User Manual

A roadmap to europisti's service requests

Vasilis Stergioulis

Version 1, Sep 10, 2019

Table of Contents

If you find errors or omissions in this document, please don't hesitate to submit an issue or open a pull request with a fix. We also encourage you to ask questions and discuss any aspects of the project on the mailing list or in the chat room. New contributors are always welcome!

Release notes

10/09/2019, v1, initial release

Chapter 1. Introduction

The Big Picture

The api gateway is a place where European Reliance's API's of web services as APIs are exposed together with a portal help site for developers.

What are the web services APIs?

Behind the gateway, exposed, are our enterprise insurance services, which can be used by external developers, for applications that produce and consume insurance data, consumed by external parties.

Namely, it is:

service-requests-api

: Service for service requests.

Chapter 2. API gateway

In front of all the services there is a common layer called ***gateway***. It is enforcing constraints common to all services like security, accounting and rate limiting.

Gateway is responsible for:

- The first level of authentication and authorization of every call to the internal services
- Counting the invocations and rate limiting them.

It is essentially the gateway guard that every call must show its credentials in order to pass.

Each API call passes from two levels of authentication and authorization.

2.1. Authentication and authorization

Levels of authentication and authorization

- First level: each call is being checked to see two things. if it contains an API key that is authorized to access the specified API resource and if it obeys some specified rate limiting access rules. If it doesn't then a specific gateway error response is returned (see General Web Service Errors and Web Service Rate Limits).
- Second level: Depending on the service, each call is checked if it contains an Authorization header with a valid Bearer ***token***.

An authorization token is fetched first from a specific service endpoint.

Steps

If a company user wants to use an externally provided program (your program) he must first authorize it to impersonate him by following a specific procedure through our company's portal.

Within that procedure the user publishes a **user-key**, a 40 character key, that ties his identity with the identity of your program. This bond can be enabled, disabled, deleted, and recreated by the user whenever he wants.

The user is instructed to deliver this user-key to your program in order for you to use it for the purpose of obtaining an authorization token from the API.

This authorization token together with your api-key must be supplied as header parameters with

any API call.

2.1.1. First check

At first, you must obtain an API key that uniquely identifies you, by following the procedure of signing up. After approval you will be given access to certain provided APIs.

2.1.2. Second check

The second level of authorization is the API specific authorization. Each call is checked to see if it contains an authorization header. If not then a `401 Unauthorized` error response is returned.

The authorization header is an `“Authorization: Bearer xxxxxxxxxxxxxxxxxxxxxxxx”` where the xxxx part is the token and is obtained by accessing a specific “tokens” address (see later). Steps

Please take care of your api-key, the client user-key and the resulting authorization tokens are to be stored safely and never be shared with third parties.

Use always secure communication protocols

Our services are served under HTTPS only. If you believe that your api-key or any other key or token got leaked to the outside world contact us immediately. We will disable them and supply you with new one.

If we don’t disable them, they may be used in an illegal way by a third party without us having a way to differentiate a valid usage from an invalid one. If you want your api-key to be tied permanently with your IP for added security, contact us.

2.2. API Key Usage

How to use your API key after signing up.

After signing up, you’ll be given your own, unique API key which is a 40 character string. The key:

- Uniquely identifies you.

- Gives you access to europisti.gr's Web services.
- Should be kept private and should not be shared.

The API may be used several ways. In order of precedence:

HTTP Header, the preferred way

Pass the API key into the X-api-key header:

```
curl -H 'X-api-key: DEMO_KEY' "https://gateway.europisti.gr/dev/service-requests-api/resources/parameters"
```

GET Request Query Param

To use your key, simply pass the key as a URL query parameter when making Web service requests. For example:

```
curl https://gateway.europisti.gr/dev/service-requests-api/resources/?api_key=YOUR_KEY"
```

Regardless of the HTTP method being called, the API key can always be passed as a GET parameter in the URL query. So even if you are POSTing or PUTing to a specific service, the api_key query parameter can be supplied in the URL query parameters.

Alternative Method

Depending on your usage, it can sometimes be easier to pass the API key along as HTTP Basic authentication. If you want to use this method, pass your API key in as the username, while leaving the password blank. For example:

```
curl "http://YOUR_KEY_HERE@gateway.europisti.gr/dev/service-requests-api/resources"
```

2.3. Web Service Rate Limits

Daily and hourly rate limits on accessing europisti.gr's Web services.

Rate Limits

There is a limit on the number of europisti.gr web service requests you can make using your API key. Rate limits may vary by service, but the defaults are:

```
Hourly Limit: 1,000 requests per hour
```

For each API key, these limits are applied across all europisti.gr web services requests. Exceeding these limits will lead to your API key being temporarily blocked from making further requests. Depending on the limit exceeded, the block will be lifted automatically by waiting until the next hour or calendar day.

