WatchGuard®

Application Programming Interface (API)

Contents

General Information

Gender Equality

For reasons of readability, the male form was chosen in the text. Nevertheless, the information refers to members of both genders.

Used symbols

The following symbols are used to improve the recognizability of relevant steps within instructional chapters:

SYMBOL	DESCRIPTION	EXPLANATION
☑	Prerequisites	Condition that must be fulfilled before performing the next step
➜	Interim result	Result that is reached after executing a step
✓	Final result	Result that is reached after the described order of steps

Safety instructions and warnings

Safety instructions and warnings are used to inform the user about residual risks and dangers and how to avoid them with the recommended procedure. Following safety instructions and warnings are used in this documentation:

SYMBOL	DESCRIPTION	EXPLANATION
ℹ	Info	Note about general information about the product and/or the services
💬	Important	Further information that is relevant for the next steps

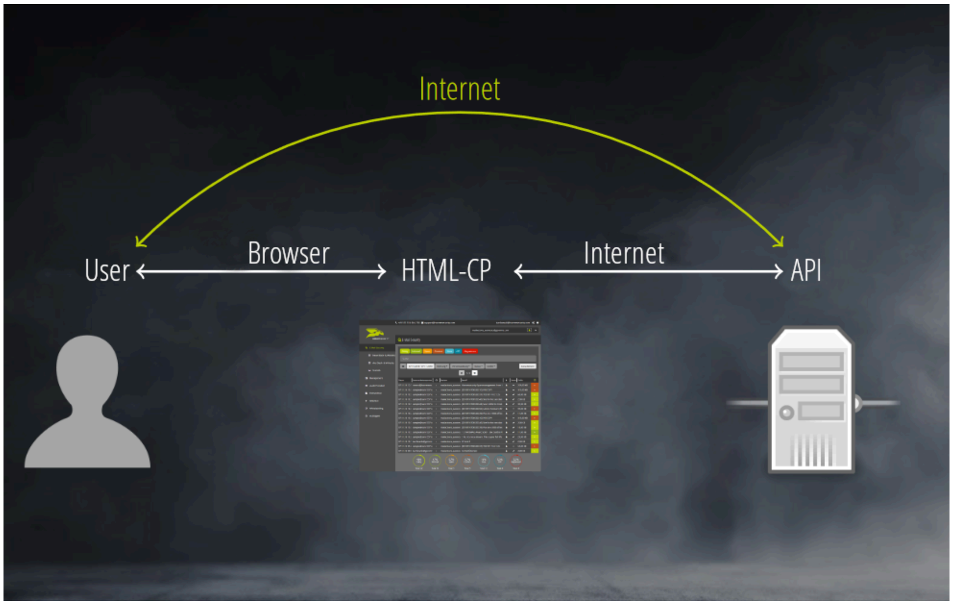
SYMBOL	DESCRIPTION	EXPLANATION
❗	ATTENTION	Warning about additional costs that may incur depending on the booked services
(!)	CAUTION	Warning about the potential loss of data
⚠	DANGER	Warning about the potential infection of the system with malware

General Information

This manual is intended to help you better understand our API and only explains the first steps with the product. For a complete documentation with all possible endpoints that can be addressed inside our API, see https://cp.<your domain>.com/api/docs/.

Introduction

An API is used to exchange information between an application and individual program parts in a standardized way. A previously defined syntax allows a structured transfer of data and commands.



