TorneAPI v3.0

TorneAPI v3.0 is very much reminding of TorneAPI v2.0. However, authentication and requests are rebuilt from scratch to meet new requirements (actually my own needs). The response interface looks a bit different, but much of it is compatible.

Table of contents

What's new?

The big update for APIv3, is actually the control panel that is reachable via https://auth.tornevall.net (when writing this, AUTHv3 does not support this, so you can also check out the development site for AUTHv4 at https://auth.tornevall.com). Have in mind that, since auth.tornevall.com is only a development /testing site, it might not be up all the time. Through TorneAUTHv4, you should be able to configure the most of the new API, which is not possible with the old releases.

Another new thing with API v3 is the ability to use basic authentication instead of posting authentication paramteters through POST/GET requests.

API URLs

Production URL: https://api.tornevall.net/3.0

Development URL: https://api.tornevall.com/3.0

Deprecated API: https://api.tornevall.net/2.0

Deprecated Development API: https://api.tornevall.nu/2.0

API Syntax

A standard request to the API could look like this:

https://api.tornevall.net/3.0/test/returnRandom

This is the simplest request mode for the API and returns default responses like this:

Example request


```
{
        "response": {
                "endpoint": {
                        "name": "test",
                        "verb": "returnRandom",
                        "request": [],
                        "misc": []
                },
                "returnRandomResponse": "42666"
        },
        "errors": {
                "code": "200",
                "success": "1",
                "faultstring": ""
        }
}
```

As you can see, errors are always returned, but the success value is only set to 1 (true) when the call is successful. The code returned from the errors object is not necessarily always indicating an error; if the call is successful the API returns the samve (default) response code as a webserver usually do - 200. Checking for errors can be done by checking the success response (that should be empty - or false) when errors occurs, together with a non empty faultstring.

All successful responses are stored in the "response" object. The endpoint are also returned here, as we sometimes want to know asynchronously where the response was returned. Responses are also returned with the format of **<verb>Response**. So if you request, as in the example above, for a random value at the test endpoint (returnRandom), you should look for the **returnRandomResponse**.

Simulate an error

It's easy to simulate a standard error by calling a non existing endpoint and/or a non existing verb, like below:

https://api.tornevall.net/3.0/notAnEndPoint/notTheProperVerb

Errors

```
{
    "response": [],
    "errors": {
        "code": "404",
        "success": "",
        "faultstring": "Invalid endpoint"
    }
}
```

Using the test endpoint API to validate requests

The test API is integrated in the API core (that might be released in the future) and contains a "key validator". Once, when you registered a client application at TorneAUTHv4, the key validator can be used, something like this:

https://api.tornevall.net/3.0/test/key/application/<myTestClientApplication>/authKey/<tehAuthKey>

The output response might look like this:

Key validator


```
{
    "response": {
        "endpoint": {
            "name": "test",
            "verb": "key",
            "request": [],
            "misc": []
        },
        "keyResponse": {
            "appClientData": {
                "AUTHORIZED": "1",
                "HAS_KEY": "1",
                "HAS_SECRET": "",
                "HAS_TOKEN": "",
                "IS_ADMIN": "",
                "HAS_DEVELOPER": "",
                "APPLICATION_ID": "3",
                "API_ENDPOINT_PERMISSIONS": [],
                "API_EXTENDED_PERMISSIONS": []
            },
            "appServiceData": {
                "appid": "3",
                "classname": "test",
                "version": "3.0",
                "requireKey": "0",
                "requireSecret": "0",
                "requireToken": "0",
                "requireAdmin": "0",
                "description": "Testing API",
                "requireEndpointPermission": "0",
                "applicationRegisterDate": "2016-10-25 21:20:31",
                "applicationLastUsage": "2018-07-03 14:52:47"
            }
        }
    },
    "errors": {
        "code": "200",
        "success": "1",
        "faultstring": ""
    }
}
```


The rendered response will give you information about the client you've registered and also lists information about extra permissions (**API_ENDPOINT_PERMISSIONS** and **API_EXTENDED_PERMISSIONS**) if they exists. Such permissions is formatted like this (example taken from our local application that handles DNSBL requests):

API Permissions example


```
                "API_ENDPOINT_PERMISSIONS": [
                    {
                        "clientid": "1",
                        "endpointid": "2",
                        "endpoint": "dnsbl"
                    },
                    {
                        "clientid": "1",
                        "endpointid": "204",
                        "endpoint": "application"
                    }
                ],
                "API_EXTENDED_PERMISSIONS": [
                    {
                        "clientid": "1",
                        "permissionid": "1",
                        "permission": "global_delist"
                    },
                    {
                        "clientid": "1",
                        "permissionid": "2",
                        "permission": "local_delist"
                    },
                    {
                        "clientid": "1",
                        "permissionid": "3",
                        "permission": "fraudbl_update"
                    }
                ]
```


You can also see what kind of authentication you're using.

