InstapushPHP Documentation

Release 1.0.0

Aymen FNAYOU

May 26, 2017

Contents

InstapushPHP is full php wrapper for instapush.im APIs that use the wonderful PHP-HTTP.

instapush.im uses a simple REST API to receive messages from your application and send them to devices running mobile clients.

InstapushPHP will make it easy to interact with theses APIs :

- Abstraction of the http client so you can use : Curl, Guzzle 5, Guzzle 6, etc.
- Abstraction of request and response.
- Auto transformation of response into Objects.
- Easy installation process using **composer**
- Respect of PSR recommendations from the PHP-FIG.
- Code quality guaranteed
- And much more.

CHAPTER 1

Installation

The installation process of the InstapushPHP.

With Composer

You can use Composer to install the last version of the library.

```
$ composer require fnayou/instapush-php
```

Now, all you have to todo is to choose and install the **HTTP Client** you want to use.

Note: Note that we implement Discovery feature, so if you already use a http client like **Guzzle** then no more actions are required

Please, take some minutes to check Client Section of the PHP-HTTP documentation.

Use With cURL Client

In order to use cURL Client all you have to do is :

```
$ composer require fnayou/instapush-php
$ composer require php-http/curl-client
```

Use With Guzzle 6 Adapter

In order to use Guzzle 6 Adapter all you have to do is :

```
$ composer require fnayou/instapush-php
$ composer require php-http/guzzle6-adapter
```

CHAPTER 2

Usage

InstapushPHP is easy and simple to use.

Initiate InstapushPHP Client

First of all if you are **already** using a http client like *curl* or *guzzle*, then let Discovery feature make the magic and all you have to do is to initiate InstapushPHP and start requesting:

```
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client
$apiClient = InstapushClient::create('user-token', 'app-id', 'app-secret');

// continue requesting ...
```

if you want to define **http cURL client**

```
use Fnayou\InstapushPHP\InstapushClient;
use Http\Client\Curl\Client as CurlClient;

$curlOptions = [
    CURLOPT_CONNECTTIMEOUT => 10,
    CURLOPT_SSL_VERIFYPEER => false,
];

// initiate cURL client
$curlClient = new CurlClient(null, null, $curlOptions);

// initiate Instapush client
$client = new InstapushClient($curlClient);
$apiClient = $client::create('user-token', 'app-id', 'app-secret');

// continue requesting ...
```

if you want to define **Guzzle client**

```
use Fnayou\InstapushPHP\InstapushClient;
use GuzzleHttp\Client as GuzzleClient;
use Http\Adapter\Guzzle6\Client as GuzzleAdapter;

$config = [
```

```
    'timeout' => 60.0,
    'allow_redirects' => true,
    'verify' => false,
];

// initiate guzzle client
$guzzleClient = new GuzzleClient($config);
$guzzleAdapter = new GuzzleAdapter($guzzleClient);

// initiate Instapush client
$client = new InstapushClient($guzzleAdapter);
$apiClient = $client::create('user-token', 'app-id', 'app-secret');

// continue requesting ...
```

Note: no matter what **http client** you will use, the request of APIs will be the same.

Type of instance

InstapushPHP provide 3 types of instance :

Instance to use Instapush applications API and it require only **UserToken**:

```
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client for User use
$apiClient = InstapushClient::createForUser('user-token');
```

Instance to use Instapush events and notifications APIs and it require **AppId** and **AppSecret**:

```
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client for App use
$apiClient = InstapushClient::createForApp('app-id', 'app-secret');
```

Instance to use full Instapush APIs but for specific user and specific application:

```
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client for both user and app use
$apiClient = InstapushClient::create('user-token', 'app-id', 'app-secret');
```

Note: Please check Instapush REST documentation for more information.

CHAPTER 3

Requesting APIs

InstapushPHP make it easy to request Instapush.im APIs.

Note: Please check Instapush REST documentation for more information about REST APIs.

List of applications

In order to get list of applications for given user, all you have to do

```
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client for User use
$apiClient = InstapushClient::createForUser('user-token');

// list of applications
/** @var \Fnayou\InstapushPHP\Model\Applications $applications */
$applications = $apiClient->applications()->list();
```

The response `$applications` will be instance of `Fnayou\InstapushPHP\Model\Applications` that extend `Doctrine\Common\Collections\ArrayCollection`.

Note: Full description of *Fnayou\InstapushPHP\Model\Applications* model.

Add application

In order to add application for given user, all you have to do

```
use Fnayou\InstapushPHP\InstapushClient;
use Fnayou\InstapushPHP\Model\Application;

// initiate Instapush client for User use
$apiClient = InstapushClient::createForUser('user-token');

// initiate new Application Model
$newApplication = new Application('My New Application');
```

```php
// add the new application
/** @var \Fnayou\InstapushPHP\Model\Application $application */
$application = $apiClient->applications()->add($newApplication);
```

The response `$application` will be instance of `Fnayou\InstapushPHP\Model\Application`.

