Бесплатная электронная книга

УЧУСЬ zend-framework2

Free unaffiliated eBook created from
Stack Overflow contributors.

#zend-framework2

Около

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: zend-framework2

It is an unofficial and free zend-framework2 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 zend-framework2.

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

глава 1: Начало работы с zend-framework2

замечания

Zend Framework 2 (ZF2) - это современная и гибкая инфраструктура PHP, которая помогает веб-разработчикам создавать веб-приложения различной сложности. Основным спонсором компании Zend Framework является Zend Technologies , что делает ее очень сильной и стабильной. Есть два основных улучшения этой второй версии над ZF1. Во-первых, модульная архитектура была принята по умолчанию без каких-либо настроек. Это пригодится при разработке крупногабаритного веб-приложения, которое требует разложения модулей. Во-вторых, ZF2 реализует все функции, которые PHP5.3 + может предложить, в частности, пространства имен. В предыдущих версиях класс контроллера назван следующим образом:

```
class IndexController extends Zend_Controller_Action
{

}
```

Этот же класс переписан в ZF2 следующим образом:

```
namespace Application\Controller;
use Zend\Mvc\Controller\AbstractActionController;

class IndexController extends AbstractActionController
{

}
```

Ниже приведены некоторые другие интересные функции ZF2:

- Внедрение зависимости
- Менеджер по корпоративным мероприятиям
- Сервис-менеджер

Examples

Установка или настройка

Подробные инструкции по установке или установке Zend Framework 2. Существуют различные способы установки фреймворка. Ниже приведены некоторые из них:

Использование композитора - рекомендуемый способ

Предполагая, что `composer` установлен в целевом поле.

Чтобы установить скелетное MVC-приложение, запустите в своем терминале для создания нового проекта 2-го проекта zend в указанном месте:

```
php composer.phar create-project -sdev \
    --repository-url="https://packages.zendframework.com" \
    zendframework/skeleton-application path/to/install
```

чтобы вручную установить **минимальный** ZF2 (Zend MVC + его несколько зависимостей), запустите в командной строке:

```
composer require zendframework/zend-mvc
```

или для **полноценного** d ZF2 (+64 модулей):

```
composer require zendframework/zendframework`
```

Обратите внимание, что первый вариант запускает установщик, который предоставит вам полностью функциональное приложение вместе с обычной структурой каталогов приложений. Другие варианты позволят вам создать целое приложение с нуля, поскольку он просто предоставляет модули ZF2 для разработки.

Использование субмодулей Git

Выполните приведенную ниже команду для клонирования zf2 и ее рекурсивно рекурсивно из Github:

```
git clone git://github.com/zendframework/ZendSkeletonApplication.git --recursive
```

Настройка HTTP-сервера

Для типичного веб-приложения требуется запуск службы HTTP, прослушивающей выделенный порт (обычно: 80) для передачи входящих запросов в приложение, обработки и подачи вывода (ответа) обратно.

> Примечание. Вы также можете записывать приложения с поддержкой консоли в Zend Framework 2 без необходимости HTTP-сервера.

ВАРИАНТ 1 - PHP CLI Server

Самый простой способ начать работу, если вы используете PHP 5.4 или выше, - это запустить внутренний PHP-cli-сервер в корневом каталоге.

Перейдите в каталог проекта и запустите:

```
php -S 0.0.0.0:8080 -t public/ public/index.php`.
```

Это запустит встроенный cli-сервер на порту `8080` и привяжет его ко всем сетевым интерфейсам.

ВАРИАНТ 2 - Пользовательский HTTP-сервер

Настройте виртуальный хост на Apache *или* Microsoft IIS Server *или* Nginx и передайте входящие HTTP-запросы в приложение.

Простой Hello World

В командной строке `composer create-project zendframework/skeleton-application helloWorldTest` в каталог, в который вы хотите создать проект, затем введите: `composer create-project zendframework/skeleton-application helloWorldTest` . Во время установки вас спросят, хотите ли вы минимальную установку: скажем да, на данный момент мы просто тестируем.

