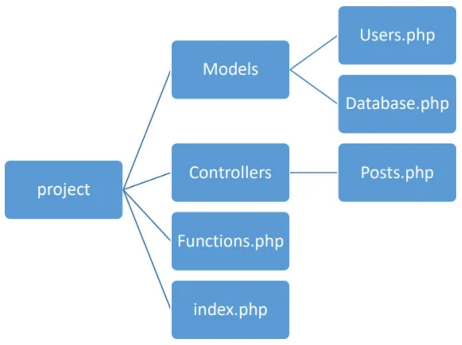
Autoload Multiple Files From Different Directories in PHP

Oyekunle Opeyemi · Follow
3 min read · Dec 16, 2023

12

Let's take a look at the file structure below:



File Structure

This is not a real project, but a sample good enough to explain how to autoload multiple files from different directories. First off, let's start with autoloading files from one directory.

1. **Autoload files from a single directory:** Let's make an assumption that files would be loaded from only the Models directory, we can simply use the PHP built-in autoload function:

spl_autoload_register(?callable $callback = null, bool $throw = true, bool $prepend = false): bool

i. The first parameter is a callback (optional). The default value is null. The default implementation of spl_autoload will be registered if no parameter is provided.

ii. The second parameter is a boolean (optional). The default is true. If the autoload_function cannot be registered, spl_autoload_register() should throw exceptions.

iii. The third parameter is also a boolean (optional). If true, spl_autoload_register() will prepend the autoloader on the autoload stack instead of appending it.

For more explanation about this function, you can follow this link

https://www.php.net/manual/en/function.spl-autoload-register.php

Let's create Database.php and Users.php files in the Model directory, and an index.php file in the root directory:

Database.php

```php
<?php

class Database{
    public function connect() : string {
        return "I am connected to the database.";
    }
}
```

Users.php

```php
<?php

```

```
class Users{
    public function get_user() : array {
        return [
            'id'=> 1,
            'name'=> 'Oyekunle Opeyemi'
        ];
    }
}
```

index.php

```
<?php

spl_autoload_register(function ($class){
   $class_name = substr($class, strrpos ($class, "\\"));
   require_once 'Models/'.ucfirst($class_name).".php";
  }
 );

$Database = new Database();
$Users = new Users();
```

In the index.php file, we have been able to include all the php files in the Model directory by using the spl_autoload_register() function. So, let's look at how we can autoload multiple files from multiple directories.

2. **Autoload files from multiple directories:** This is more of an upgrade to our last example. Let start by creating two more files: Posts.php in the Controllers directory and Functions.php in the root directory:

Posts.php

```
<?php

class Posts{
    public function get_post() : array {
        return [
            'id'=> 1,
            'title'=> 'Autoload Multiples Files From Different Directories in PH
        ];
    }
}
```

Functions.php

```
<?php

class Functions{
    public static function auto_load_files(string $class, array $dirs) : void {
        $class_name = substr($class, strrpos ($class, "\\"));
        foreach ($dirs as $dir) {
            $classes = "$dir/".ucfirst($class_name).".php";
            if (file_exists($classes)) require_once $classes;
        }

    }
}
```

Let's take a look at what we have in the Functions.php file:

a. A static function was created with two parameters that accepts a string and an array respectfully.

b. The $dirs array was looped through to pick each directory in the array to form a path to the class file names.

c. In order not to assign a file to a directory that does not exist, a file_exists() function was used to check the file's existence in the directory before creating a path for it.

We would have to update the index.php file.

index.php

```
<?php
require 'Functions.php';

spl_autoload_register(function ($class){
    $data = [
        "Models",
        "Controllers"
    ];
      Functions::auto_load_files($class, $data);
    }
);

$Database = new Database();
$Users = new Users();
```

```
$Posts= new Posts();
```

a. In the callback, $data is an array of directories, where the class files we want to access are.

b. We called the static function auto_load_files($class, $data) and provided the class registered, and the array of directories.

c. The Posts class from Controllers was added.

I hope this helps someone out there.

Thanks for reading.

PHP Autoload Autoloading

12

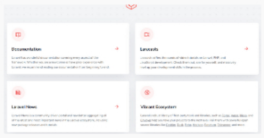
Written by Oyekunle Opeyemi

Follow

21 Followers

An interesting Software Engineer

More from Oyekunle Opeyemi



Oyekunle Opeyemi

Laravel 10: Step by Step on How To Build a Simple Laravel Website...

