

# Programmation WEB avec PHP

Matière : Programmation WEB avec PHP

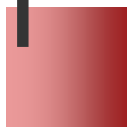
Date : 13.06.2022

Formateur : Malerba Sylvain

Auteur : Malerba Sylvain

# SOMMAIRE

01



Les API

02



cURL

**01**

# **Les API**

# 01

## Les API (API Rest)

- **REpresentational State Transfer Application Program Interface**
- **Permet aux logiciels de communiquer entre eux sur un réseau**

# 01

# Les API (Exemple)

## API OVH

```
$endpoint = 'ovh-eu';
$applicationKey = "your_app_key";
$applicationSecret = "your_app_secret";
$consumer_key = "your_consumer_key";

$conn = new Api(    $applicationKey,
                  $applicationSecret,
                  $endpoint,
                  $consumer_key);

$smsServices = $conn->get('/sms/');
foreach ($smsServices as $smsService) {

    print_r($smsService);
}

$content = (object) array(
    "charset"=> "UTF-8",
    "class"=> "phoneDisplay",
    "coding"=> "7bit",
    "message"=> "Bonjour les SMS OVH par api.ovh.com",
    "noStopClause"=> false,
    "priority"=> "high",
    "receivers"=> [ "+3360000000" ],
    "senderForResponse"=> true,
    "validityPeriod"=> 2880
);
$resultPostJob = $conn->post('/sms/'. $smsServices[0] . '/jobs', $content);

print_r($resultPostJob);
```

# 01

## Les API

- **GET:** permet la récupération de données
- **POST:** permet l'ajout de données
- **PUT:** permet la modification de données
- **DELETE:** permet la suppression de données

<https://editor.swagger.io/> permet documenter une API

# 01

# Les API (Exercice)

**Documenter l'API pour un utilisateur**

## Composer:

- **Gestionnaire de dépendances PHP**
- **Permet d'installer différentes librairies de façon "propre" avec système de versions**





# 01

## Les API

### Installation de slimframework

**composer require slim/slim:4.0.0**

**composer require slim/psr7**



*a micro framework for PHP*

# 01

# Les API

## Dans un fichier index.php

```
<?php
use Psr\Http\Message\ResponseInterface as Response;
use Psr\Http\Message\ServerRequestInterface as Request;
use Slim\Factory\AppFactory;

$loader = require __DIR__ . '/vendor/autoload.php';

$app = AppFactory::create();

$app->get('/', function (Request $request, Response $response, $args)
{
    $response->getBody()->write("Hello world!");
    return $response;
});

$app->run();
```

Hello world!

# 01

# Les API

## Dans un fichier index.php

```
$app = AppFactory::create();

$app->get('/', function (Request $request, Response $response, $args)
{
    $response->getBody()->write("Hello world!");
    return $response;
});

$app->get('/hello/{name}', function (Request $request, Response $
response, array $args) {
    $name = $args['name'];
    $response->getBody()->write("Hello, $name");
    return $response;
});

$app->run();
```

Hello, sylvain

# 01

# Les API (Postman)

## Installer Postman



GET ⌵ http://localhost:8080/user/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

| KEY | VALUE | DESCRIPTION |
|-----|-------|-------------|
| Key | Value | Description |

Body Cookies Headers (5) Test Results 🌐 200 OK 42 ms 270 B S

Pretty Raw Preview Visualize JSON ⌵ ☰

```
1 2
2  "result": "ok",
3  "user": {
4    "id": "1",
5    "login": "toto",
6    "prenom": "a@a.fr"
7  }
8
```

# 01

## Les API (Modification composer.json)

```
{  
    "require": {  
        "slim/slim": "4.0.0",  
        "slim/psr7": "^1.5"  
    },  
    "autoload": {  
        "classmap": [  
            "classes"  
        ]  
    }  
}
```

```
composer dumpautoload -o
```

## 01

# Les API (récupération)

```
$app = AppFactory::create();

$app->get('/user/list[//]', function (Request $request, Response $
    response, array $args) {
    $rows = [];
    $users = Utilisateur::getAllUser();
    foreach ($users as $user) {
        $rows[] = [
            "id" => $user['id'],
            "login" => $user['login'],
            "mail" => $user['mail'],
        ];
    }
    $payload = json_encode($rows, JSON_PRETTY_PRINT);
    $response->getBody()->write($payload);
    return $response->withHeader('Content-Type', 'application/json');
});

$app->run();
```

```
{
  "id": "1",
  "login": "aa",
  "mail": "vvv"
},
{
  "id": "4",
  "login": "bbb",
  "mail": "b@b.fr"
},
{
  "id": "5",
  "login": "ccc"
```

# 01

# Les API (récupération)

```
$app = AppFactory::create();
$app->get('/user/{id}', function (Request $request, Response $response
, array $args) {
    $id = $args['id'];

    $user = new Utilisateur($id);
    if ($user->getId() != 0) {
        $row = [
            "result" => "ok",
            "user" => [
                "id" => $user->getId(),
                "login" => $user->getLogin(),
                "prenom" => $user->getMail(),
            ]
        ];
    } else {
        $row = ["result" => "error", "message" => "User not found"];
    }
    $payload = json_encode($row, JSON_PRETTY_PRINT);
    $response->getBody()->write($payload);
    return $response->withHeader('Content-Type', 'application/json');
});

$app->run();
```

```
{
  "result": "ok",
  "user": {
    "id": "4",
    "login": "bbb",
    "prenom": "b@b.fr"
  }
}
```

