PHP Yousign API client

Jun 08, 2023

Contents

Yousign API client is a wrapper for the Yousign API v2 in PHP.

Its purpose is to use this API without having to write the HTTP calls yourself and to retrieve the returned data through an object model.

As all features are implemented, it aims to be a full-featured client.

```php
use Yousign\YousignApi;

/*
 * token
 */
$token = '123456789';

$yousign = new YousignApi($token);

$users = $yousign->getUsers();

foreach ($users as $user) {
    echo $user->getId();
}
```

CHAPTER 1

User Guide

1.1 Overview

All API calls are wrapped into an object model.

If you still want to make HTTP calls to check the API responses, this is possible thanks to the low-level calls.

It provides an API wrapper (*Yousign\YousignApi*) and some shortcut methods for basic and advanced modes.

1.1.1 Requirements

- PHP 7.1+
- Before using this library, you have to create your account on Yousign platform to get an API token before using this library.

1.1.2 Install

```
composer require landrok/yousign-api
```

1.1.3 Production & staging environments

YousignApi accepts 2 parameters:

- *string $token* is the bearer token
- *bool $production*
  - ***false* is the default value, for staging environment. Requests** are sent to https://staging-api.yousign.com/
  - *true* for production environment. Requests are sent to https://api.yousign.com/

Staging environment

```php
use Yousign\YousignApi;

/*
 * token
 */
$token = '123456789';

/*
 * production flag
 */
$production = false;

$yousign = new YousignApi($token, $production);
```

Production environment

```php
use Yousign\YousignApi;

/*
 * token
 */
$token = '123456789';

/*
 * production flag
 */
$production = true;

$yousign = new YousignApi($token, $production);
```

1.1.4 Contributing

All subsequent types (Member, Procedure, File, FileObject, etc. . .) are implemented too.

- Contribute on Github
- To discuss new features, make feedback or simply to share ideas, you can contact me on Mastodon at https://cybre.space/@landrok

1.1.5 Yousign API manual

Official Yousign API manual

1.2 Quickstart

This page provides short examples on how to make your first call and retrieve your data.

1.2.1 Making your first call

In this example, we will get all users in staging mode.

```
use Yousign\YousignApi;

/*
 * token
 */
$token = '123456789';

$yousign = new YousignApi($token);

$users = $yousign->getUsers();
```

$users contains an iterable *UserCollection* object.

All API responses are converted into objects (See *api_reference*).

All of them offer *toArray()*, *toJson()* methods and getters to access their properties.

Collection objects are iterable.

toArray()

You can use toArray() method to dump all data as a PHP array.

```
print_r(
    $users->toArray()
);
```

toJson()

You can use toJson() method to serialize all data as a JSON object.

```
echo $users->toJson();
```

Iterate over a collection

You can iterate over all items of a collection.

```
foreach ($users as $user) {
    /*
     * For each User model, some methods are available
     */

    // toArray(): to get all property values
    print_r($user->toArray());

    // get + property name
    echo PHP_EOL . "User.id=" . $user->getId();

    // property (read-only)
    echo PHP_EOL . "User.id=" . $user->id;
```

(continues on next page)

(continued from previous page)

```
    // Some properties are models that you can use the same way
    echo PHP_EOL . "User.Group.id=" . $user->getGroup()->getId();
    echo PHP_EOL . "User.Group.id=" . $user->group->id;

    // Some properties are collections that you can iterate
    foreach ($user->group->permissions as $index => $permission) {
        echo PHP_EOL . "User.Group.Permission.name=" . $permission->getName();
    }

    // At any level, you can call a toArray() to dump the current model
    // and its children
    echo PHP_EOL . "User.Group=\n";
    print_r($user->group->toArray());
    echo PHP_EOL . "User.Group.Permissions=\n";
    print_r($user->group->permissions->toArray());
}
```

1.3 Basic mode

Let’s create your first signature procedure in basic mode.

