

Создание первого приложения с помощью Zend Framework

2

В этой главе мы разработаем наш первый проект в Zend Framework 2.0. Мы рассмотрим ключевые аспекты разработки приложения с помощью Zend Framework 2.0, создавая модули, контроллеры и представления. Мы создадим в Zend Framework собственный модуль, который будем расширять в следующих главах этой книги.

Подготовка

Перед тем как приступить к созданию первого проекта в Zend Framework 2.0, удостоверьтесь, что в вашей среде разработки установлены и сконфигурированы следующие программы:

- ❑ командно-строковый интерфейс PHP;
- ❑ **Git** — программа необходима для работы с исходными кодами из различных репозитариев github.com;
- ❑ **Composer** — инструмент управления PHP-зависимостями.

Следующие команды пригодятся для установки инструментов, необходимых для настройки проекта Zend Framework 2.0.

- ❑ Для установки командно-строкового интерфейса PHP:

```
$ sudo apt-get install php5-cli
```

- ❑ Для установки системы Git:

```
$ sudo apt-get install git
```

- ❑ Для установки программы Composer:

```
$ curl -s https://getcomposer.org/installer | php
```

Приложение ZendSkeletonApplication

Проект `ZendSkeletonApplication` представляет собой приложение-заготовку, с помощью которого разработчики могут начать освоение Zend Framework 2.0. В приложении-заготовке используется поддерживаемый Zend Framework 2 паттерн модель-представление-контроллер (Model-View-Controller, MVC), в том числе новая модульная система. Приложение `ZendSkeletonApplication` можно загрузить с ресурса GitHub (<https://github.com/zendframework/ZendSkeletonApplication>).

Время действовать — создание проекта Zend Framework

Чтобы настроить новый проект Zend Framework, нам потребуется загрузить последнюю версию приложения `ZendSkeletonApplication` и настроить виртуальный хост так, чтобы он указывал на новый проект Zend Framework. Выполните следующие шаги.

1. Перейдите в папку, в которой вы хотите разместить проект Zend Framework:

```
$ cd /var/www/
```

2. Выполните клонирование проекта `ZendSkeletonApplication` из GitHub:

```
$ git clone git://github.com/zendframework/  
ZendSkeletonApplication.git CommunicationApp
```



СОВЕТ

В некоторых конфигурациях Linux текущий пользователь может не обладать правами доступа, требуемыми для записи в каталог `/var/www`. В таких случаях можно использовать любой каталог, доступный для записи, и вносить необходимые изменения в конфигурацию виртуального хоста.

3. Установите зависимости с помощью программы Composer:

```
$ cd CommunicationApp/
$ php composer.phar self-update
$ php composer.phar install
```

На следующем рисунке показано, как программа Composer загружает и устанавливает необходимые зависимости.

```
krishnav@ubuntu:/var/www/CommunicationApp$ php composer.phar self-update
Updating to version 172414a.
  Downloading: 100%
krishnav@ubuntu:/var/www/CommunicationApp$ php composer.phar install
Loading composer repositories with package information
Installing dependencies
  - Installing zendframework/zendframework (2.0.3)
    Downloading: 100%

zendframework/zendframework suggests installing doctrine/common (Doctrine\Common >=2
zendframework/zendframework suggests installing ext-intl (ext/intl for i18n features)
zendframework/zendframework suggests installing pecl-weakref (Implementation of weak
zendframework/zendframework suggests installing zendframework/zendpdf (ZendPdf for c
zendframework/zendframework suggests installing zendframework/zendservice-recaptcha
as in Zend\Captcha and/or Zend\Form)
Writing lock file
Generating autoload files
krishnav@ubuntu:/var/www/CommunicationApp$
```

4. Перед тем как создать запись для виртуального хоста, мы должны создать запись для имени хоста в файле хостов, чтобы система всегда указывала на локальный компьютер при использовании нового имени хоста. В операционной системе Linux это можно сделать, добавив запись в файл `/etc/hosts`:

```
$ sudo vim /etc/hosts
```

ПРИМЕЧАНИЕ

В операционной системе Windows этот файл находится в папке `%SystemRoot%\system32\drivers\etc\hosts`.