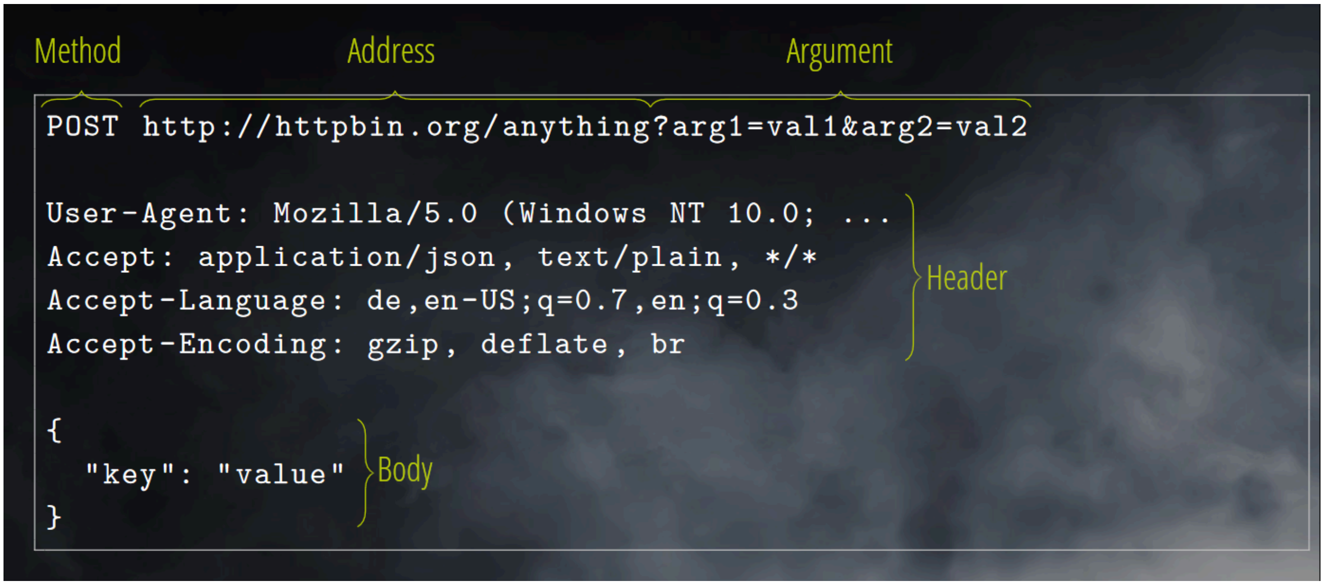
Using our API has the following advantages:

- HTML CP data and functions can be integrated into your own interfaces.
- Repetitive administrative tasks can be automated (e.g., triggering content emails, evaluating spam or editing deny list and allow list entries).

REST (REpresentational State Transfer)

REST is a standard that can be used to communicate with the endpoints of an API through structured requests. REST is based on the concept of a resource whose location can be identified with a unique address (Unique Resource Identifier (URI)). The resources can be requested in different formats (e.g., JSON, XML or EDN). To successfully execute a request, the request must contain certain information:

1. **Method**: The method specifies which operation of the service is to be executed (see HTTP Methods on page 8).
2. **Address**: The address specifies the unique path of the resource to be addressed.
3. **Argument**: The argument specifies how the results of the request are to be filtered.
4. **Header**: The header is used to exchange general metadata.
5. **Body**: The body is used to exchange specific information.



```
POST http://httpbin.org/anything?arg1=val1&arg2=val2

User-Agent: Mozilla/5.0 (Windows NT 10.0; ...
Accept: application/json, text/plain, */*
Accept-Language: de,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br

{
  "key": "value"
}
```

API requests according to the REST standard can, for example, be made with CURL (see CURL on page 9).

HTTP Methods

To address an endpoint via the API, a request must contain a method. The method specifies which operation of the service is to be executed when a resource is addressed. The following methods are used by default:

- **GET** requests the specified resource from the server.
- **PUT** creates a resource. If the resource already exists, it is modified.
- **POST** creates a new resource under the specified resource and issues a unique ID.
- **PATCH** modifies or updates the specified part of a resource.
- **DELETE** deletes the specified resource.

HTTP Status Codes

After a request, the system returns a response in the form of a status code. The server uses the HTTP status code to inform the client whether the request has been successfully processed. If an error occurs, the status code provides information about the error type. Status codes are divided into the following categories:

- Information (100-199).
- Successful operation (200-299).
- Redirection (300-399).
- Client errors (400-499).
- Server errors (500-599).

For an overview of all possible status codes, see https://developer.mozilla.org/en-US/docs/Web/HTTP/Status.

CURL

You can use CURL to execute custom REST API requests.

Example of a request using CURL:

```
curl --request POST \ --url 'http://httpbin.org/anything?=&arg2=val2&arg1=val1'\ --header
 'content-type: application/json'\ --data'{ "key": "value" }'
```

In the following chapters, examples of API requests with CURL are presented (see First Queries on page 16).

For the latest CURL version for your operating system, see https://curl.haxx.se/download.html.

For a complete documentation on CURL, see https://curl.haxx.se/docs/httpscripting.html.

First Steps

In order to establish a connection to our API and to be able to execute specific requests, you need the following data:

INFORMATION	DESCRIPTION	RESOURCE
App ID	The ID of the app you are using	We provide you with the app ID. Please contact the support.
App version	The version of the app you are using	The app version is the version of the app that you use for requests to our API.
CP Token	Your personal access token	You can create your CP token in the Control Panel (see Creating an API Token on page 10).