In this example, we will accomplish this mode with low-level features.

```
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';


/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * 1st step : send a file
 */
$file = $yousign->postFile([
    'name'    => 'My filename.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/my_storage_path/test-file-1.pdf'
        )
    )
]);

/*
```

(continues on next page)

(continued from previous page)

```
 * 2nd step : create the procedure
 */
$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Awesome! Here is the description of my first procedure",
    "members"     => [
        [
            "firstname" => "John",
            "lastname" => "Doe",
            "email" => "john.doe@yousign.fr",
            "phone" => "+33612345678",
            "fileObjects" => [
                [
                    "file" => $file->getId(),
                    "page" => 2,
                    "position" => "230,499,464,589",
                    "mention" => "Read and approved",
                    "mention2" => "Signed by John Doe"
                ]
            ]
        ]
    ]
]);

// toJson() supports all PHP json_encode flags
echo $procedure->toJson(JSON_PRETTY_PRINT);
```

When the procedure is created, you can retrieve all the data with the getters or dump all data with *toJson()* and *toArray()* methods.

It would output something like:

```
{
    "id": "/procedures/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "name": "My first procedure",
    "description": "Awesome! Here is the description of my first procedure",
    "createdAt": "2018-12-01T11:49:11+01:00",
    "updatedAt": "2018-12-01T11:49:11+01:00",
    "finishedAt": null,
    "expiresAt": null,
    "status": "active",
    "creator": null,
    "creatorFirstName": null,
    "creatorLastName": null,
    "workspace": "/workspaces/XXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "template": false,
    "ordered": false,
    "parent": null,
    "metadata": [],
    "config": [],
    "members": [
        {
            "id": "/members/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "user": null,
            "type": "signer",
            "firstname": "John",
            "lastname": "Doe",
```

(continues on next page)

(continued from previous page)

```
            "email": "john.doe@yousign.fr",
            "phone": "+33612345678",
            "position": 1,
            "createdAt": "2018-12-01T11:49:11+01:00",
            "updatedAt": "2018-12-01T11:49:11+01:00",
            "finishedAt": null,
            "status": "pending",
            "fileObjects": [
                {
                    "id": "/file_objects/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                    "file": {
                        "id": "/files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                        "name": "The best name for my file.pdf",
                        "type": "signable",
                        "contentType": "application/pdf",
                        "description": null,
                        "createdAt": "2018-12-01T11:36:20+01:00",
                        "updatedAt": "2018-12-01T11:49:11+01:00",
                        "sha256":
↪"bb57ae2b2ca6ad0133a699350d1a6f6c8cdfde3cf872cf526585d306e4675cc2",
                        "metadata": [],
                        "workspace": "/workspaces/XXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                        "creator": null,
                        "protected": false,
                        "position": 0,
                        "parent": null
                    },
                    "page": 2,
                    "position": "230,499,464,589",
                    "fieldName": null,
                    "mention": "Read and approved",
                    "mention2": "Signed by John Doe",
                    "createdAt": "2018-12-01T11:49:11+01:00",
                    "updatedAt": "2018-12-01T11:49:11+01:00",
                    "parent": null,
                    "reason": "Signed by Yousign"
                }
            ],
            "comment": null,
            "notificationsEmail": [],
            "operationLevel": "custom",
            "operationCustomModes": [
                "sms"
            ],
            "operationModeSmsConfig": null,
            "parent": null
        }
    ],
    "subscribers": [],
    "files": [
        {
            "id": "/files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "name": "The best name for my file.pdf",
            "type": "signable",
            "contentType": "application/pdf",
            "description": null,
            "createdAt": "2018-12-01T11:36:20+01:00",
```

(continues on next page)

(continued from previous page)

```
            "updatedAt": "2018-12-01T11:49:11+01:00",
            "sha256":
↪"bb57ae2b2ca6ad0133a699350d1a6f6c8cdfde3cf872cf526585d306e4675cc2",
            "metadata": [],
            "workspace": "/workspaces/XXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "creator": null,
            "protected": false,
            "position": 0,
            "parent": null
        }
    ],
    "relatedFilesEnable": false,
    "archive": false,
    "archiveMetadata": [],
    "fields": [],
    "permissions": []
}
```

If you want to create your signature procedure in basic mode with a more high-level feature, see this manual (Coming soon).