5. Добавьте следующую строку в файл `hosts`:

```
127.0.0.1 comm-app.local
```

В результате файл `hosts` должен выглядеть так.

```
127.0.0.1      localhost
127.0.1.1      ubuntu
127.0.0.1      comm-app.local

# The following lines are desirable for IPv6 capable hosts
::1           ip6-localhost ip6-loopback
fe00::0       ip6-localnet
ff00::0       ip6-mcastprefix
ff02::1       ip6-allnodes
ff02::2       ip6-allrouters
```

6. На следующем шаге добавим запись виртуального хоста на наш веб-сервер; это можно сделать, создав конфигурационный файл нового виртуального хоста:

```
sudo vim /usr/local/zend/etc/sites.d/vhost_comm-app-80.conf
```

ПРИМЕЧАНИЕ

В вашем случае имя нового виртуального хоста может быть иным в зависимости от того, какой веб-сервер вы используете. Чтобы настроить новый виртуальный хост, обратитесь к документации вашего веб-сервера.

Например, если вы работаете с веб-сервером Apache2 в операционной системе Linux, то вам нужно создать файл нового виртуального хоста в каталоге `/etc/apache2/sites-available` и включить этот сайт командой `a2ensite comm-app.local`.

7. Добавьте следующую конфигурацию в файл виртуального хоста:

```
<VirtualHost *:80>
  ServerName comm-app.local
  DocumentRoot /var/www/CommunicationApp/public
  SetEnv APPLICATION_ENV "development"
  <Directory /var/www/CommunicationApp/public>
    DirectoryIndex index.php
    AllowOverride All
    Order allow,deny
    Allow from all
  </Directory>
</VirtualHost>
```

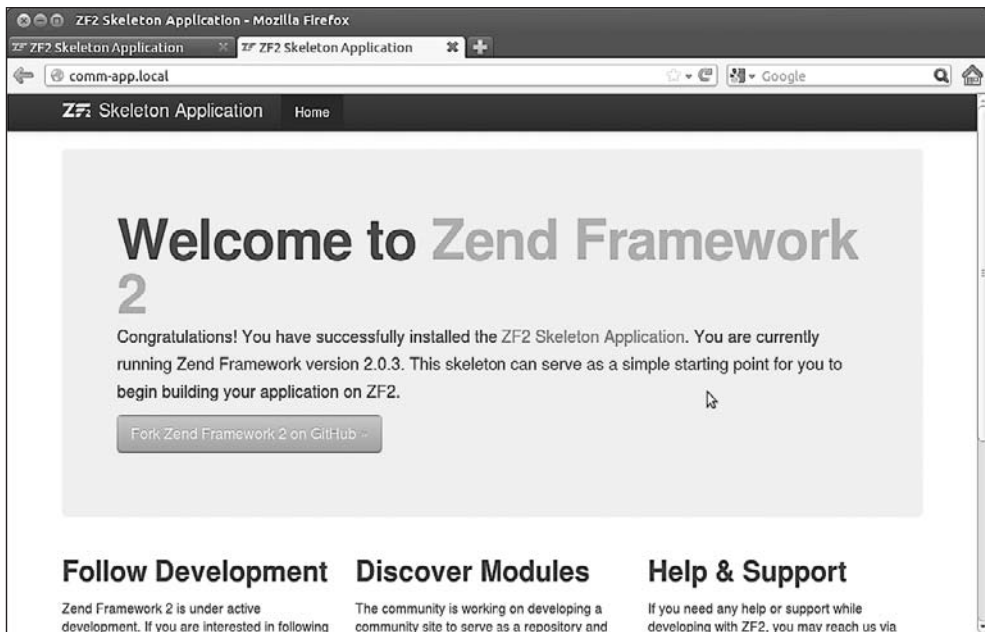
ПРИМЕЧАНИЕ

Если вы используете другой путь для работы с проектом `ZendSkeletonApplication`, удостоверьтесь в том, что этот путь используется как в директиве `DocumentRoot`, так и в директиве `Directory`.

