->getContactCompanies($contactId);

```json
{
  "total":1,
  "companies":[
    {
      "company_id":"420",
      "date_associated":"2016-12-27 15:03:43",
      "is_primary":"0",
      "companyname":"test",
      "companyemail":"test@company.com",
      "companycity":"Raleigh",
      "score":"0",
      "date_added":"2016-12-27 15:03:42"
    }
  ]
}
~~~

Get a list of contact's companies the contact belongs to.

HTTP REQUEST

`GET /contacts/ID/companies`

RESPONSE

`Expected response code: 200`

List Properties

Name	Type	Description
company_id	int	Company ID
date_associated	datetime	Date and time when the contact was associated to the company
date_added	datetime	Date and time when the company was created
is_primary	bool	Flag whether the company association is primary (current)
companyname	string	Name of the company
companyemail	string	Email of the company
companycity	string	City of the company

Name	Type	Description
score	int	Score of the company

GET CONTACT'S DEVICES

~~~
<?php

$devices = $contactApi->getContactDevices($contactId);

```json
{
  "total":1,
  "devices":[
    {
      "id":60,
      "lead":[],
      "clientInfo":[],
      "device":"desktop",
      "deviceOsName":"Ubuntu",
      "deviceOsShortName":"UBT",
      "deviceOsPlatform":"x64"
    }
  ]
}
~~~

Get a list of contact's devices the contact has used.

HTTP REQUEST

`GET /contacts/ID/devices`

RESPONSE

`Expected response code: 200`

List Properties

Name	Type	Description
id	int	Device ID
clientInfo	array	Array with various information about the client (browser)
device	string	Device type; desktop, mobile..
deviceOsName	string	Full device OS name
deviceOsShortName	string	Short device OS name
deviceOsPlatform	string	OS platform

Data - Dashboard widget data

Use this endpoint to obtain details on Mautic's dashboard statistical data.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;
```

```
// ...
$initAuth   = new ApiAuth();
$auth       = $initAuth->newAuth($settings);
$apiUrl     = "https://your-mautic.com";
$api        = new MauticApi();
$contactApi = $api->newApi("data", $auth, $apiUrl);
```

GET LIST OF AVAILABLE WIDGET TYPES

```
<?php
$data = $dataApi->getList();
```

HTTP REQUEST

`GET /data`

`Expected Response Code: 200`

GET AN INDIVIDUAL WIDGET DATA BY TYPE.

```
<?php
$data = $dataApi->get($type, $options);
```

HTTP REQUEST

`GET /data/{type}?dateFrom={YYYY-mm-dd}&dateTo={YYYY-mm-dd}&timeUnit={m}`

Returns response which can be directly visualized by the chartJS library.

RESPONSE

`Expected Response Code: 200`

HTTP REQUEST

`GET /data/{type}?dateFrom={YYYY-mm-dd}&dateTo={YYYY-mm-dd}&timeUnit={m}&dataFormat={raw}`

Returns raw format which can be more easily processed.

RESPONSE

`Expected Response Code: 200`

“EMAILS IN TIME” WIDGET

FILTER PARAMETERS

filtercompanyId Filter only emails from contacts assigned to provided company.

filter[campaignId (int) Filter only emails from contacts that were sent as part of provided campaign.

filtersegmentId Filter only emails from contacts assigned to provided segment.

DATASET PARAMETER

dataset (array) - sent - opened - unsubscribed - clicked - bounced - failed Provide more datasets in response based on request.

HTTP REQUEST:

```
GET /api/data/emails.in.time?dateFrom={YYYY-mm-dd}&dateTo={YYYY-mm-dd}&timeUnit={m}&filter[campaignId]=
{int}&filter[companyId]={int}&filter[segmentId]=
{int}&withCounts&dataset[]=sent&dataset[]=opened&dataset[]=unsubscribed&dataset[]=clicked
```

“SENT EMAIL TO CONTACTS” WIDGET

FILTER PARAMETERS

filtercompanyId Filter only emails from contacts assigned to provided company.

filter[campaignId (int) Filter only emails from contacts that were sent as part of provided campaign.

filtersegmentId Filter only emails from contacts assigned to provided segment.

