Занятие №7
Symfony, Composer, Реализация таблицы

О себе

Евгений Ермолаев

- 8 лет в веб-разработке (PHP)
- Magento, X-Cart, Symfony
- Тим-лид , разработчик

Composer

Что это?

Composer

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

Getting Started

Define Your Dependencies

Put a file named composer.json at the root of your project, containing your project dependencies:

```
{
    "require": {
        "vendor/package": "1.3.2",
        "vendor/package2": "1.*",
        "vendor/package3": ">=2.0.3"
    }
}
```

```
php composer.phar install
```

Установка Symfony2

Как создать приложение?

Установка Symfony2

Устанавливаем Symfony в желаемой папке:

php composer.phar create-project symfony/framework-standard-edition path/ '2.5.*'

```
Installing symfony/framework-standard-edition (v2.5.3)
  - Installing symfony/framework-standard-edition (v2.5.3)
    Downloading: 100%

Created project in path/
Loading composer repositories with package information
Installing dependencies (including require-dev)
  - Installing jdorn/sql-formatter (v1.2.17)
    Loading from cache

  - Installing psr/log (1.0.0)
    Loading from cache

  - Installing twig/twig (v1.16.0)
    Loading from cache

  - Installing doctrine/lexer (v1.0)
    Loading from cache

  - Installing doctrine/annotations (v1.2.0)
    Loading from cache

  - Installing doctrine/collections (v1.2)
    Loading from cache

  - Installing doctrine/cache (v1.3.0)
    Loading from cache

  - Installing doctrine/inflector (v1.0)
    Loading from cache
```

Ввод параметров

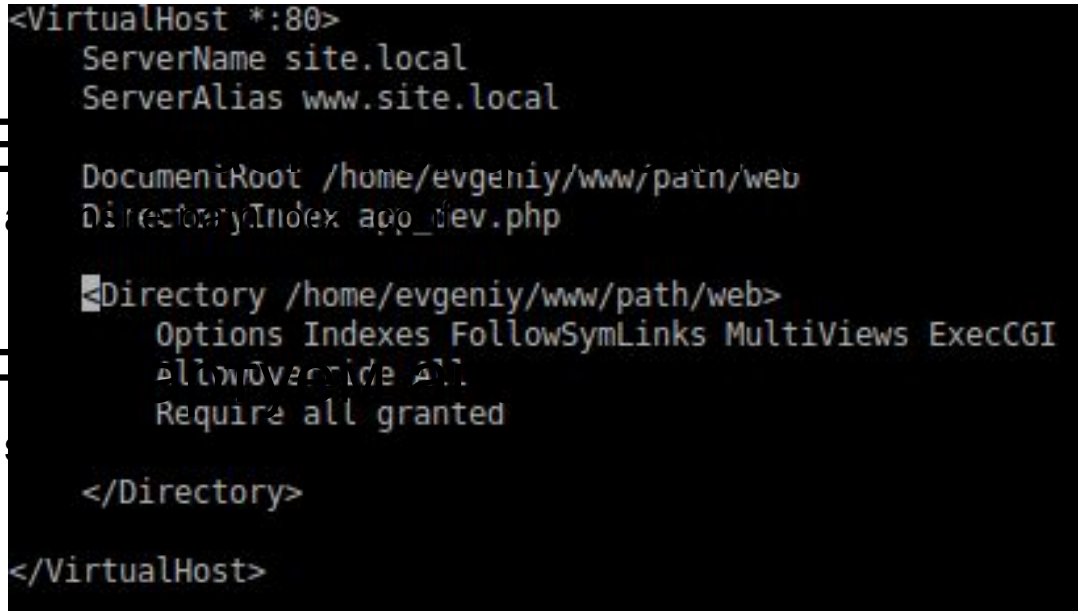
```
Generating autoload files
Would you like to install Acme demo bundle? [y/N] y
Installing the Acme demo bundle.
Creating the "app/config/parameters.yml" file
Some parameters are missing. Please provide them.
database_driver (pdo_mysql):
database_host (127.0.0.1):
database_port (null):
database_name (symfony):
database_user (root):
database_password (null):
mailer_transport (smtp):
mailer_host (127.0.0.1):
mailer_user (null):
mailer_password (null):
locale (en):
secret (ThisTokenIsNotSoSecretChangeIt):
debug_toolbar (true):
debug_redirects (false):
use_assetic_controller (true):
Clearing the cache for the dev environment with debug true
Installing assets as hard copies
```