8. После конфигурирования файла виртуального хоста необходимо перезапустить веб-сервер командой

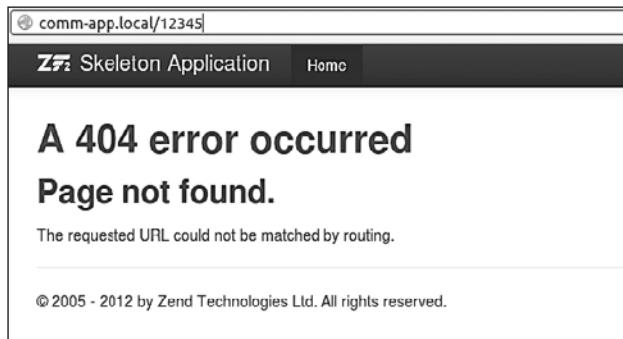
```
$ sudo service zend-server restart
```

9. После завершения установки у вас должна открываться в браузере ссылка <http://comm-app.local>, ведущая к следующей тестовой странице.



ПРОВЕРЯЙТЕ ПРАВИЛА ПЕРЕЗАПИСИ

Иногда модуль `mod_rewrite` по умолчанию может не подключаться в веб-браузере. Чтобы проверить корректность работы механизма перенаправления URL-адресов, попробуйте перейти по неправильному URL-адресу, например `http://comm-app.local/12345`. Если вы получите от Apache «ошибку 404», то правила перезаписи `.htaccess` не работают — их нужно исправить, в противном случае, получая страницу наподобие представленной на следующем рисунке, вы можете быть уверены, что URL-адрес работает должным образом.



Что сейчас произошло?

Мы успешно создали новый проект Zend Framework 2, воспользовавшись приложением ZendSkeletonApplication из ресурса GitHub, и загрузили с помощью программы Composer необходимые зависимости, в том числе для Zend Framework 2.0. Мы также создали конфигурацию виртуального хоста, указывающую на папку public проекта, и протестировали проект в веб-браузере.

ЗАГРУЗКА ПРИМЕРОВ КОДА

Все файлы с примерами кода для книг издательства Packt, приобретенных с помощью учетной записи на <http://www.packtpub.com>, доступны для загрузки. Если вы приобрели эту книгу в другом месте, то вы можете зарегистрироваться на сайте <http://www.packtpub.com/support> и получить файлы напрямую по электронной почте.

АЛЬТЕРНАТИВНЫЕ СПОСОБЫ УСТАНОВКИ

Мы рассмотрели лишь один из способов установки приложения ZendSkeletonApplication, но есть и другие.

Вы можете напрямую загрузить приложение-заготовку с помощью программы Composer и создать проект командой

```
$ php composer.phar create-project --repository-url="http://packages.zendframework.com" zendframework/skeleton-application путь/для/установки
```

Можно также воспользоваться рекурсивным клонированием программы Git для создания такого же проекта:

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

С дополнительной информацией можно ознакомиться в документе <http://framework.zend.com/downloads/skeleton-app>.

Модули Zend Framework 2.0

В Zend Framework 2.0 модуль можно определить как единицу программного обеспечения, которую можно переносить, многократно использовать и связывать с другими модулями для создания более крупных и сложных приложений.

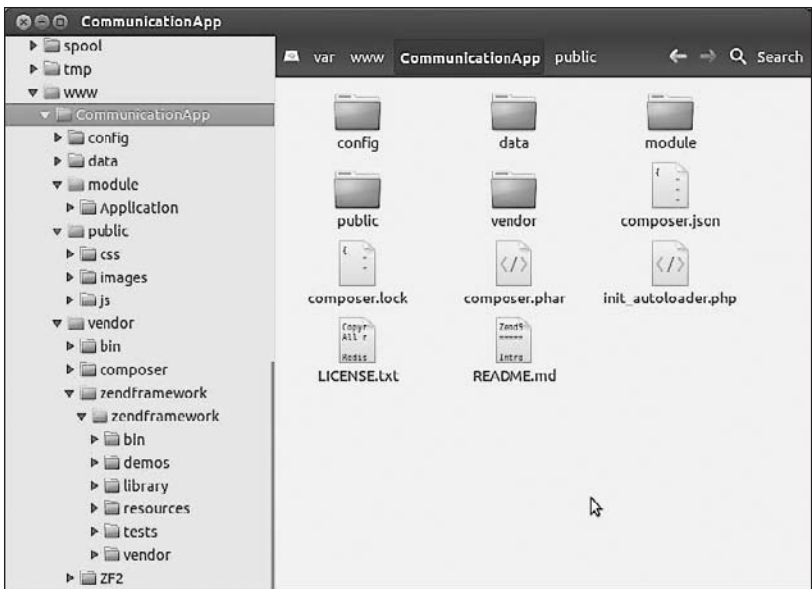
Модули — не новая концепция для Zend Framework, однако в Zend Framework 2 модули применяются совершенно особым образом. В Zend Framework 2 модули могут совместно использоваться несколькими системами, а действия по их переупаковке и распространению относительно просты. Еще одно существенное нововведение в Zend Framework 2 заключается в том, что теперь даже главное приложение преобразуется в модуль — модуль приложения.