HTTP REQUEST:

`GET /api/data/sent.email.to.contacts?dateFrom={YYYY-mm-dd}&dateTo={YYYY-mm-dd}&timeUnit={m}&filter[campaignId]={int}&filter[companyId]={int}&filter[segmentId]={int}&limit=10&offset=0`

"MOST HIT EMAIL REDIRECTS" WIDGETS

FILTER PARAMETERS

filtercompanyId Filter only emails from contacts assigned to provided company.

filter[campaignId (int) Filter only emails from contacts that were sent as part of provided campaign.

filtersegmentId Filter only emails from contacts assigned to provided segment.

HTTP REQUEST:

`GET /api/data/most.hit.email.redirects?dateFrom={YYYY-mm-dd}&dateTo={YYYY-mm-dd}&timeUnit={m}&filter[campaignId]={int}&filter[companyId]={int}&filter[segmentId]={int}&limit=10&offset=0`

Available data URL query params

Name|Type|Example|Description —-|—-|———– timezone|string|America/New_York|PHP timezone dateFrom|string|2016-28-03|Date from in the YYYY-mm-dd HH:ii:ss format dateTo|string|2016-28-04|Date to in the YYYY-mm-dd HH:ii:ss format timeUnit|string|m|Date/Time unit. Available options: Y, m, W, d, H limit|int|10|Limit of the table widget items filter|array|[lead_id => 23]|filters which should be applied to the SQL query

Dynamic Content

Use this endpoint to obtain details on Mautic's web dynamic content.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth          = new ApiAuth();
$auth              = $initAuth->newAuth($settings);
$apiUrl            = "https://your-mautic.com";
$api               = new MauticApi();
$dynamicContentApi = $api->newApi("dynamicContents", $auth, $apiUrl);
```

GET DYNAMIC CONTENT

```
<?php

//...
$dynamicContent = $dynamicContentApi->get($id);
```

Get an individual dynamicContent by ID.

HTTP REQUEST

`GET /dynamiccontents/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Dynamic Content Properties

Name	Type	Description
id	int	ID of the dynamic content
name	string	Name of the dynamic content
description	string/null	Description of the dynamic content
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the dynamic content should be published
publishDown	datetime/null	Date/time the dynamic content should be un published
dateAdded	datetime	Date/time dynamic content was created
createdBy	int	ID of the user that created the dynamic content
createdByUser	string	Name of the user that created the dynamic content
dateModified	datetime/null	Date/time dynamic content was last modified
modifiedBy	int	ID of the user that last modified the dynamic content
modifiedByUser	string	Name of the user that last modified the dynamic content
variantChildren	array	Array of Dynamic Content entities for variants of this landing dynamic content
variantParent	object	The parent/main dynamic content if this is a variant (A/B test)
sentCount	int	Count of how many times the dynamic content was sent

LIST DYNAMIC CONTENTS

```
<?php
// ...

$dynamicContents = $dynamicContentApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

HTTP REQUEST

`GET /dynamiccontents`

Query Parameters

Name	Description
search	String or search command to filter entities by.
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.

Name	Description
orderByDir	Sort direction: asc or desc.
publishedOnly	Only return currently published entities.
minimal	Return only array of entities without additional lists in it.

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Properties

Same as Get Dynamic Content.

CREATE DYNAMIC CONTENT

```php
<?php

$data = array(
    'name'        => 'Dynamic Content A',
    'isPublished' => 1
);

$dynamicContent = $dynamicContentApi->create($data);
```

Create a new dynamicContent.

HTTP REQUEST

`POST /dynamiccontents/new`

Post Parameters

Name	Type	Description
id	int	ID of the dynamic content
name	string	Name of the dynamic content
description	string/null	Description of the dynamic content
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the dynamic content should be published
publishDown	datetime/null	Date/time the dynamic content should be un published
dateAdded	datetime	Date/time dynamic content was created
createdBy	int	ID of the user that created the dynamic content
createdByUser	string	Name of the user that created the dynamic content
dateModified	datetime/null	Date/time dynamic content was last modified
modifiedBy	int	ID of the user that last modified the dynamic content
modifiedByUser	string	Name of the user that last modified the dynamic content

Name	Type	Description
variantChildren	array	Array of Dynamic Content entities for variants of this landing dynamic content
variantParent	object	The parent/main dynamic content if this is a variant (A/B test)
sentCount	int	Count of how many times the dynamic content was sent

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Dynamic Content.