When you fail to validate your client, the **appClientData**-object **AUTHORIZED** will be shown as empty (false).

Not validated request


```
{
    "response": {
        "endpoint": {
            "name": "test",
            "verb": "key",
            "request": [],
            "misc": []
        },
        "keyResponse": {
            "appClientData": {
                "AUTHORIZED": "",
                "HAS_KEY": "",
                "HAS_SECRET": "",
                "HAS_TOKEN": "",
                "IS_ADMIN": "",
                "HAS_DEVELOPER": "",
                "APPLICATION_ID": "3",
                "API_ENDPOINT_PERMISSIONS": [],
                "API_EXTENDED_PERMISSIONS": []
            },
            "appServiceData": {
                "appid": "3",
                "classname": "test",
                "version": "3.0",
                "requireKey": "0",
                "requireSecret": "0",
                "requireToken": "0",
                "requireAdmin": "0",
                "description": "Testing API",
                "requireEndpointPermission": "0",
                "applicationRegisterDate": "2016-10-25 21:20:31",
                "applicationLastUsage": "2018-07-03 14:58:08"
            }
        }
    },
    "errors": {
        "code": "200",
        "success": "1",
        "faultstring": ""
    }
}
```


Available endpoints (API-Browser API: endpoints)

When your application are registered and ready to roll, you can get a list (this is the API Browser API "**endpoints**") of AP Endpoints with this URL:

https://api.tornevall.net/3.0/endpoints/

By time and upgrades those responses might change but the base response (example) might look like this:

Endpoint list request example


```
          "endpointsResponse": {
            "enabledEndpoints": [
                "application",
                "test",
                "vz",
                "urltest",
                "endpoints",
                "test20",
                "captcha",
                "spamassassin",
                "dnsbl",
                "sms"
            ],
            "disabledEndpoints": [],
            "describedEndpoints": {
                "application": "Application handler",
                "test": "Standard API tester",
                "vz": "Tornevall Networks VPS (OpenVZ) Configurator",
                "urltest": "URL Testing Endpoint (Availability)",
                "endpoints": "Endpoint Viewier",
                "test20": "Standard API tester (API 2.0 compatible)",
                "captcha": "Simple captcha module",
                "spamassassin": "Antispam configurator for Tornevall Networks",
                "dnsbl": "Tornevall Networks DNSBL API",
                "sms": "SMS Sender application"
            },
            "endpointStates": [],
            "endPointsVerifyEntry": []
        }
```


To get available verbs/methods from an API getEndpointMethods can be used (https://api.tornevall.net/3.0/endpoints/getEndpointMethods/endpoint/test/).

> *Note: Some returned data are related to APIv2 and may be removed in the future (like the list of variables). Also note that the "behaviours" object are based on inline docs and might not be the correct return data. This is also related to a former APIv2 system.*

Endpoint methods request


```
            "getEndPointMethodsResponse": [
            {
                "methods": [
                    "key",
                    "getVersion",
                    "returnRandom",
                    "returnHtml",
                    "returnRandomArg",
                    "returnFailure"
                ],
                "descriptions": {
                    "getVersion": "Get version of this application",
                    "returnRandom": "Return a random value to test functionality in TorneAPI v2",
                    "returnRandomArg": "",
                    "returnFailure": "Return an error to the API endpoint handler to test functionality in
TorneAPI v2"
                },
                "behaviours": {
                    "returnRandom": {
                        "return": "int"
                    },
                    "returnRandomArg": {
                        "return": "array"
                    },
                    "returnFailure": {
                        "throws": "\\Exception"
                    }
                },
                "variables": [
                    "BaseURL",
                    "BaseURI",
                    "DESCRIPTION",
                    "APPLICATION_VERSION",
                    "HTML",
                    "TorneLIB",
                    "htmlHead",
                    "htmlFoot"
                ]
            }
        ]
```