Перечислим некоторые ключевые преимущества модулей Zend Framework 2.0:

- ❑ автономность, переносимость, возможность многократного использования;
- ❑ управление зависимостями;
- ❑ легковесность и быстродействие;
- ❑ поддержка упаковки с помощью программы Phar и распространения с помощью инструмента Pkgus.

Структура папок проекта Zend Framework

Структура папок проекта Zend Framework 2 показана на рисунке.



Имя папки	Описание
config	Используется для управления конфигурацией приложения
data	Используется как временное хранилище для данных приложения, в том числе файлов кэша, файлов сессий, журналов и индексов
module	Служит для управления всем кодом приложения
module/Application	Модуль приложения, предлагаемый по умолчанию; входит в состав ZendSkeletonApplication

Имя папки	Описание
public	Служит точкой входа в приложение и корневой папкой для документов веб-сайта. В ней хранятся все веб-ресурсы, в том числе файлы каскадных таблиц стилей, изображения и Java-сценарии
vendor	Применяется для управления общими библиотеками, которые использует приложение. Zend Framework также устанавливается в эту папку
vendor/zendframework	Zend Framework 2.0 устанавливается в эту папку

Время действовать — создание модуля

Цель наших следующих действий — создать новый модуль **Users** в Zend Framework 2.0. Этот модуль будет управлять пользователями, в том числе осуществлять их регистрацию, аутентификацию и т. д. Мы задействуем модуль **ZendSkeletonModule** из состава Zend Framework.

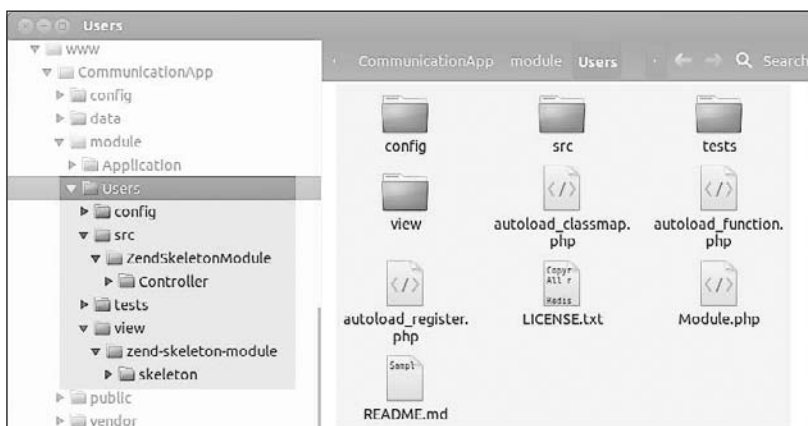
1. Перейдите в папку **module** приложения:

```
$ cd /var/www/CommunicationApp/
$ cd module/
```

2. Выполните клонирование модуля **ZendSkeletonModule**, назвав клон как сочтете нужным, в данном случае — **Users**:

```
$ git clone git://github.com/zendframework/ZendSkeletonModule.git Users
```

3. По окончании работы структура папок должна выглядеть следующим образом.



4. Отредактируйте файл `Module.php`, который будет расположен в папке `Users` раздела модулей (`CommunicationApp/module/Users/module.php`): измените пространство имен на `Users`. Замените инструкцию `namespace ZendSkeletonModule;` инструкцией `namespace Users;`.
5. Следующие папки можно удалить, поскольку мы не будем использовать их в нашем проекте:
 - `Users/src/ZendSkeletonModule;`
 - `Users/view/zend-skeleton-module.`

Что сейчас произошло?