EDIT DYNAMIC CONTENT

```php
<?php

$id   = 1;
$data = array(
    'name'        => 'New dynamicContent name',
    'isPublished' => 0
);

// Create new a dynamicContent of ID 1 is not found?
$createIfNotFound = true;

$dynamicContent = $dynamicContentApi->edit($id, $data, $createIfNotFound);
```

Edit a new dynamicContent. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a dynamicContent if the given ID does not exist and clears all the dynamic content information, adds the information from the request. **PATCH** fails if the dynamic content with the given ID does not exist and updates the dynamic content field values with the values form the request.

HTTP REQUEST

To edit a dynamicContent and return a 404 if the dynamic content is not found:

`PATCH /dynamiccontents/ID/edit`

To edit a dynamicContent and create a new one if the dynamic content is not found:

`PUT /dynamiccontents/ID/edit`

Post Parameters

Name	Type	Description
id	int	ID of the dynamic content
name	string	Name of the dynamic content
description	string/null	Description of the dynamic content
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the dynamic content should be published
publishDown	datetime/null	Date/time the dynamic content should be un published

Name	Type	Description
dateAdded	datetime	Date/time dynamic content was created
createdBy	int	ID of the user that created the dynamic content
createdByUser	string	Name of the user that created the dynamic content
dateModified	datetime/null	Date/time dynamic content was last modified
modifiedBy	int	ID of the user that last modified the dynamic content
modifiedByUser	string	Name of the user that last modified the dynamic content
variantChildren	array	Array of Dynamic Content entities for variants of this landing dynamic content
variantParent	object	The parent/main dynamic content if this is a variant (A/B test)
sentCount	int	Count of how many times the dynamic content was sent

RESPONSE

If `PUT`, the expected response code is `200` if the dynamic content was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Dynamic Content.

DELETE DYNAMIC CONTENT

```
<?php

$dynamicContent = $dynamicContentApi->delete($id);
```

Delete a dynamicContent.

HTTP REQUEST

`DELETE /dynamiccontents/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Dynamic Content.

Emails

Use this endpoint to obtain details, create, update or delete Mautic’s emails.

```
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth = new ApiAuth();
```

```
$auth     = $initAuth->newAuth($settings);
$apiUrl   = "https://your-mautic.com";
$api      = new MauticApi();
$emailApi = $api->newApi("emails", $auth, $apiUrl);
```

GET EMAIL

```
<?php

//...
$email = $emailApi->get($id);
```

Get an individual email by ID.

HTTP REQUEST

`GET /emails/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Email Properties

Name	Type	Description
id	int	ID of the email
name	string	Internal name of the email
subject	stringl	Subject of the email
fromAddress	string	The from email address if it’s different than the one in the Mautic configuration
fromName	string	The from name if it’s different than the one in the Mautic configuration
replyToAddress	string	The reply to email address if it’s different than the one in the Mautic configuration
bccAddress	string	The BCC email address if it’s different than the one in the Mautic configuration
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the email should be published
publishDown	datetime/null	Date/time the email should be un published
dateAdded	datetime	Date/time email was created
createdBy	int	ID of the user that created the email
createdByUser	string	Name of the user that created the email
dateModified	datetime/null	Date/time email was last modified
modifiedBy	int	ID of the user that last modified the email
modifiedByUser	string	Name of the user that last modified the email
language	string	Language locale of the email

Name	Type	Description
readCount	int	Total email read count
sentCount	int	Total email sent count
revision	int	Email revision
customHtml	string	The HTML content of the email
plainText	string	The plain text content of the email
template	string	The name of the template used as the base for the email
emailType	string	If it is a segment (former list) email or template email. Possible values are 'list' and 'template'
translationChildren	array	Array of Page entities for translations of this landing page
translationParent	object	The parent/main page if this is a translation
variantSentCount	int	Sent count since variantStartDate
variantReadCount	int	Read count since variantStartDate
variantChildren	array	Array of Email entities for variants of this landing email
variantParent	object	The parent/main email if this is a variant (A/B test)
variantSettings	array	The properties of the A/B test
variantStartDate	datetime/null	The date/time the A/B test began
category	object/null	Category information
unsubscribeForm	int	Id of the form displayed in the unsubscribe page
dynamicContent	object	Dynamic content configuration
lists	array	Array of segment IDs which should be added to the segment email
assetAttachments	array	asset IDs Array for email attachment