To access the Control Panel via our API, you need your username and password. The credentials will be sent after a successful login in the authorization header together with an object ID (object_id) and an authentication token. The object ID represents the scope in which the logged-in user performs their role, and the token represents a personal access key for API queries. For more information, see Performing a Login Request on page 13.

Creating an API Token

With an API token, you can grant applications access to the Control Panel API. Each application needs its own token. In your user settings (see User Settings), you can create an API token for authorization in the Control Panel.

1. Log in to the Control Panel with your administrative credentials.

2. Click on ⚙ in the upper right corner of the Control Panel.

   ➡

   The user settings are displayed.

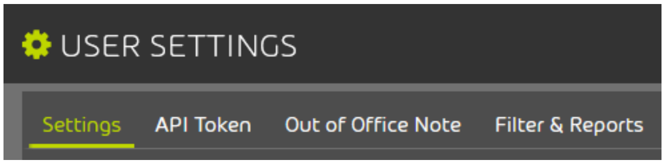
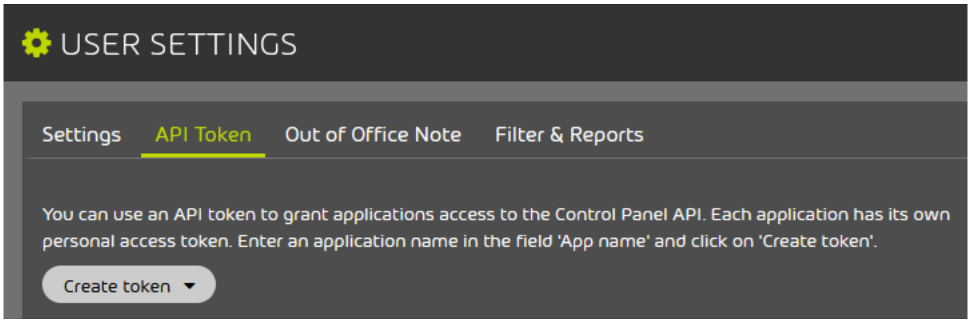


Figure 1: User settings

3. Select the tab **API Token**.
4. Click on **Create token**.



You can use an API token to grant applications access to the Control Panel API. Each application has its own personal access token. Enter an application name in the field 'App name' and click on 'Create token'.

Figure 2: Create Token

➡

Additional settings are displayed.

5. In the **App Name** field, enter a name for the application that will use the token.

> **Notice:**
>
> The actions performed with this token will be recorded in the **Auditing 2.0** module under the name of the user who created the token.

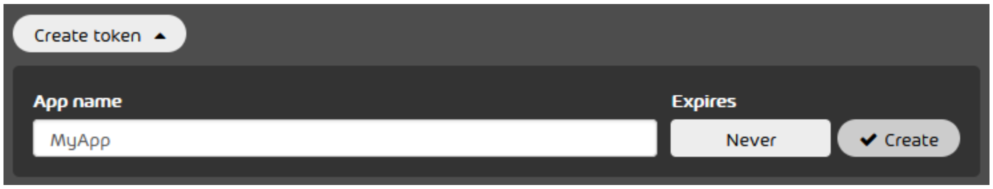


Figure 3: Enter app name

6. Optional: Under **Expires**, select an expiration date for the token.

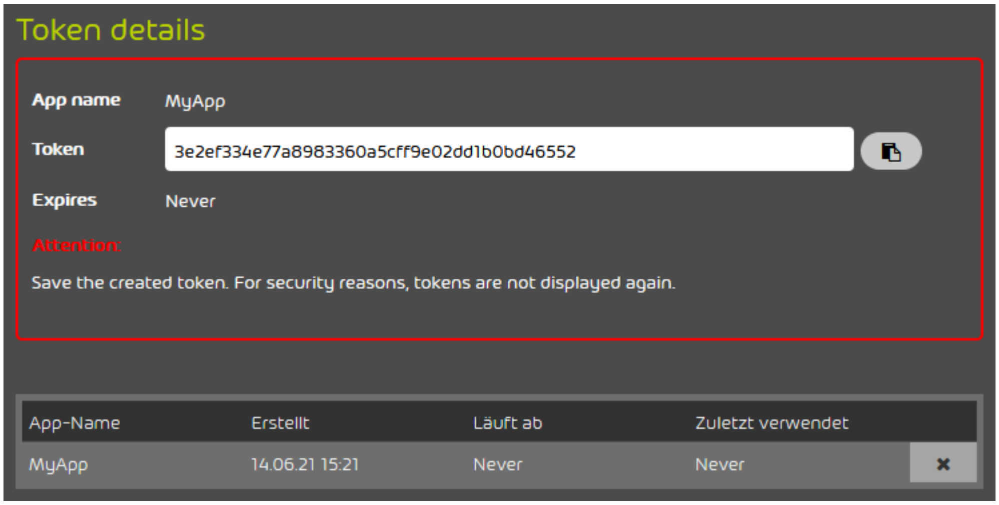
> **Notice:**
>
> **Never** is selected as the default value.