1.4 Advanced mode

Here is how to create a procedure in 5 steps with the advanced mode.

```php
use Yousign\YousignApi;

/*
 * Token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * Step 1 - Create your procedure
 */
$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Description of my procedure with advanced mode",
    "start"       => false,
]);

/*
 * Step 2 - Add the files
 */
```

(continues on next page)

(continued from previous page)

```php
$file = $yousign->postFile([
    'name'    => 'Name of my signable file.pdf',
    'content' => base64_encode(
        file_get_contents(
            dirname(__DIR__, 2) . '/tests/samples/test-file-1.pdf'
        )
    ),
    'procedure' => $procedure->getId(),
]);

/*
 * Step 3 - Add the members
 */
$member = $yousign->postMember([
    "firstname"     => "John",
    "lastname"      => "Doe",
    "email"         => "john.doe@yousign.fr",
    "phone"         => "+33612345678",
    "procedure"     => $procedure->getId(),
]);


/*
 * Step 4 - Add the signature images
 */
$fileObject = $yousign->postFileObject([
    "file"      => $file->getId(),
    "member"    => $member->getId(),
    "position"  => "230,499,464,589",
    "page"      => 2,
    "mention"   => "Read and approved",
    "mention2"  => "Signed By John Doe"
]);

 /*
  * Step 5 - Start the procedure
  */
$procedure = $yousign->putProcedure(
    $procedure->getId(), [
        "start" => true,
    ]
);


echo $procedure->toJson(JSON_PRETTY_PRINT);
```

In step 3, you may add several members. In step 4, you may add one or more signature images for each one.

1.5 Advanced features

Here are the implementation of advanced features.

1.5.1 Attachment file

To add an attachment file in your procedure, simply add the type parameter with attachment value.

In the below example, we'll see how to configure the files within a procedure.

```php
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * Step 1 - Create your procedure
 */
$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Description of my procedure with advanced mode",
    "start"       => false,
]);

/*
 * Step 2 - Add a signable file
 */
$signableFile = $yousign->postFile([
    'name'    => 'Name of my signable file.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/signable-file.pdf'
        )
    ),
    'procedure' => $procedure->getId(),
]);

/*
 * Step 3 - Add an attachment file
 */
$attachmentfile = $yousign->postFile([
    'name'    => 'Name of my attachment file.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/attachment-file.pdf'
        )
    ),
    'procedure' => $procedure->getId(),
    'type'      => 'attachment',
]);
```

(continues on next page)

(continued from previous page)

```
// [...] then, you can add members, file objects and start your
// procedure
```

When the 3 steps are done, you can add members, file objects and start the procedure. See the steps 3, 4 ,5 in *Advanced mode*.

1.5.2 Members

It's necessary to highlight 2 categories of members:

- On the one hand, signers who are in your organization and who already have an account on the Yousign application. These participants are usually called **users** or **internal members**. These users are licensed with a paid account.
- On the other hand, signers who do not have access to the Yousign application. These are mainly your customers, partners, suppliers and so on. These participants are usually called **external members**.

Add an internal member

Previously, we saw how to add an external member to sign a document (See *Basic mode* or *Advanced mode*).

In the following example, we'll see how to attach an internal member (an user) to a procedure. You must know the user id. In this case, you don't need to add name, phone or email.

```
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * User id
 */
$userId = "/users/10d3730f-d056-422d-a8d1-a5252236246d";

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * Step 1 - Add a signable file
 */
$signableFile = $yousign->postFile([
    'name'     => 'Name of my signable file.pdf',
```

(continues on next page)

