slimphp / Twig-View

Code Issues 3 Pull requests 3 Actions Projects Wiki Security Insights

Slim Framework view helper built on top of the Twig templating component

slimframework.com

MIT license

359 stars 87 forks 25 watching Branches Activity Custom properties

Tags

Public repository

3.x 5 Branches 21 Tags Go to file t Go to file Add file Code

dfranco and Davide Franco Fix Twig security issue (#328) b4268d8 · 2 months ago

.github	Update build (#324)	4 months ago
src	Update build (#324)	4 months ago
tests	Update build (#324)	4 months ago
.coveralls.yml	fix Travis-CI config and add Coveralls config	5 years ago
.editorconfig	Update build (#324)	4 months ago
.gitattributes	Update to PSR-12	3 years ago
.gitignore	Update build (#324)	4 months ago
CONTRIBUTING.md	update CONTRIBUTING.md	2 years ago
LICENSE.md	First	9 years ago
README.md	Update usage with DI container (#323)	4 months ago
composer.json	Fix Twig security issue (#328)	2 months ago
phpcs.xml	Update build (#324)	4 months ago
phpstan.neon	Update build (#324)	4 months ago
phpunit.xml	Update build (#324)	4 months ago

Slim Framework Twig View

release v3.4.1 license MIT tests passing coverage 100% downloads 5.4M

This is a Slim Framework view helper built on top of the Twig templating component. You can use this component to create and render templates in your Slim Framework application.

Install

Via Composer

```
composer require slim/twig-view
```

Requires Slim Framework 4, Twig 3 and PHP 7.4 or newer.

Usage

With DI Container

```php
use DI\Container;
use Slim\Factory\AppFactory;
use Slim\Views\Twig;
use Slim\Views\TwigMiddleware;

require __DIR__ . '/../vendor/autoload.php';

// Create Container
$container = new Container();

// Set view in Container
$container->set(Twig::class, function() {
    return Twig::create(__DIR__ . '/../templates', ['cache' => 'path/to/cache']);
});

// Create App from container
$app = AppFactory::createFromContainer($container);

// Add Twig-View Middleware
$app->add(TwigMiddleware::create($app, $container->get(Twig::class)));

// Add other middleware
$app->addRoutingMiddleware();
$app->addErrorMiddleware(true, true, true);

// Render from template file templates/profile.html.twig
$app->get('/hello/{name}', function ($request, $response, $args) {
    $viewData = [
        'name' => $args['name'],
    ];

    $twig = $this->get(Twig::class);
```

README MIT license

```php
// Render from string
$app->get('/hi/{name}', function ($request, $response, $args) {
    $viewData = [
        'name' => $args['name'],
    ];

    $twig = $this->get(Twig::class);
    $str = $twig->fetchFromString('<p>Hi, my name is {{ name }}.</p>', $viewData);
    $response->getBody()->write($str);

    return $response;
});

// Run app
$app->run();
```

Without DI container

```php
use Slim\Factory\AppFactory;
use Slim\Views\Twig;
use Slim\Views\TwigMiddleware;

require __DIR__ . '/../vendor/autoload.php';

// Create App
$app = AppFactory::create();

// Create Twig
$twig = Twig::create('path/to/templates', ['cache' => 'path/to/cache']);

// Add Twig-View Middleware
$app->add(TwigMiddleware::create($app, $twig));

// Define named route
$app->get('/hello/{name}', function ($request, $response, $args) {
```

```
    $view = Twig::fromRequest($request);
    return $view->render($response, 'profile.html.twig', [
        'name' => $args['name']
    ]);
})->setName('profile');

// Render from string
$app->get('/hi/{name}', function ($request, $response, $args) {
    $view = Twig::fromRequest($request);
    $str = $view->fetchFromString(
        '<p>Hi, my name is {{ name }}.</p>',
        [
            'name' => $args['name']
        ]
    );
    $response->getBody()->write($str);

    return $response;
});

// Run app
$app->run();
```

Custom template functions

`TwigExtension` provides these functions to your Twig templates:

- `url_for()` - returns the URL for a given route. e.g.: /hello/world
- `full_url_for()` - returns the URL for a given route. e.g.: https://www.example.com/hello/world
- `is_current_url()` - returns true is the provided route name and parameters are valid for the current path.
- `current_url()` - returns the current path, with or without the query string.
- `get_uri()` - returns the `UriInterface` object from the incoming `ServerRequestInterface` object
- `base_path()` - returns the base path.

You can use `url_for` to generate complete URLs to any Slim application named route and use `is_current_url` to determine if you need to mark a link as active as shown in this example Twig template:

```
<h1>User List</h1>
<ul>
    <li><a href="{{ url_for('profile', { 'name': 'josh' }) }}" {% if is_current_url('profile', { 'name': 'josh' }) %}class="act
    <li><a href="{{ url_for('profile', { 'name': 'andrew' }) }}">Andrew</a></li>
</ul>
```

Tests

To execute the test suite, you'll need to clone the repository and install the dependencies.

```
$ git clone https://github.com/slimphp/Twig-View
$ composer install
$ composer test
```

Contributing

Please see CONTRIBUTING for details.

Security

If you discover any security related issues, please email security@slimframework.com instead of using the issue tracker.

Credits

- Josh Lockhart
- Pierre Bérubé

License

Releases 17

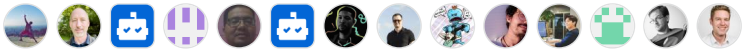
3.4.1 Latest
on Sep 26

+ 16 releases

Packages

No packages published

Contributors 43



+ 29 contributors

Languages

PHP 99.5% HTML 0.5%