7. **CAUTION:**

   For security reasons, the token is only displayed once. If you do not save the token, it will be lost. Please save the token right away.

   Click on **Create**.

   The API token is created and displayed.



Token details

App name MyApp

Token 3e2ef334e77a8983360a5cff9e02dd1b0bd46552

Expires Never

Attention:

Save the created token. For security reasons, tokens are not displayed again.

App-Name	Erstellt	Läuft ab	Zuletzt verwendet
MyApp	14.06.21 15:21	Never	Never

Figure 4: Created token

An API token has been created.

If you no longer need an API token, you can delete it from the Control Panel (see **Deleting an API Token**).

Performing a Login Request

You can log in to the Control Panel with your credentials using our API.

Important:

The credentials must be valid.

Execute the following request. Replace the placeholders with your data.

```
curl --request POST \ --url https://cp.<your domain>.com/api/v0/auth/login/ \ --header
 'app-id: <your app ID>'\ --header 'app-version: <your app version following the pattern
 X.X.X.X>'\ --header 'cp-token: <your CP token>\ --header 'content-type: application/json'\ --
data'{ "username": "<your email address>", "password": "<Your password>" }'
```

The request is sent. If the login is successful, a response according to the following pattern is returned.

```
{ "object_id": 133456, "token": "b332ec9e19b88f8fdf5bfc1def03291f361c479d" }
```

Notice:

The object ID (**object_id**) is the ID for an application scope and is only required for some scope-related requests. The issued token must be sent with any future request as a means of authorization.

Notice:

If the response contains an error message, check the status code (see HTTP Status Codes on page 8) and perform an error analysis (see Identifying and Solving Errors on page 21). If the problems persist, please contact Support.

The Control Panel login has been performed.

Example of an authorized request

```
curl --request GET \ --url 'https://cp.<your domain>.com/api/v0/user/format/?
object_id=12345678'\ --header 'app-id: 0123456789'\  --header 'app-version:
```

```
X.X.X.X'\  --header 'cp-token: m31nCpT0k3n'\ --header 'authorization: Token
b332ec9e19b88f8fdf5bfc1def03291f361c479d'
```

Performing a Logout Request

You can log out from the Control Panel using our API.

Execute the following request. Replace the placeholders with your data.

```
curl --request POST \ --url https://cp.<your domain>.com/api/v0/auth/logout/ \ --header
 'app-id: <your app ID>'\ --header 'app-version: <your app version following the pattern
 X.X.X.X>'\ --header 'cp-token: <your CP token>'\ --header 'authorization: Token <your
 authorization token>'
```

Notice:

The authorization token is issued after logging in to the Control Panel (Performing a Login Request on page 13).

The request is sent. If the request is successful, a response with the status code 200 is returned. The authorization token then becomes invalid and can no longer be used.

Notice:

If the response contains an error message, check the status code (see HTTP Status Codes on page 8) and perform an error analysis (see Identifying and Solving Errors on page 21). If the problems persist, please contact Support.

A logout from the Control Panel has been performed.

First Queries

This section describes the first requests with examples. The examples will help you get familiar with , and they can be used as a basis for all desired requests. For a complete documentation about all possible API endpoints, see https://cp.<your domain>.com/api/v0/docs/.

As already mentioned in the section First Steps on page 10, the following information must be sent together with every request:

- App ID
- App version
- CP Token
- Username
- Password
- Object ID
- Token

Querying the Version Number

You can query the current version number of the Control Panel.