2.3.1. How Do I See My Current Usage?

Your can check your current rate limit and usage details by inspecting the X-RateLimit-Limit and X-RateLimit-Remaining HTTP headers that are returned on every API response. For example, if an API has the default hourly limit of 1,000 request, after making 2 requests, you will receive these HTTP headers in the response of the second request:

```
X-RateLimit-Limit: 1000
X-RateLimit-Remaining: 998
```

2.3.2. Understanding Rate Limit Time Periods

Hourly Limit

The hourly counters for your API key reset on a rolling basis. Example:

1. If you made 500 requests at 10:15AM and 500 requests at 10:25AM, your API key would become temporarily blocked.
2. This temporary block of your API key would cease at 11:15AM, at which point you could make 500 requests.
3. At 11:25AM, you could then make another 500 requests.

2.3.3. Rate Limit Error Response

If your API key exceeds the rate limits, you will receive a response with an HTTP status code of `429 (Too Many Requests)`.

2.3.4. Need Higher Limits?

If you’re building an application that needs higher rate limits, we’d be happy to work with you. Contact us for more details.

2.3.5. Errors at gateway's level

Certain, general errors will be returned in a standardized way from all web services. Additional, service-specific error messages may also be returned (see individual service documentation for those details). The following list describes the general errors any application may return:

Error Codes

Error Code	HTTP Status Code	Description
API_KEY_MISSING	403	An API key was not supplied. Read API key usage for details on how to pass your API key to the API.
API_KEY_INVALID	403	An invalid API key was supplied. Double check that the API key being passed in is valid, or sign up for an API key.
API_KEY_DISABLED	403	The API key supplied has been disabled by an administrator. Please contact us for assistance.
API_KEY_UNAUTHORIZED	403	The API key supplied is not authorized to access the given service. Please contact us for assistance.
API_KEY_UNVERIFIED	403	The API key supplied has not been verified yet. Please check your e-mail to verify the API key. Please contact us for assistance if needed.
HTTPS_REQUIRED	400	Requests to this API must be made over HTTPS. Ensure that the URL being used is over HTTPS.
OVER_RATE_LIMIT	429	The API key has exceeded the rate limits. Read rate limits for more details or contact us for assistance.
NOT_FOUND	404	An API could not be found at the given URL. Check your URL.

Error Response Body

The error response body will contain an error code value from the table above and a brief description of the error. The descriptions are subject to change, so it's suggested that any error handling must use the HTTP status code or the error code value for error handling (and not the content of the message description).

Error Message Response Formats

Depending on the detected format of the request, the error message response may be returned in

JSON, XML, CSV, or HTML. Requests of an unknown format will return errors in JSON format.

JSON Example

```
{
 "error": {
  "code": "API_KEY_MISSING",
  "message": "No api_key was supplied. Get one at https://gateway.europisti.gr"
 }
}
```

XML Example

```
<response>
 <error>
  <code>API_KEY_MISSING</code>
  <message>No api_key was supplied. Get one at https://gateway.europisti.gr</message>
 </error>
</response>
```

CSV Example

```
Error Code,Error Message
API_KEY_MISSING,No api_key was supplied. Get one at https://gateway.europisti.gr
```

HTML Example

```
<html>
 <body>
  <h1>API_KEY_MISSING</h1>
  <p>No api_key was supplied. Get one at {uri-mtr-gw}</p>
 </body>
</html>
```

2.4. Dev vs Prod

In order to access a developer's sandbox for developing and testing reasons, it is enough just to add the word dev in the path after the server name and before the actual api's path, for example:

```
Production URI      https://gateway.europisti.gr/service-requests-api/resources
Sandbox URI         https://gateway.europisti.gr/dev/service-requests-api/resources
```

Of course it is necessary for your API key to be accepted and given access to both environments by the authorization procedures that are established with your registration.

By having a valid and authorized `X-api-key`, any call can pass gateway's checks and access any internal APIs that was given access to it. One such API is the mtr-api.

Chapter 3. SERVICE-REQUESTS-API intro

General information

The service-requests-api is accessible via https://gateway.europisti.gr/dev/service-requests-api/resources. In order to use this API, you must pass:

- Your API key via one of the supported methods, preferably as header parameter
- AND an authorization token via the Authorization header.

To make requests to the API two headers are required:

```
X-Api-Key:    YOUR_API_KEY_HERE
Authorization: Bearer CLIENT_BEARER_TOKEN_HERE
```

Without these two headers all calls to mtr-api will be rejected with the response `401-Unauthorized`

3.1. Authorization

3.1.1. Getting authorization tokens

An Authorization client bearer token is thus required, and the way to get it is by using MTR_API token service.