Note: Full description of *Fnayou\InstapushPHP\Model\Application* model.

List of events

In order to get list of events for given application, all you have to do

```php
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client for App use
$apiClient = InstapushClient::createForApp('app-id', 'app-secret');

// list of events
/** @var \Fnayou\InstapushPHP\Model\Events $events */
$events = $apiClient->events()->list();
```

The response `$events` will be instance of `Fnayou\InstapushPHP\Model\Events` that extend `Doctrine\Common\Collections\ArrayCollection`.

Note: Full description of *Fnayou\InstapushPHP\Model\Events* model.

Add event

In order to add event for given application, all you have to do

```php
use Fnayou\InstapushPHP\InstapushClient;
use Fnayou\InstapushPHP\Model\Event;

// initiate Instapush client for App use
$apiClient = InstapushClient::createForApp('app-id', 'app-secret');

// initiate new Event Model
$newEvent = new Event(
    'event_exemple',
    'the {example_1} is pretty cool, more than the {example_2}',
    ['example_1', 'example_2']
);

// add the new event
/** @var \Fnayou\InstapushPHP\Model\Event $event */
$event = $apiClient->events()->add($newEvent);
```

The response `$event` will be instance of `Fnayou\InstapushPHP\Model\Event`.

Note: Full description of *Fnayou\InstapushPHP\Model\Event* model.

Post notification

In order to post notification for given application, all you have to do

```
use Fnayou\InstapushPHP\InstapushClient;
use Fnayou\InstapushPHP\Model\Notification;

// initiate Instapush client for App use
$apiClient = InstapushClient::createForApp('app-id', 'app-secret');

// initiate new Notification Model
$notification = new Notification(
    'event_exemple',
    [
        'example_1' => 'Guzzle Client',
        'example_2' => 'cURL Client',
    ]
);

// post notification, response will be boolean `true` if success
$response = $apiClient->notification()->post($notification);
```

The response will be `boolean` if success.

Note: Full description of *Fnayou\InstapushPHP\Model\Notification* model.

CHAPTER 4

Handling Exception

InstapushPHP can handle exception/error in 2 different ways.

The method `setHandleException` specify if you want to throw an `ApiException` or to return `ApiError` model:

```
use Fnayou\InstapushPHP\Exception\ApiException;
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client
// handling is set to true by default so it will throw an `ApiException`
$apiClient = InstapushClient::createForUser('user-token');

// if you want to get `ApiError` instead
$apiClient->setHandleException(false);
```

ApiException

`ApiException` extend from `Http\Client\Exception\HttpException` so you can access **Request** and **Response** objects:

```
use Fnayou\InstapushPHP\Exception\ApiException;
use Fnayou\InstapushPHP\InstapushClient;

// initiate Instapush client
$apiClient = InstapushClient::createForUser('user-token');

try {
    /** @var \Fnayou\InstapushPHP\Model\Applications $applications */
    $applications = $apiClient->applications()->list();
} catch (ApiException $apiException) {
    /** @var \Psr\Http\Message\RequestInterface $request */
    $request = $apiException->getRequest();

    /** @var \Psr\Http\Message\ResponseInterface $response */
    $response = $apiException->getResponse();
}
```

ApiError

if handling exception is turned off then you will get and `ApiError` object:

```
use Fnayou\InstapushPHP\InstapushClient;
use Fnayou\InstapushPHP\Model\ApiError;

// initiate Instapush client
$apiClient = InstapushClient::createForUser('user-token');
$apiClient->setHandleException(false);

$applications = $apiClient->applications()->list();

if ($applications instanceof ApiError) {
    /** @var \Fnayou\InstapushPHP\Model\ApiError */
    $apiError = $applications;

    var_dump($apiError->getMessage());
    var_dump($apiError->getStatus());
}
```

Note: Full description of *Fnayou\InstapushPHP\Model\ApiError* model.

CHAPTER 5

Applications

`Applications` object will be used in *List of applications* API.

Note: `Applications` object extend Doctrine\Common\Collections\ArrayCollection.

CHAPTER 6

Application

`Application` object will be used in *Add application* API and *List of applications* API.

Field	Type	Description
title	string	the title of application
appId	string	the application ID
appSecret	string	the application Secret

CHAPTER 7

Events

`Events` object will be used in *List of events* API.

Note: `Events` object extend Doctrine\Common\Collections\ArrayCollection.

CHAPTER 8

Event

`Event` object will be used in *Add event* API and *List of events* API.

Field	Type	Description
title	string	the title of the event
message	string	the push message with trackers
trackers	array	the trackers used within message

CHAPTER 9

Notification

`Notification` object will be used in *Post notification* API.

Field	Type	Description
event	string	the title of the event to use
trackers	array	the trackers and there values

CHAPTER 10

ApiError

`ApiError` will be created and returned only if you set the `handleException` to `false` in your `InstapushClient` instance.

Note: Please check *Handling Exception* documentation.

Field	Type	Description
status	string	the http status of the API request
message	string	the message returnedby the API