Execute the following request. Replace the placeholders with your data.

```
curl --request GET \ --url https://cp.<your domain>.com/api/v0/version/ \ --header 'app-id:
 <your app ID>'\ --header 'app-version: <your app version following the pattern X.X.X.X>'\
 --header 'cp-token: <your CP token>'\ --header 'authorization: Token <your authorization
 token>'
```

The version number is displayed. For example:

```
6.17.0.0
```

The version number of the Control Panel has been retrieved.

Querying an Object ID

Some domain-specific requests can be only performed if the user role is authorized to make the request and the correct application scope is selected. If the user is not authorized to make a request, an error message with the status code 400 will be returned as a response. In this case, the application scope must be changed in the same way as in the Control Panel (see Changing the Application Scope on page 23). The application scope is determined by the object ID. You can query the object ID via the **/hierarchy/** endpoint and use the object id for upcoming requests.

Perform the following request. Replace the placeholders with your data.

```
curl --request POST \ --url https://cp.<your domain>.com/api/v0/hierarchy/_search/ \
 --header 'app-id: <your app ID>'\ --header 'app-version: <your app version following
 the pattern X.X.X.X>'\ --header 'cp-token: <your CP token>'\ --header 'authorization:
 Token <your authorization token>'\ --header 'content-type: application/json'\ --
data'{ "search_string":"<your domain>", "filter":"customer" }'
```

The requested data is displayed. The data contains the object ID (**id**).

```
[ { "id":1111111, "name":"partner", "is_user":false, "children_count":116, "children":
[ { "id":2222222, "name":"<your domain>", "is_user":false, "children_count":289 } ] } ]
```

The object ID of the domain has been retrieved.

Next, you can use the object ID to execute domain-specific queries.

Retrieving Emails

You can search for emails from the Control Panel and retrieve the search results.

Execute the following request. Replace the placeholders with your data.

```
curl --request POST \ --url 'https://cp.<your domain>.com/api/v0/emails/_search/?
object_id=<your object ID>'\  --header 'app-id: <your app ID>'\ --header 'app-version:
 <your app version following the pattern X.X.X.X>'\ --header 'cp-token: <your CP token>'\
 --header 'authorization: Token <your authorization token>'\ --header 'content-type:
 application/json'\ --data '{ "limit": 50, "date_from": "<start date and start time in the
 format 2018-06-21T00:00:00Z>", "date_to": "<end date and end time in the format
 2018-07-21T00:00:00Z>" }
```

Your emails are retrieved and displayed in the response:

```
{ "source_ip":"94.100.135.100", "gateway":"mx-gate70-hz2", "comm_partner":"<email
address of the communication partner>", ... "classification":{ "text":"clean", ... },
"owner":"<email address of the email owner>", "encryption":{ "text":"EMIG", ... },
"date":"<email date and time in the format 2018-07-18T09:14:56Z>", "subject":"<email
subject>", ... }
```

Emails have been retrieved from the Control Panel.

Retrieving Email Statistics

You can retrieve the email statistics of a selected period from the Control Panel.

Execute the following request. Replace the placeholders with your data.

```
curl --request POST \ --url 'https://cp.<your domain>.com/api/v0/emails/statistics/
by_time/?object_id=12345678'\ --header 'app-id: <your app ID>'\ --header 'app-
version: <your app version following the pattern X.X.X.X>'\ --header 'cp-token: <your
 CP token>'\ --header 'authorization: Token <your authorization token>'\ --header
 'content-type: application/json'\ --data '{ "date_from": "<start date and start time in
 the format 2018-06-21T00:00:00Z>", "date_to": "<end date and end time in the format
 2018-07-21T00:00:00Z>" }'
```

You have successfully retrieved the email statistics from the Control Panel.

```
[ { "type":2, "value":45, "name":"clean" }, { "type":1, "value":4, "name":"spam" },
 { "type":11, "value":0, "name":"info" }, { "type":8, "value":0, "name":"content" },
 { "type":12, "value":0, "name":"atp" }, { "type":5, "value":0, "name":"virus" }, { "type":3,
 "value":0, "name":"rejected" } ]
```

Email statistics have been retrieved from the Control Panel.

Querying Customizations

You can query the settings from the **Customization** module for your domain from the Control Panel.

Execute the following request. Replace the placeholders with your data.

```
curl --request GET \ --url 'https://cp.<your domain>.com/api/v0/whitelabeling/?
object_id=<your object ID>'\ --header 'app-id: <your app ID>'\ --header 'app-version: <your
 app version following the pattern X.X.X.X>'\ --header 'cp-token: <your CP token>'\ --header
 'authorization: Token <your authorization token>'
```

Notice:

In order to query information about customization from the **/whitelabeling/** endpoint, you need the object ID (**object_id**) of the domain.