LIST EMAILS

```
<?php
// ...

$emails = $emailApi->getList($searchFilter, $start, $limit, $orderBy, $orderByDir, $publishedOnly, $minimal);
```

HTTP REQUEST

`GET /emails`

Query Parameters

Name	Description
search	String or search command to filter entities by.

Name	Description
start	Starting row for the entities returned. Defaults to 0.
limit	Limit number of entities to return. Defaults to the system configuration for pagination (30).
orderBy	Column to sort by. Can use any column listed in the response.
orderByDir	Sort direction: asc or desc.
publishedOnly	Only return currently published entities.
minimal	Return only array of entities without additional lists in it.

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Properties

Same as Get Email.

CREATE EMAIL

```php
<?php

$data = array(
    'title'        => 'Email A',
    'description' => 'This is my first email created via API.',
    'isPublished' => 1
);

$email = $emailApi->create($data);
```

Create a new email.

HTTP REQUEST

`POST /emails/new`

Post Parameters

Name	Type	Description
id	int	ID of the email
name	string	Internal name of the email
subject	stringl	Subject of the email
fromAddress	string	The from email address if it’s different than the one in the Mautic configuration
fromName	string	The from name if it’s different than the one in the Mautic configuration
replyToAddress	string	The reply to email address if it’s different than the one in the Mautic configuration
bccAddress	string	The BCC email address if it’s different than the one in the Mautic configuration
isPublished	bool	Published state

Name	Type	Description
publishUp	datetime/null	Date/time when the email should be published
publishDown	datetime/null	Date/time the email should be un published
language	string	Language locale of the email
readCount	int	Total email read count
sentCount	int	Total email sent count
revision	int	Email revision
customHtml	string	The HTML content of the email
plainText	string	The plain text content of the email
template	string	The name of the template used as the base for the email
emailType	string	If it is a segment (former list) email or template email. Possible values are 'list' and 'template'
translationChildren	array	Array of Page entities for translations of this landing page
translationParent	object	The parent/main page if this is a translation
variantSentCount	int	Sent count since variantStartDate
variantReadCount	int	Read count since variantStartDate
variantChildren	array	Array of Email entities for variants of this landing email
variantParent	object	The parent/main email if this is a variant (A/B test)
variantSettings	array	The properties of the A/B test
variantStartDate	datetime/null	The date/time the A/B test began
category	object/null	Category information
unsubscribeForm	int	Id of the form displayed in the unsubscribe page
dynamicContent	object	Dynamic content configuration
lists	array	Array of segment IDs which should be added to the segment email

RESPONSE

`Expected Response Code: 201`

Properties

Same as Get Email.

EDIT EMAIL

```
<?php

$id   = 1;
$data = array(
    'title'        => 'New email title',
    'isPublished' => 0
);

// Create new a email of ID 1 is not found?
$createIfNotFound = true;

$email = $emailApi->edit($id, $data, $createIfNotFound);
```

Edit a new email. Note that this supports PUT or PATCH depending on the desired behavior.

PUT creates a email if the given ID does not exist and clears all the email information, adds the information from the request. **PATCH** fails if the email with the given ID does not exist and updates the email field values with the values form the request.

HTTP REQUEST

To edit a email and return a 404 if the email is not found:

`PATCH /emails/ID/edit`

To edit a email and create a new one if the email is not found:

`PUT /emails/ID/edit`

Post Parameters

Name	Type	Description
id	int	ID of the email
name	string	Internal name of the email
subject	stringl	Subject of the email
fromAddress	string	The from email address if it’s different than the one in the Mautic configuration
fromName	string	The from name if it’s different than the one in the Mautic configuration
replyToAddress	string	The reply to email address if it’s different than the one in the Mautic configuration
bccAddress	string	The BCC email address if it’s different than the one in the Mautic configuration
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the email should be published
publishDown	datetime/null	Date/time the email should be un published
language	string	Language locale of the email
readCount	int	Total email read count
sentCount	int	Total email sent count
revision	int	Email revision
customHtml	string	The HTML content of the email
plainText	string	The plain text content of the email
template	string	The name of the template used as the base for the email