Мы установили модуль-заготовку для Zend Framework; этот модуль пуст, и нам понадобится доработать его, создав собственные контроллеры и представления. Наши следующие действия будут направлены на создание новых контроллеров и представлений для этого модуля.

СОЗДАНИЕ МОДУЛЯ С ПОМОЩЬЮ ПРОГРАММЫ ZFTOOL

ZFTool — это утилита для управления приложениями и проектами Zend Framework; ее также можно использовать для создания новых модулей. Вам потребуется установить программу ZFTool и с ее помощью создать модуль командой `create module`:

```
$ php composer.phar require zendframework/zftool:dev-master
$ cd vendor/zendframework/zftool/
$ php zf.php create module Users2 /var/www/CommunicationApp
```

Дополнительную информацию об утилите ZFTool можно получить по ссылке <http://framework.zend.com/manual/2.0/en/modules/zendtool.introduction.html>.

Модель, представление, контроллер

Фундаментальная цель любого фреймворка, поддерживающего паттерн модель-представление-контроллер (Model-View-Controller, MVC), — упростить разделение трех уровней: модели, представления и контроллера. Перед тем как погрузиться в детали создания модулей, попробуем быстро разобраться в том, как эти три уровня работают в MVC-фреймворке.

- **Модель** является отражением данных. Модель также включает в себя бизнес-логику для различных транзакций приложений.
- **Представление** содержит в себе логику вывода на экран, которая используется для визуализации различных элементов пользовательского интерфейса в веб-браузере.

- ❑ **Контроллер** управляет прикладной логикой в любом MVC-приложении; все действия и события обрабатываются контроллером. Уровень контроллера служит коммуникационным интерфейсом между моделью и представлением, который управляет состоянием модели и отражает изменения в представлении. Контроллер также предоставляет точку входа для доступа к приложению.

В новой MVC-структуре Zend Framework 2 все модели, представления и контроллеры сгруппированы по модулям. Каждый модуль имеет свой набор моделей, представлений и контроллеров, используя некоторые компоненты совместно с другими модулями.

Структура папок модуля Zend Framework

Структура папок модуля Zend Framework 2.0 содержит три ключевых компонента — варианты конфигурации, логику модуля, а также представления. В следующей таблице описано, как организовано содержимое модуля:

Имя папки	Описание
config	Используется для управления конфигурацией модуля
src	Содержит весь исходный код модуля, в том числе все контроллеры и модели
view	Содержит все представления, используемые в модуле

Время действовать — создание контроллеров и представлений

Теперь, когда мы создали модуль, наш следующий шаг — определить собственные контроллеры и представления. В этом разделе мы создадим два простых представления и напишем контроллер, который будет обеспечивать переключение между ними.

1. Перейдите в папку внутри модуля:

```
$ cd /var/www/CommunicationApp/module/Users
```

2. Создайте папку для контроллеров:

```
$ mkdir -p src/Users/Controller/
```

3. Создайте новый файл `IndexController` в папке `<имя модуля>/src/<имя модуля>/Controller/`:

```
$ cd src/Users/Controller/
$ vim IndexController.php
```

4. Добавьте следующий код в файл `IndexController`:

```
<?php
namespace Users\Controller;
use Zend\Mvc\Controller\AbstractActionController;
use Zend\View\Model\ViewModel;
class IndexController extends AbstractActionController
{
    public function indexAction()
    {
        $view = new ViewModel();
        return $view;
    }
    public function registerAction()
    {
        $view = new ViewModel();
        $view->setTemplate('users/index/new-user');
        return $view;
    }
    public function loginAction()
    {
        $view = new ViewModel();
        $view->setTemplate('users/index/login');
        return $view;
    }
}
```

5. Этот код выполняет следующие действия: если пользователь посещает домашнюю страницу, то ему выводится представление, предлагаемое по умолчанию; если пользователь совершает действие **register**, то ему выводится шаблон **new-user**; если пользователь совершает действие **login**, то выводится шаблон **login**.
6. Теперь, когда мы создали контроллер, нам нужно создать требуемые представления, которые будут выводиться при каждом из действий контроллера.
7. Создайте папки для представлений:

```
$ cd /var/www/CommunicationApp/module/Users
$ mkdir -p view/users/index/
```

8. Перейдите в папку представлений `<модуль>/view/<имя модуля>/index`:

```
$ cd view/users/index/
```