Для простоты мы будем использовать встроенный сервер CLI на PHP. Из командной строки `php -S 0.0.0.0:8080 -t public/ public/index.php` в корневую директорию вашего проекта (`helloWorldTest`), затем запустите: `php -S 0.0.0.0:8080 -t public/ public/index.php` . Теперь откройте свой веб-браузер и перейдите по адресу http: // localhost / , вы увидите страницу приветствия приложения Skeleton ZF2.

Если вы это сделаете, мы теперь настроим новую страницу. В `module/Application/config/module.config.php` вы можете увидеть, что динамический маршрут уже настроен для подпапки приложения:

```
return [
    'router' => [
        'routes' => [
            'home' => [
                ...
            ],
            'application' => [
                'type'    => Segment::class,
                'options' => [
                    'route'    => '/application[/:action]',
                    'defaults' => [
                        'controller'    => Controller\IndexController::class,
                        'action'        => 'index',
                    ],
                ],
            ],
        ],
    ],
```

Установите новое действие « `helloWorldAction()` » в

`module/Applicaiton/src/Controller/IndexController.php` :

```
class IndexController extends AbstractActionController
{
    public function indexAction()
    {
        ...
    }

    public function helloWorldAction()
    {
        return new ViewModel();
    }
}
```

Наконец, создайте файловый `module/Application/view/application/index/hello-world.phtml` со следующим содержимым:

```
<?php
echo "Hello World !";
```

Теперь перейдите по адресу http: // localhost / application / hello-world и скажите привет ZF2!

Как создать завод

Когда класс должен быть обеспечен жесткими зависимостями, лучшей практикой является использование шаблона инъекции конструктора, в котором эти зависимости вводятся с использованием фабрики.

Предположим, что `MyClass` сильно зависит от зависимости `$dependency` которую нужно решить из конфигурации приложения.

```
<?php
namespace Application\Folder;

use Zend\ServiceManager\FactoryInterface;
use Zend\ServiceManager\ServiceLocatorInterface;

class MyClass
{
    protected $dependency;

    public function __construct($dependency)
    {
        $this->dependency = $dependency;
    }
}
```

Чтобы ввести эту зависимость, создается фабричный класс. Эта фабрика разрешит зависимость от конфигурации и добавит значение конфигурации при построении класса и вернет результат:

```php
<?php
namespace Application\Factory;

use Zend\ServiceManager\FactoryInterface;
use Zend\ServiceManager\ServiceLocatorInterface;

class MyClassFactory implements FactoryInterface
{
    public function createService(ServiceLocatorInterface $serviceLocator)
    {
        $config = $servicelocator->get('Config');
        $dependency = $config['dependency'];
        $myClass = new MyClass($dependency);
        return $myClass;
    }
}
```

Теперь, когда фабричный класс был создан, он должен быть зарегистрирован в конфигурации менеджера сервиса в файле конфигурации модуля `module.config.php` под ключевыми фабриками. Рекомендуется использовать одинаковые имена для класса и фабрики, поэтому их легко найти в дереве папок проекта:

```php
<?php

namespace Application;

return array(
    //...
    'service_manager' => [
        'factories' => [
            'Application\Folder\MyClass' => 'Application\Factory\MyClassFactory'
        ]
    ],
    //...
);
```

В качестве альтернативы константы имени класса могут использоваться для их регистрации:

```php
<?php

namespace Application;

use Application\Folder\MyClass;
use Application\Factory\MyClassFactory;

return array(
    //...
    'service_manager' => [
        'factories' => [
            MyClass::class => MyClassFactory::class'
        ]
    ],
    //...
);
```

Теперь класс можно собрать в диспетчере сервисов с помощью ключа, который мы использовали при регистрации фабрики для этого класса:

```
$serviceManager->get('Application\Folder\MyClass');
```

или же

```
$serviceManager->get(MyClass::class);
```

Менеджер сервиса найдет, соберет и запустит завод, а затем вернет экземпляр класса с введенной зависимостью.

Прочитайте Начало работы с zend-framework2 онлайн: https://riptutorial.com/ru/zend-framework2/topic/1304/начало-работы-с-zend-framework2

глава 2: Введение в Zend Expressive

Examples

Простой Hello World

Используя композитор, выполните следующую команду в каталоге, в котором будет установлено приложение: `composer create-project zendframework/zend-expressive-skeleton expressive-skeleton` .