Name	Type	Description
emailType	string	If it is a segment (former list) email or template email. Possible values are 'list' and 'template'
translationChildren	array	Array of Page entities for translations of this landing page
translationParent	object	The parent/main page if this is a translation
variantSentCount	int	Sent count since variantStartDate
variantReadCount	int	Read count since variantStartDate
variantChildren	array	Array of Email entities for variants of this landing email
variantParent	object	The parent/main email if this is a variant (A/B test)
variantSettings	array	The properties of the A/B test
variantStartDate	datetime/null	The date/time the A/B test began
category	object/null	Category information
unsubscribeForm	int	Id of the form displayed in the unsubscribe page
dynamicContent	object	Dynamic content configuration
lists	array	Array of segment IDs which should be added to the segment email

RESPONSE

If `PUT`, the expected response code is `200` if the email was edited or `201` if created.

If `PATCH`, the expected response code is `200`.

Properties

Same as Get Email.

DELETE EMAIL

```
<?php

$email = $emailApi->delete($id);
```

Delete a email.

HTTP REQUEST

`DELETE /emails/ID/delete`

RESPONSE

`Expected Response Code: 200`

Properties

Same as Get Email.

SEND EMAIL TO CONTACT

```
<?php
```

```
$email = $emailApi->sendToContact($emailId, $contactId);
```

Send a predefined email to existing contact.

Assets can be referenced for attaching documents (either ids of existing assets or ids returned by the Create Asset).

HTTP REQUEST

`POST /emails/ID/contact/CONTACT_ID/send`

Post Parameters

Name	Type	Description
tokens	array	Array of tokens in email
assetAttachments	array	Array of asset ids

RESPONSE

`Expected Response Code: 200`

Properties `json { "success": 1 }`

SEND EMAIL TO SEGMENT

```
<?php

$email = $emailApi->send($id);
```

Send a segment email to linked segment(s).

HTTP REQUEST

`POST /emails/ID/send`

RESPONSE

`Expected Response Code: 200`

Properties `json { "success": 1, "sentCount": 1, "failedCount": 0 }`

CREATE A REPLY TO A SEND EMAIL SEND ROW

This endpoint can create a record that a specific email stat row received a reply. It will also mark an email send stat as read.

HTTP REQUEST

`POST /emails/reply/TRACKING_HASH`

Tracking hash is created as unique hash for each email send stat record.

RESPONSE

`Expected Response Code: 200`

Properties `json { "success": 1, }`

Fields

Use this endpoint to work with Mautic’s contact/company fields.

```php
<?php
use Mautic\MauticApi;
use Mautic\Auth\ApiAuth;

// ...
$initAuth = new ApiAuth();
$auth     = $initAuth->newAuth($settings);
$apiUrl   = "https://your-mautic.com";
$api      = new MauticApi();
$assetApi = $api->newApi("assets", $auth, $apiUrl);

// Get contact field context:
$fieldApi = $api->newApi("contactFields", $auth, $apiUrl);

// Or use 'companyFields' for company fields:
$fieldApi = $api->newApi("companyFields", $auth, $apiUrl);
```

GET FIELD

```php
<?php

//...
$field = $fieldApi->get($id);
```

Get an individual field by ID.

HTTP REQUEST

`GET /fields/contact/ID` or `GET /fields/company/ID`

RESPONSE

`Expected Response Code: 200`

See JSON code example.

Field Properties

Name	Type	Description
id	int	ID of the field
isPublished	bool	Published state
publishUp	datetime/null	Date/time when the field should be published
publishDown	datetime/null	Date/time the field should be un published
dateAdded	datetime	Date/time field was created
createdBy	int	ID of the user that created the field
createdByUser	string	Name of the user that created the field
dateModified	datetime/null	Date/time field was last modified
modifiedBy	int	ID of the user that last modified the field
modifiedByUser	string	Name of the user that last modified the field
label	string	Name of the field
alias	string	Unique alias of the field used in the form field name attributes