(continued from previous page)

```php
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/signable-file.pdf'
        )
    ),
    'procedure' => $procedure->getId(),
]);

/*
 * 2nd step : create the procedure with your internal member
 */
$procedure = $yousign->postProcedure([
    "name"        => "My procedure",
    "description" => "Awesome! Here is the description of my procedure",
    "members"     => [
        [
            "user" => $userId,
            "fileObjects" => [
                [
                    "file" => $file->getId(),
                    "page" => 2,
                    "position" => "230,499,464,589",
                    "mention" => "Read and approved",
                    "mention2" => "Signed by John Doe"
                ]
            ]
        ]
    ]
]);
```

Add an external member

This is just a reminder on how to add an external member. In this case, you don't need an user identifier, it's generated when the procedure is created.

```php
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
```

(continues on next page)

(continued from previous page)

```
 * Step 1 - Add a signable file
 */
$signableFile = $yousign->postFile([
    'name'    => 'Name of my signable file.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/signable-file.pdf'
        )
    ),
    'procedure' => $procedure->getId(),
]);

/*
 * 2nd step : create the procedure with your external member
 */
$procedure = $yousign->postProcedure([
    "name"        => "How to add an external member",
    "description" => "Simply with following information: first name, last name, email␣
↪address and phone number.",
    "members"     => [
        [
            "firstname" => "John",
            "lastname"  => "Doe",
            "email"     => "john.doe@yousign.fr",
            "phone"     => "+33612345678",
            "fileObjects" => [
                [
                    "file" => $file->getId(),
                    "page" => 2,
                    "position" => "230,499,464,589",
                    "mention" => "Read and approved",
                    "mention2" => "Signed by John Doe"
                ]
            ]
        ]
    ]
]);
```

Create a user

Before using this API client to create users, please consider this notes from the Yousign API For Developers site.

Warning: Our API makes it possible to create users but a fundamental concept to understand on this topic is that a user here is not only considered as a signer but also as a user of the Yousign application.

This implies that each user created by this means (API) or through the application will be billed according to your plan.

As a reminder, with each of our API plans you benefit from a free user with access to our application. Others will be billed.

The need to automatically create, via API, a user with access to the application can be interesting for software editors, resellers or large organizations who want to control their access centrally.

These are therefore very specific cases and in the majority of cases it's not necessary to use this feature. Before any development on your side, we invite you to contact our technical support to share your needs with us and we can advise you to achieve the best possible integration.

So, let's create users.

```
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * Create an user
 */
$user = $yousign->postUser([
    "firstname" => "John",
    "lastname" => "Doe",
    "email" => "api@yousign.fr",
    "title" => "API teacher",
    "phone" => "+33612345678",
    "organization" => "/organizations/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
]);

echo $user->toJson(JSON_PRETTY_PRINT);
```

```
{
    "id": "/users/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "firstname": "John",
    "lastname": "Doe",
    "email": "api@yousign.fr",
    "title": "API teacher",
    "phone": "+33612345678",
    "status": "not_activated",
    "organization": "/organizations/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "workspaces": [
        {
            "id": "/workspaces/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "name": "Acme"
        }
    ],
    "permission": "ROLE_MANAGER",
    "group": {
        "id": "/user_groups/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        "name": "Gestionnaire",
```

(continues on next page)

(continued from previous page)

```
        "permissions": [
            "procedure_write",
            "procedure_template_write",
            "procedure_create_from_template",
            "contact",
            "archive"
        ]
    },
    "createdAt": "2018-12-03T07:33:01+01:00",
    "updatedAt": "2018-12-03T07:33:01+01:00",
    "deleted": false,
    "deletedAt": null,
    "config": [],
    "inweboUserRequest": null,
    "samlNameId": null,
    "defaultSignImage": null,
    "notifications": {
        "procedure": true
    },
    "fastSign": false,
    "fullName": "John Doe"
}
```

Add a validator

Before using this API client to create users, please consider this notes from the Yousign API For Developers site.

Warning: Our API makes it possible to create users but a fundamental concept to understand on this topic is that a user here is not only considered as a signer but also as a user of the Yousign application.

This implies that each user created by this means (API) or through the application will be billed according to your plan.