Notice:

To retrieve customization information for an unknown domain, use the **/whitelabeling/infos/** endpoint instead of **/whitelabeling/**. This endpoint provides the current customization information based on the host header.

The customization information is displayed. The code is shortened in the following example.

```
{ "theme":"dark", "primary_colour":"#15358B", "favicon":"data:image/x-
icon;base64,AAAB...", "domain":"<domain of the Control Panel>", "logo":"data:image/
png;base64,iVBOR...", "email_sender":"", "id":542 }
```

Notice:

The logo and favicon are rendered as base64-encoded strings in the response.

The domain customizations have been retrieved from the Control Panel.

Troubleshooting

Identifying and Solving Errors

On API requests with incorrect or missing parameters, our API returns error messages. To be able to interpret these errors correctly, status codes are displayed in the error messages. You can use the status codes to determine what type of error affects your request (see HTTP Status Codes on page 8). Since the status codes are mostly very generic, the errors are explained in more detail.

In order to properly interpret the error messages and take suitable measures, we recommend the following procedure.

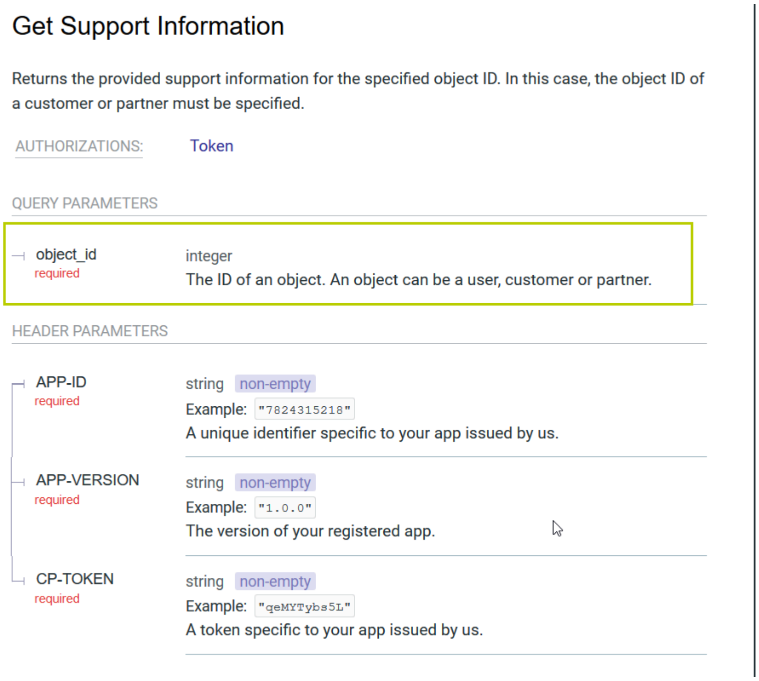
1. Check the status code (see HTTP Status Codes on page 8) and try to find out the cause based on the error message.

2. Check the API documentation (see https://cp.<your domain>.com/api/v0/docs/) to see which parameters are required for the request.

   Error message example

```
{ "error_data":{ "object_id":[ "This field is required." ] },
"error_message":"Validationerror", "error_id":2009, "status_code":400 }
```

   In this example, the object_id is missing, which is required for the support request according to the API documentation.



Get Support Information

Returns the provided support information for the specified object ID. In this case, the object ID of a customer or partner must be specified.

AUTHORIZATIONS: Token

QUERY PARAMETERS

object_id
required
integer
The ID of an object. An object can be a user, customer or partner.

HEADER PARAMETERS

APP-ID
required
string non-empty
Example: "7824315218"
A unique identifier specific to your app issued by us.

APP-VERSION
required
string non-empty
Example: "1.0.0"
The version of your registered app.

CP-TOKEN
required
string non-empty
Example: "qeMYTybs5L"
A token specific to your app issued by us.

Figure 5: Check API documentation

3. If you cannot solve the error on your own, please contact Support.

Errors in API requests have been identified and solved.

Changing the Application Scope

You have no permission to access domain-specific information.

Missing permission

Some domain-specific queries can only be performed if the user role is authorized to make the request and the correct domain is selected (see Querying an Object ID on page 17).

Index