Настройка сервера

1. Создаем файл настроек:

sudo nano /etc/apache2/sites-available/path.local.conf

2. В открывшемся редакторе вводим:



3. В

sudo

4. Р

sudo

Настройка сервера

5. Добавляем свой url в файл hosts:

sudo nano /etc/hosts

Пишем такую такую строчку в конец файла:

127.0.0.1 site.local

6. В папке проекта выставляем права:

sudo chmod 777 -R app/cache app/logs

7. Проверяем в браузере:

http://site.local/config.php
http://site.local/app_dev.php

Установка Symfony2

Как выглядит default page симфони проекта?

Symfony

Search on Symfony website OK

Welcome!

Congratulations! You have successfully installed a new Symfony application.

READ THE QUICK TOUR

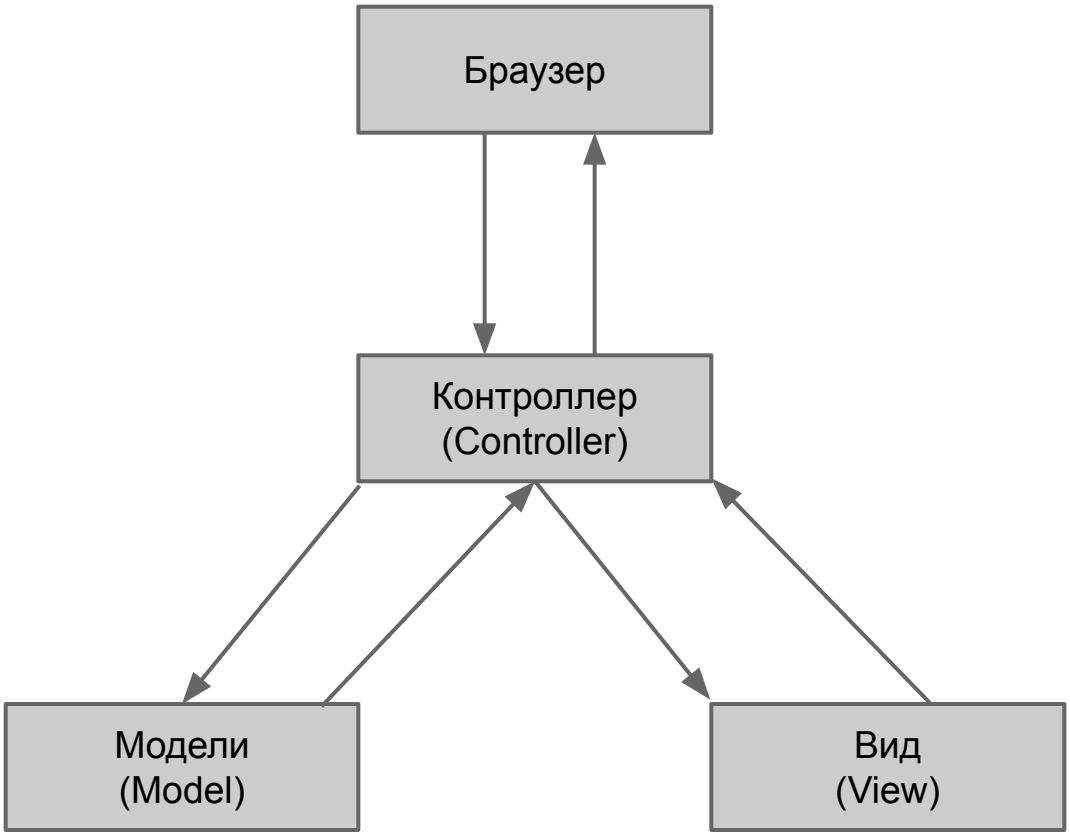
CONFIGURE

RUN THE DEMO

Принципы MVC-приложения

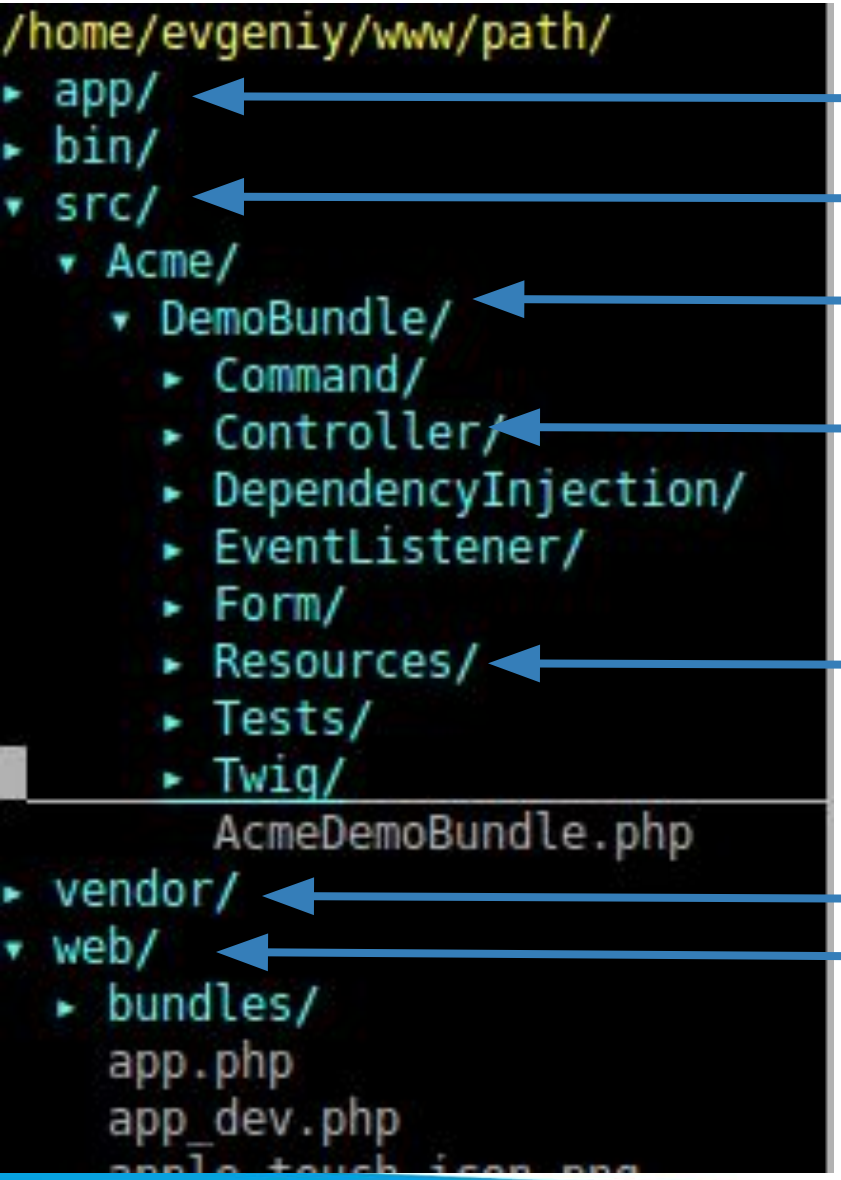
Что такое MVC?