As a reminder, with each of our API plans you benefit from a free user with access to our application. Others will be billed.

The need to automatically create, via API, a user with access to the application can be interesting for software editors, resellers or large organizations who want to control their access centrally.

These are therefore very specific cases and in the majority of cases it's not necessary to use this feature. Before any development on your side, we invite you to contact our technical support to share your needs with us and we can advise you to achieve the best possible integration.

So, let's create users.

```php
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

```

(continues on next page)

(continued from previous page)

```
/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);

/*
 * Create an user
 */
$user = $yousign->postUser([
    "firstname" => "John",
    "lastname" => "Doe",
    "email" => "api@yousign.fr",
    "title" => "API teacher",
    "phone" => "+33612345678",
    "organization" => "/organizations/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
]);

echo $user->toJson(JSON_PRETTY_PRINT);
```

```
{
    "id": "/users/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "firstname": "John",
    "lastname": "Doe",
    "email": "api@yousign.fr",
    "title": "API teacher",
    "phone": "+33612345678",
    "status": "not_activated",
    "organization": "/organizations/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "workspaces": [
        {
            "id": "/workspaces/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "name": "Acme"
        }
    ],
    "permission": "ROLE_MANAGER",
    "group": {
        "id": "/user_groups/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        "name": "Gestionnaire",
        "permissions": [
            "procedure_write",
            "procedure_template_write",
            "procedure_create_from_template",
            "contact",
            "archive"
        ]
    },
    "createdAt": "2018-12-03T07:33:01+01:00",
    "updatedAt": "2018-12-03T07:33:01+01:00",
    "deleted": false,
    "deletedAt": null,
    "config": [],
```

(continues on next page)

(continued from previous page)

```
    "inweboUserRequest": null,
    "samlNameId": null,
    "defaultSignImage": null,
    "notifications": {
        "procedure": true
    },
    "fastSign": false,
    "fullName": "John Doe"
}
```

1.6 API reference

Yousign\YousignApi offers the following methods to interact with all domains of the REST API.

Before all examples, you must instanciate the client.

```
use Yousign\YousignApi;

/*
 * API token
 */
$token = '123456789';

/*
 * Production mode
 */
$production = false;

/*
 * Instanciate API wrapper
 */
$yousign = new YousignApi($token, $production);
```

1.6.1 User

getUsers()

Get all users.

Parameters

None

Return Values

Yousign\Model\UserCollection See *Making your first call*

Examples

```
$users = $yousign->getUsers();
```

---

postUser(array $user)

Create an user.

> **Warning:** This implies that each user created by this means (API) or through the application will be billed according to your plan.

Parameters

array $user

Return Values

Yousign\Model\User

Examples

```
$user = $yousign->postUser([
    "firstname" => "John",
    "lastname" => "Doe",
    "email" => "api@yousign.fr",
    "title" => "API teacher",
    "phone" => "+33612345678",
    "organization" => "/organizations/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
]);
```

1.6.2 File

postFile(array $file)

Create a new file as a signable file or an attachment.

Parameters

array See *Advanced mode* Step 2 or *Basic mode* Step 1

Return Values

Yousign\Model\File See *Advanced mode* Step 4

Examples

Create a signable file

```
$file = $yousign->postFile(
    'name'    => 'Name of my signable file.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/filename.pdf'
        )
    ),
    // A procedure must have been created before to link
    // file to it.
    'procedure' => $procedure->getId(),
);
```

Create an attachment file

```
$file = $yousign->postFile([
    'name'    => 'Name of my signable file.pdf',
    'content' => base64_encode(
        file_get_contents(
            '/storage_path/filename.pdf'
        )
    ),
    // A procedure must have been created before to link
    // file to it.
    'procedure' => $procedure->getId(),
    'type'      => 'attachment',
]);

// /files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $file->getId();
```

1.6.3 File objects

postFileObject(array $fileObject)

Create a new signature image in a signable file.