A convenience header `X-Token-Expires-In: xxxx seconds` is returned by default in every request. It is the time remaining in seconds after which the token will be valid. With this header it is possible to calculate beforehand when a token renewal call will be needed.

By default the token is valid for 30 minutes after which the server will reject subsequent calls with the response `401 Unauthorized` and an extra error header `X-Error: Token Expired`

If the client receives this error he can request a new token which will be valid for 30 more minutes.

No extra header

If the client gets response `401 Unauthorized` without the extra header, then that means the authentication failed on the server side due to either wrong keys or because the client of the user-key refuses access to the X-Api-Key holder. In such case it is advised to first get in contact with your client (of user-key) to remedy the situation.

string ISO date

When the specification of a date named field is **“string ISO date”**, it is expected to be supplied with a date in ISO 8601 format like “2018-10-28” or a datetime like “2018-10-28T13:55:00”.

The date fields are expected to be in the Greek time zone and the conversions are handled by the api. When a datetime field is returned and it doesn’t end with a 'Z' (which denotes a UTC date) then it is supposed to be a *Greek LocalDate* value.

3.2. Errors

When the parties-api encounters an error condition outside the normal flow, it tries to return that error condition with a specific header named **`X-Error`**. So when it encounters incomplete or bad requests, incoming data validation errors or specific internal error conditions, it sets the error header informing the client application with a more specific message.

You will use this information for correcting the condition and maybe retrying the operation. For example when the authentication token expires, the client application receives a `401 Unauthorized` status with the specific header **`X-Error`**`: Token Expired` signalling that a new token must be obtained first from the “tokens” resource and then the operation can be retried.

3.2.1. Other headers

When an operation returns with a failure status, it is possible that a header named **X-Error** is returned and usually an error entity with error details for both.

For operations like POSTing new entities and for the success status **201 Created**, a **Location** header is set, according to standards with the uri of the new entity.

Some operations return extra information or actions in the Link headers. For example when issuing an application a link for the notification document is included. The operations that return this kind of information are documented.

For reference purposes, click this link to find out more about known response status codes.

3.3. CORS

The parties-api supports CORS and set required headers.

Chapter 4. Service requests

Overview

The following API endpoints can be used to programmatically create service requests.

Available actions are:

Insert a new service request

Create a new service requests.

Guides

The api informs you for any payload validation errors or other type of errors with an error status and an entity.

4.1. Insert

Fields category, subject and body are mandatory. The service will insert a new service request. On success returns the serviceRequestId.

For the available categories check parameters

Resource URL

POST https://gateway.europisti.gr/dev/service-requests-api/resources/srs

Resource Information

Response formats	JSON
Requires authentication?	Yes (user context)
Rate limited?	Yes

Example Request

```
curl -H 'X-Api-Key: your_api_key' \
-H 'Authorization: Bearer user_token' \
-H 'Content-type: application/json' \
-X POST "https://gateway.europisti.gr/dev/service-requests-api/resources/srs"
-d '{
  "category": "MOTOR",
  "subject": "ΚΑΠΟΙΟ ΘΕΜΑ",
  "body": "ΚΕΙΜΕΝΟ ΘΕΜΑΤΟΣ"
}'
```

Example Error Response


```
{
 "status": "ERROR",
 "error": "-2",
 "message": "...."
}
```


Example Success Response


```
{
  "serviceRequestId": "685638"
}
```

Chapter 5. Parameters

The following API endpoints can be used to programmatically retrieve parameters

Overview

All of entity properties that are supposed to be used as input fields, have a method in parameters resource that returns the acceptable values usually with descriptions. All of the methods support a GET request. Available calls are:

Methods

Method	Description
…/parameters/categories	Categories for service requests

Guides

Calls that return almost static data are: **categories**

5.1. categories

Categories for service requests. Depending on the category, a request may follow different delivery paths. It is used in order to simplify proccessing for the caller.

Resource URL

GET https://gateway.europisti.gr/dev/service-requests-api/resources/parameters/categories

Resource Information

Response formats	JSON
Requires authentication?	Yes (user context)
Rate limited?	Yes

Example Request

```
curl -H 'X-Api-Key: your_api_key' \
-H 'Authorization: Bearer user_token' \
"https://gateway.europisti.gr/dev/service-requests-api/resources/parameters/categories"
```

Example Response


```
[
  {
    "id": "MOTOR",
    "description": "Αυτοκίνητα"
  },
  {
    "id": "MOTOR-CLAIMS",
    "description": "Ζημιές αυτοκινήτων"
  },
 ...
]
```