Во время процесса установки вас попросят принять различные решения.

1. Для вопроса установки по умолчанию, скажем, no (`n`);
2. Для маршрутизатора, давайте использовать Aura Router (# `1`);
3. Для контейнера используйте Zend ServiceManager (# `3`);
4. Для шаблона давайте использовать Zend View (# `3`);
5. Наконец, для обработчика ошибок, давайте использовать Whoops (# `1`).

После установки войдите в корневой каталог, `expressive-skeleton` , запустите встроенный сервер CLI на PHP: `php -S 0.0.0.0:8080 -t public public/index.php` . Перейдите в http: // localhost: 8080 / с вашим браузером, теперь ваше приложение должно быть запущено.

Давайте настроим новый путь к новому промежуточному программному обеспечению. Сначала откройте файл `config/autoload/routes.global.php` маршрутизатора в `config/autoload/routes.global.php` и добавьте следующие строки:

```
<?php

return [
    'dependencies' => [
        ...
    ],

    'routes' => [
        [
            'dependencies' => [
                'invokables' => [
                    ...
                ],
                'factories' => [
                    ...
                    // Add the following line
                    App\Action\HelloWorldAction::class => App\Action\HelloWorldFactory::class,
                ],
            ],
        ],
        // Following lines should be added
        [
            'name' => 'hello-world',
            'path' => '/hello-world',
```

```
            'middleware' => App\Action\HelloWorldAction::class,
            'allowed_methods' => ['GET'],
        ],
    ],
];
```

Поместите следующий контент в `src/App/Action/HelloWorldFactory.php` :

```
<?php

namespace App\Action;

use Interop\Container\ContainerInterface;
use Zend\Expressive\Template\TemplateRendererInterface;

class HelloWorldFactory
{
    public function __invoke(ContainerInterface $container)
    {
        $template = ($container->has(TemplateRendererInterface::class))
            ? $container->get(TemplateRendererInterface::class)
            : null;

        return new HelloWorldAction($template);
    }
}
```

Затем этот контент в `src/App/Action/HelloWorldAction.php` :

```
<?php

namespace App\Action;

use Psr\Http\Message\ResponseInterface;
use Psr\Http\Message\ServerRequestInterface;
use Zend\Diactoros\Response\HtmlResponse;
use Zend\Diactoros\Response\JsonResponse;
use Zend\Expressive\Template;
use Zend\Expressive\Plates\PlatesRenderer;
use Zend\Expressive\Twig\TwigRenderer;
use Zend\Expressive\ZendView\ZendViewRenderer;

class HelloWorldAction
{
    private $template;

    public function __construct(Template\TemplateRendererInterface $template = null)
    {
        $this->template = $template;
    }

    public function __invoke(ServerRequestInterface $request, ResponseInterface $response,
callable $next = null)
    {
        $data = [];

        return new HtmlResponse($this->template->render('app::hello-world'));
    }
```

```
}
```

Затем, наконец, просто поставьте следующее в `templates/app/hello-world.phtml` :

```
<?php echo 'Hello World'; ?>
```

Мы сделали ! Перейдите к http: // localhost: 8080 / hello-world и скажите «привет» Zend Expressive!

Прочитайте Введение в Zend Expressive онлайн: https://riptutorial.com/ru/zend-framework2/topic/6109/введение-в-zend-expressive

глава 3: Монтаж

Examples

установка через композитор (github)

```
1 cd my/project/dir
2 git clone git://github.com/zendframework/ZendSkeletonApplication.git
3 cd ZendSkeletonApplication
4 php composer.phar self-update
5 php composer.phar install
```

Прочитайте Монтаж онлайн: https://riptutorial.com/ru/zend-framework2/topic/6458/монтаж

кредиты

S. No	Главы	Contributors
1	Начало работы с zend-framework2	Community, edigu, Hassan, Sanjeev kumar, Shirraz, Wilt
2	Введение в Zend Expressive	Shirraz
3	Монтаж	edigu, Waqar Haider