FREE eBook

LEARNING slim

#slim

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: slim

It is an unofficial and free slim ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official slim.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with slim

Remarks

This section provides an overview of what slim is, and why a developer might want to use it.

It should also mention any large subjects within slim, and link out to the related topics. Since the Documentation for slim is new, you may need to create initial versions of those related topics.

Versions

Version	Description	Release date
3.8.1	This content related to slim version 3	2017-03-19

Examples

Installation or Setup

Recommended way to install Slim framework is by using composer.

1. **Create an empty directory**

This directory will contain all required files for our Slim application to run. We call this directory **root** directory so we can address all other application files and directories relative to root directory.

```
mkdir slim-app
```

2. **Install Slim framework and its requirements using composer**

```
cd slim-app
composer require slim/slim "^3.0"
```

From now on, we assume this is our working directory.

After composer finishes downloading required files, there should be two files **composer.json** and **composer.lock** and a directory named **vendor** which includes files downloaded by composer. Now we're ready to create our application. To organize our application, we create another directory:

```
mkdir public
```

We call this the **public** directory and we're going to tell our web server to server our application from this directory.

3. **Sample "Hello World" application**

To use Slim create an index.php in public directory with following code:

public/index.php

```
<?php

include "../vendor/autoload.php";

$app = new \Slim\App();

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

$app->run();
```

4. **Start PHP built in server**

We can now use PHP built in server to serve our application:

```
php -S localhost:8080 -t public
```

and run project by opening this address in a web browser:

> http://localhost:8080

Output

> Hello world!

Now configure the webserver so that all requests are passed to this file:

Apache configuration for clean URLs (Optional)

This is not required but recommended for slim projects to remove index.php in API URL.

Create `.htaccess` in the same folder where your `index.php` is located. The file should contain the following code:

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^ index.php [QSA,L]
```

Make sure your Apache virtual host is configured with the `AllowOverride` option so that the `.htaccess` declared rewrite rules can actually be used:

```
AllowOverride All
```

Ngnix configuration

TBA

Getting Started With Slim Framework

Installation or Setup Slim framework

- Install Composer
- Open cmd
- Go to Root directory of your Project Folder and Run Following Command.

  composer require slim/slim "^3.0"

Now you will have vendor directory in your project

Next Create Index.php in root folder and add following code

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

require 'vendor/autoload.php';

$app = new \Slim\App;
$app->get('/hello/{name}', function (Request $request, Response $response) {
    $name = $request->getAttribute('name');
    $response->getBody()->write("Hello, $name");

    return $response;
});
$app->run();
```

Then Run Project on Localhost and try with following command

> http://localhost/project-root/index.php/hello/any-thing

Output

> Hello any-thing

Get Json Data From Database table (REST API)

```
use \Psr\Http\Message\ServerRequestInterface as Request;
use \Psr\Http\Message\ResponseInterface as Response;

require 'vendor/autoload.php';

$app = new \Slim\App;

$app->get('/employee/view', function ($req, $res) {

    $con = new mysqli('localhost','USERNAME','PASSWORD','DATABASE');

    $query = $con->query("SELECT * FROM employee");
    while ($row = $query->fetch_assoc()) {
        $data[] = $row;
    }
    return $res->withJson($data);
});

$app->run();
```

Hello World Example

```
<?php

include "vendor/autoload.php";

$app = new \Slim\App();

$app->get('/hello', function () {
    echo "Hello, world";
});

$app->run();
```

Scaffold skeleton project with `composer` and @akrabats `Slim-Skeleton`

1. Depending on whether you have installed composer globally or locally.
    - Locally: `php composer.phar create-project slim/slim-skeleton {my-app-name}`
    - Globally: `composer create-project slim/slim-skeleton {my-app-name}`
2. If you are running a webserver (ex. Apache or nginx) point your virtualhost document root to your new applications public folder ex. `{my-app-name}/public`.
3. Ensure `logs/` is writeable by your webserver setup.

You can also run the whole shebang in dev mode with:

- Locally: `php composer.phar start`
- Globally: `composer start`

Credit: Rob Allen @akrabat and the SlimPHP team.

PS: I take absolutely NO credit for this. Almost verbatim from slimphp/Slim-Skeleton

Read Getting started with slim online: https://riptutorial.com/slim/topic/6878/getting-started-with-slim

Chapter 2: Return a JSON Response with Slim

Examples

Return a JSON Response with Slim

Slim can use withJson() that can return JSON responses with desired route and method

```
$app = new \Slim\App();
$app->get('/api/person', function ($request, $response, $args) {

$payload=[];
array_push($payload, array("name"=>"Bob"  ,"birth-year"=>1993));
array_push($payload, array("name"=>"Alice","birth-year"=>1995));

 return $response->withJson($payload,200);

});

$app->run();
```

Try it with curl

```
`curl -X GET http://127.0.0.1/api/person`
```

it can return an array in json

```
[{"name":"Bob","birth-year":1993},{"name":"Alice","birth-year":1995}]
```

POST Request With Slim 3 (REST API)

```
use \Psr\Http\Message\ServerRequestInterface as Request;
use \Psr\Http\Message\ResponseInterface as Response;

$app = new \Slim\App;

$app->post('/post/data', function (Request $request, Response $response, $arg){

    $_input = $request->getParsedBody();

    $_data_1 = $_input['name'];
    $_data_2 = $_input['email'];
    $rsp = array();

    if(!empty($_data_1 && !empty($_data_2))){

        $rsp["error"] = false;
        $rsp['message'] = "hello my name is ".$_data_1." and my email is ".$_data_2;
    }else{
        $rsp["error"] = false;
        $rsp['message'] = "you have not posted any data" ;
    }

    return $response
        ->withStatus(201)
        ->withJson($rsp);
});

$app->run();
```

Read Return a JSON Response with Slim online: https://riptutorial.com/slim/topic/9332/return-a-json-response-with-slim

Credits

S. No	Chapters	Contributors
1	Getting started with slim	Community, Dhruvang Gajjar, geggleto, Georgy Ivanov, martisj, Mika Tuupola, Nima
2	Return a JSON Response with Slim	Inovationist Lloyd Idrix, martisj, Ramy hakam, YChi Lu