## 01

# Les API (ajout de données)

|                                     | KEY   | VALUE  |
|-------------------------------------|-------|--------|
| <input checked="" type="checkbox"/> | login | abc    |
| <input checked="" type="checkbox"/> | mail  | z@z.fr |

```
$app = AppFactory::create();
$app->post('/user[/]', function (Request $request, Response $response,
    array $args) {
    $body = $request->getParsedBody();
    $login = isset($body["login"]) ? $body["login"] : "";
    $mail = isset($body["mail"]) ? $body["mail"] : "";
    try {
        $user = Utilisateur::create($login,$mail);
        $row = [
            "result" => "ok",
        ];
    } catch (Exception $e) {
        $row = [
            "result" => "error",
            "message" => $e->getMessage()
        ];
    }
    $payload = json_encode($row, JSON_PRETTY_PRINT);
    $response->getBody()->write($payload);
    return $response->withHeader('Content-Type', 'application/json');
});
```

```
"result": "ok"
```



## 01

# Les API (modification de données)

```
$app = AppFactory::create();
$app->put('/user/{id}', function (Request $request, Response $response
, array $args) {
    $id = $args['id'];
    $data = json_decode($request->getBody()->getContents(), true);
    try {
        $user = Utilisateur::update($data["login"], $data["mail"], $id);

        $row = [
            "result" => "ok",
        ];
    } catch (Exception $e) {
        $row = [
            "result" => "error",
            "message" => $e->getMessage()
        ];
    }
    $payload = json_encode($row, JSON_PRETTY_PRINT);
    $response->getBody()->write($payload);
    return $response->withHeader('Content-Type', 'application/json');
});

$app->run();
```




# 01

## Les API (suppression)

```
$app = AppFactory::create();
$app->delete('/user/{id}', function (Request $request, Response $
    response, array $args) {
    $id = $args['id'];
    $user = new Utilisateur($id);
    $user->delete();
    $row = array("result"=> "ok");
    $payload = json_encode($row, JSON_PRETTY_PRINT);
    $response->getBody()->write($payload);
    return $response->withHeader('Content-Type', 'application/json');
});

$app->run();
```



```
"result": "ok"
```

02

**cURL**

- **Permet d'appeler une URL en spécifiant plusieurs options :**
  - **méthode appelée (GET, POST, DELETE, PATCH...)**
  - **passage de cookies ou sessions**
  - **spécification de headers**
- **Utile pour les appels d'API**

# 02

## cURL

- Initialisation d'un appel

```
<?php  
$curl = curl_init();  
,
```

- Définition de l'URL d'appel

```
<?php  
$curl = curl_init();  
  
curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/list/');
```

# 02

## cURL

- Demande de retour du résultat

```
<?php
$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/list/');

curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
```

- Exécution de la requête

```
<?php
$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/list/');

curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

$return = curl_exec($curl);

echo $return;
```



- Fermeture de l'appel

```
<?php
$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/list/');

curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

$return = curl_exec($curl);

echo $return;

curl_close($curl);
```

# 02

## cURL (Exemple GET)

```
<?php
$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/list/');

curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

$return = curl_exec($curl);

echo "<pre>";
echo $return;
echo "</pre>";

curl_close($curl);
```

```
[
  {
    "id": "1",
    "login": "aa",
    "mail": "vvv"
  },
  {
    "id": "4",
    "login": "toto",
    "mail": "toto@toto.fr"
  },
  {
    "id": "6",
    "login": "abc",
    "mail": "z@z.fr"
  }
]
```

# 02

## cURL (Exemple POST)

```
$postfields = [  
    "login" => "hermione",  
    "mail" => "hgranger@gmail.com"  
];  
$curl = curl_init();  
curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/');  
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);  
curl_setopt($curl, CURLOPT_POST, true);  
curl_setopt($curl, CURLOPT_POSTFIELDS, $postfields);  
$return = curl_exec($curl);  
echo "<pre>";  
echo $return;  
echo "</pre>";  
curl_close($curl);
```

```
{  
    "result": "ok"  
}
```

# 02

## cURL (Exemple DELETE)

```
$id = 7;  
$curl = curl_init();  
curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/' . $id);  
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);  
curl_setopt($curl, CURLOPT_CUSTOMREQUEST, "DELETE");  
$return = curl_exec($curl);  
echo "<pre>";  
echo $return;  
echo "</pre>";  
curl_close($curl);
```

```
{  
  "result": "ok"  
}
```

# 02

## cURL (Exemple PUT)

```
$id = 8;
$postfields = [
    "login" => "Harry",
    "mail" => "hpotter@gmail.com"
];
$curl = curl_init();
curl_setopt($curl, CURLOPT_URL, 'http://localhost:8080/user/'.$id);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
curl_setopt($curl, CURLOPT_CUSTOMREQUEST, "PUT");
curl_setopt($curl, CURLOPT_POSTFIELDS, json_encode($postfields));
$return = curl_exec($curl);
echo "<pre>";
echo $return;
echo "</pre>";
curl_close($curl);
```

```
{
  "result": "ok"
}
```