Parameters

array See *Advanced mode* Step 4

Return Values

Yousign\Model\FileObject

Examples

Create a signature image

A file and a member must have been created before (See *Advanced mode* Step 4)

```
$fileObject = $yousign->postFileObject(
    "file"      => $file->getId(),
    "member"    => $member->getId(),
    "position"  => "230,499,464,589",
    "page"      => 2,
    "mention"   => "Read and approved",
    "mention2"  => "Signed By John Doe"
);

// /file_objects/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $fileObject->getId();

// /files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $fileObject->getFile();

// /members/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $fileObject->getMember()->getId();
```

1.6.4 Members

As they are referenced by a procedure, members are created during (*Basic mode*) or after(*Advanced mode*) procedure creation.

For different types of member, see _advanced_features_tutorial.members.

postMember(array $member)

Create a new member.

Parameters

array See *Advanced mode* Step 3

Return Values

Yousign\Model\Member See *Advanced mode* Step 3

Examples

Create a member

```php
$member = $yousign->postMember([
    "firstname"     => "John",
    "lastname"      => "Doe",
    "email"         => "john.doe@yousign.fr",
    "phone"         => "+33612345678",
    "procedure"     => $procedure->getId(),
]);
```

1.6.5 Procedure

postProcedure(array $procedure)

Create a new procedure.

Parameters

array See *Basic mode* Step 2 or *Advanced mode* Step 1

Return Values

Yousign\Model\Procedure See *Basic mode* Step 3

Examples

Create a procedure in advanced mode

For more, see *Advanced mode*.

```php
$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Description of my procedure with advanced mode",
    "start"       => false,
]);

// /procedures/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $procedure->getId();
```

Create a procedure in basic mode

For more, see *Basic mode*.

```php
$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Awesome! Here is the description of my first procedure",
    "members"     => [
        [
            "firstname" => "John",
            "lastname" => "Doe",
            "email" => "john.doe@yousign.fr",
            "phone" => "+33612345678",
```

(continues on next page)

(continued from previous page)

```
            "fileObjects" => [
                [
                    "file" => $file->getId(),
                    "page" => 2,
                    "position" => "230,499,464,589",
                    "mention" => "Read and approved",
                    "mention2" => "Signed by John Doe"
                ]
            ]
        ]
    ]
]);

// /procedures/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $procedure->getId();

foreach ($procedure->getMembers() as $member) {
    // /members/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
    echo $member->getId();
}

foreach ($procedure->getFiles() as $file) {
    // /files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
    echo $file->getId();
}
```

putProcedure(string $id, array $procedure)

Update a procedure. Start a procedure in *Advanced mode*.

Parameters

string A procedure identifier

array See *Advanced mode* Step 5

Return Values

Yousign\Model\Procedure

Examples

Start a procedure in advanced mode

For more, see *Advanced mode*.

```
$procedure = $yousign->putProcedure(
    $procedure->getId(), [
    "start"        => true,
]);
```

(continues on next page)

(continued from previous page)

```
// /procedures/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
echo $procedure->getId();

foreach ($procedure->getMembers() as $member) {
    // /members/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
    echo $member->getId();
}

foreach ($procedure->getFiles() as $file) {
    // /files/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
    echo $file->getId();
}
```

1.6.6 HTTP client options

Sometimes you may need to configure low-level things like HTTP headers, timeout, proxy options.

This Yousign API client uses Guzzle HTTP client under the scene. You can pass an array of options once, before calling API.

After the first call to the API is made, options are immutable.

setClientOptions(array $options)

Parameters

array See http://docs.guzzlephp.org/en/stable/request-options.html

Return Values

Yousign\YousignApi

Examples

Configure a proxy before creating a procedure

For more, see *Advanced mode*.

```
$yousign = new YousignApi($token, $production);

// Set up a proxy
$yousign->setClientOptions([
    'proxy' => 'tcp://localhost:8125',
]);

$procedure = $yousign->postProcedure([
    "name"        => "My first procedure",
    "description" => "Description of my procedure with advanced mode",
    "start"       => false,
]);
```