This article is dedicated to beginners finding it difficult to get started with Laravel. Keep...

19 min read · Aug 9, 2023

3 1

Oyekunle Opeyemi

Saving Multiple Records in Laravel: Query Builder insert() vs Eloquen...

The difference between the Query Builder insert() method and Eloquent create()...

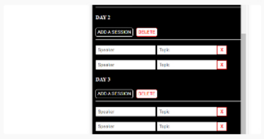
2 min read · Dec 3, 2023

Oyekunle Opeyemi

Foreign Key Constraints in Laravel

If you're new to Laravel, read my article on how to get started in Laravel —...

3 min read · Dec 30, 2023



Oyekunle Opeyemi

CREATING DYNAMIC ELEMENTS

In the web designing world, we use html elements to structure our web pages, create...

4 min read · Dec 3, 2022

3 1

See all from Oyekunle Opeyemi

Recommended from Medium

```
{
    "autoload": {
        "psr-4": {
            "YourVendor\\HelloWorld\\": "src/"
        }
    },
    "prefer-stable": true,
    "minimum-stability": "dev"
}
```

Oleg Marko

Package Development for PHP

Packages are the primary way of adding functionality to Laravel or other PHP apps…

2 min read · Dec 4, 2023

5

Andrea Pollastri

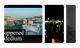
Send Transactional Email from your PHP Applications

Transactional emails are automated emails sent from one sender to one recipient, usuall…

2 min read · Feb 20, 2024

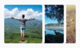
7

Lists


Staff Picks
602 stories · 827 saves

Stories to Help You Level-Up at Work
19 stories · 521 saves

Self-Improvement 101
20 stories · 1495 saves


Productivity 101
20 stories · 1377 saves

```
use omarchouman\LaraUtilX\Utilities\FilteringUtil;

public function filterItems(Request $request)
{
    // Sample data collection (replace this with your actual data)
    $data = collect([
        ['id' => 1, 'name' => 'John', 'age' => 25],
        ['id' => 2, 'name' => 'Jane', 'age' => 30],
        // Add more items as needed
    ]);

    // Get filter parameters from the request
    $name = $request->input('name');
    $operator = $request->input('operator');
    $value = $request->input('value');

    // Use LaraUtilX FilteringUtil to filter the data
    $filteredData = FilteringUtil::filter($data, $name, $operator, $value);

    // Return the filtered data
    return response()->json(['data' => $filteredData->values()]);
```

Omar Chouman

Announcing LaraUtilX: Elevate Your Laravel Development…

I am thrilled to share the exciting news of the release of my latest Laravel package—…

2 min read · Jan 29, 2024

36 1

Adrian Voicu

Enums in PHP 8

Enumerations or "Enums" allow us to define a new type of data with a finite number of…

2 min read · Jan 17, 2024

9

```
public static function stockStatusBadge($status)
    {
        $class = [
            Enum::STOCK_AVAILABLE => 'btn2-secondary',
            Enum::STOCK_ASSIGNED  => 'btn-primary',
            Enum::STOCK_WARRANTY  => 'btn-secondary',
            Enum::STOCK_DAMAGED   => 'btn-warning',
            Enum::STOCK_MISSING   => 'btn-dark',
            Enum::STOCK_EXPIRED   => 'btn-danger',
            Enum::STOCK_RETURN    => 'btn2-secondary',
            Enum::STOCK_OUT       => 'btn2-secondary',
        ];

        return '<span class="badge ' . $class[$status] . '">' .
Enum::getStockStatus($status) . '</span>';
    }
```

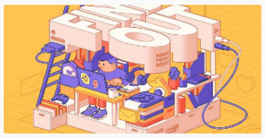
Mohasin Hossain

Using Lookup Table Makes Your Code More Readable, Clean and…

Let's talk about lookup table. Lookup table definition

2 min read · Nov 10, 2023

24 2



Adarsh Rai

Web Development Trends For 2024

The ever-shifting landscape of digital innovation can feel like a relentless race, a…

15 min read · Feb 13, 2024

384 6

See more recommendations