Crear una Clase Helper para un proyecto Laravel

Introducción

Muchas veces necesitamos ciertas funciones o métodos, para nuestros proyectos, que por su repercusión o repetición, puede ser interesante tenerlos agrupados. Solemos llamarlos Helpers.

Podemos hacerlo de tres formas Crear un fichero de funciones (el más común entre los bloggers de Laravel) con carga mediante autload Creación de una Clase estática (no muy común pero más fina y menos propensa a choques) Creación de una paquete, que sería interesante si tuviéramos muchísimos helpers en distintas clases y que pudieran ser compartido por multitud de nuestros proyectos o de otros programadores Aquí vamos a mostrar los dos primeros.

helpers.php

Crearemos un fichero cuyo lugar y nombre podría ser `app/helpers.php` con el contenido de abajo.

Crear un archivo helpers.php

Método incorrecto

No se la de veces que lo habré visto, y me parece horrible y alejado de las buenas prácticas.

```
<?php


function asString($data)
{
   $json = asJSON($data);


   return wordwrap($json, 76, "\n   ");
}
```

```
function asJSON($data)
{
   $json = json_encode($data);
   $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

   return $json;
}
```

Método correcto

Ya que no usamos una clase sino un archivo tipo include que cargaremos mediante un autload, para evitar problemas de duplicidad de nombres con las funciones de PHP, lo correcto es hacerlo como el código de abajo

```
<?php

if (!function_exists('asString'))
{
    function asString($data)
    {
        $json = asJSON($data);

        return wordwrap($json, 76, "\n   ");
    }
}

if (!function_exists('asJson'))
{
    function asJSON($data)
    {
        $json = json_encode($data);
        $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

        return $json;
    }
}
```

Carga del archivo

Editamos nuestro composer.json en la sección `autoload` añadiendo o creando la sección `files`

```
"autoload": {
     "psr-4": {
         "App\\": "app/",
         "Database\\Factories\\": "database/factories/",
         "Database\\Seeders\\": "database/seeders/"
     },
     "files": [
         "app/helpers.php
     ]
},
```

Actualización del autoload de la app

```
composer dump-autoload
```

Uso

El uso es sencillo, ya que se le llama como si fuera una función nativa de PHP.

```
$headerData = [
   'category' => 'develop',
   'unique_args' => [
       'var_1' => 'abc'
   ]
];

$header = asString($headerData);
```

Crear una clase estática

La creación de una clase estática, nos permite más seguridad, algo más de estandarización y para trabajar en grupo, y el camino a la creación de nuestro propio paquete de Helpers.

Crear el fichero de clase Helper

En mi ejemplo uso el directorio Helpers dentro de App porque tengo más clases de helpers en un proyecto largo `app/Helpers/MailHelpers`

```php
<?php

namespace App\Helpers;

class MailHelpers {
    public static function asString($data)
    {
        $json = self::asJson($data);

        return wordwrap($json, 76, "\n   ");
    }

    public static function asJson ($data)
    {
        $json = json_encode($data);
        $json = preg_replace('/(["\]}])([,:])(["\[{])/', '$1$2 $3', $json);

        return $json;
    }
}
```

De esta forma no necesitamos realizar ninguna modificación en nuestro fichero composer.json, ya que la carga se produce conforme al PSR-4, y es simplemente una clase más de tipo estático.

Uso

Este ejemplo es del uso de la clase MailHelpers en una clase Mail.

```php
<?php

namespace App\Mail;

use App\Helpers\MailHelpers;
…

public function build()
{
  $headerData = [
      'category' => 'develop',
```

```
    'unique_args' => [
        'var_1' => 'abc'
    ]
];

$header = MailHelpers::asString($headerData);
...
...
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Revision #1
Created 2021-09-09 08:29:23 UTC by Abkrim
Updated 2021-09-09 08:30:34 UTC by Abkrim