Принципы MVC-приложения



Model
View
Controller

Структура Symfony2



Папка с настройками, логами и кэшем

Папка с вашими модулями

Ваш модуль

Контроллеры

Темплейты, настройки, css, js

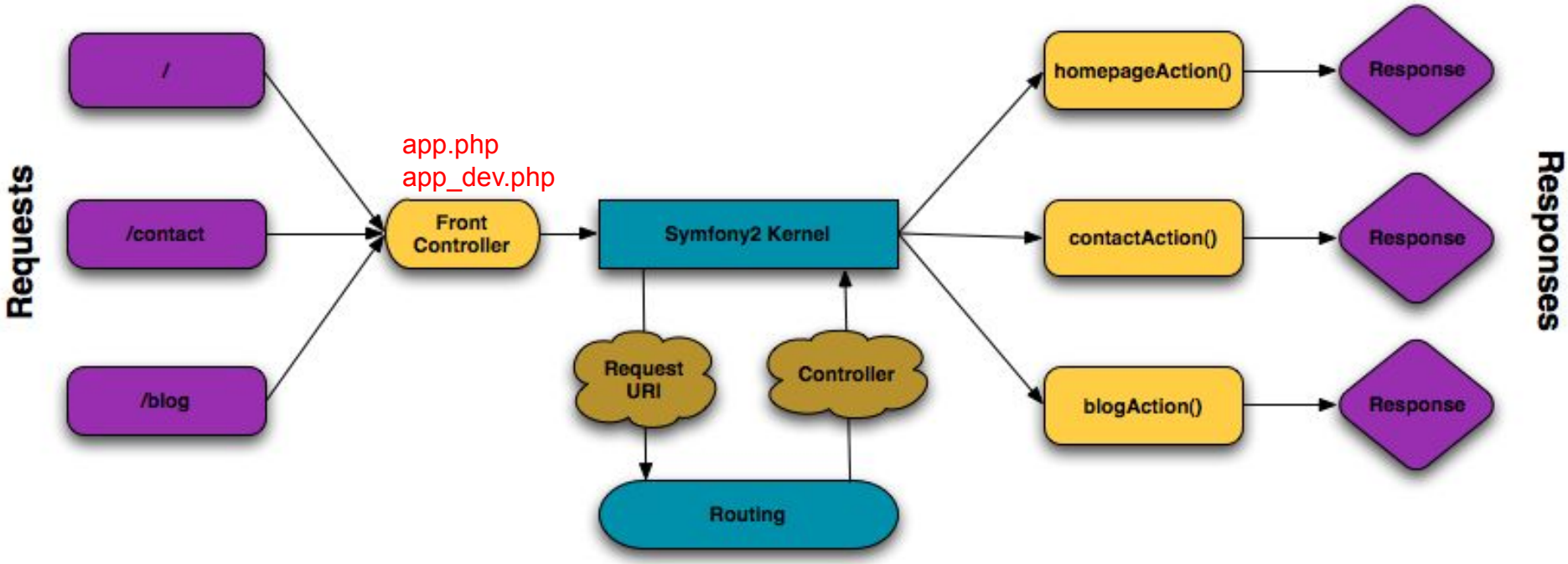
Сторонние библиотеки

Веб-папка

Роутинг

Что это?

Рабочий цикл веб-приложения



Роутинг

1. Глобальный файл app/config/routing_dev.yml

```yaml
_wdt:
    resource: "@WebProfilerBundle/Resources/config/routing/wdt.xml"
    prefix:   /_wdt

_profiler:
    resource: "@WebProfilerBundle/Resources/config/routing/profiler.xml"
    prefix:   /_profiler

_configurator:
    resource: "@SensioDistributionBundle/Resources/config/routing/webconfigurator.xml"
    prefix:   /_configurator

_main:
    resource: routing.yml

# AcmeDemoBundle routes (to be removed)
_acme_demo:
    resource: "@AcmeDemoBundle/Resources/config/routing.yml"
```

Роутинг

2. Файл модуля

src/Acme/DemoBundle/Resources/config/routing.yml

```yaml
_welcome:
    path:     /
    defaults: { _controller: AcmeDemoBundle:Welcome:index }

_demo_secured:
    resource: "@AcmeDemoBundle/Controller/SecuredController.php"
    type:     annotation

_demo:
    resource: "@AcmeDemoBundle/Controller/DemoController.php"
    type:     annotation
    prefix:   /demo
```

Роутинг

3. Файл контроллера

src/Acme/DemoBundle/Controllers/DemoController.php

```php
class DemoController extends Controller
{
    /**
     * @Route("/", name="_demo")
     * @Template()
     */
    public function indexAction()
    {
        return array();
    }

    /**
     * @Route("/hello/{name}", name="_demo_hello")
     * @Template()
     */
    public function helloAction($name)
    {
        return array('name' => $name);
    }
```

```php
/**
 * @Route("/contact", name="_demo_contact")
 * @Template()
 */
public function contactAction(Request $request)
{
    $form = $this->createForm(new ContactType());
    $form->handleRequest($request);
```

$get= $request->get(variable');

$post = $request->request->get(variable');

Генерация каркаса приложения

1. Создание модуля (бандла):

php -f app/console generate:bundle

* Название: SimbirSoft\TestBundle
* Аннотации
* Ответить “yes” на предложение генерации полной структуры папок

2. Проверка результата:

http://site.local/hello/test

Результат в браузере: “Hello test”

Генерация контроллеров

В консоле необходимо выполнить команду:

```
php -f app/console generate:controller
```

Далее в интерактивном режиме задать

- имя контроллера
- Формат роутинга – annotation
- Задать имена контроллерам, action route, темплейты

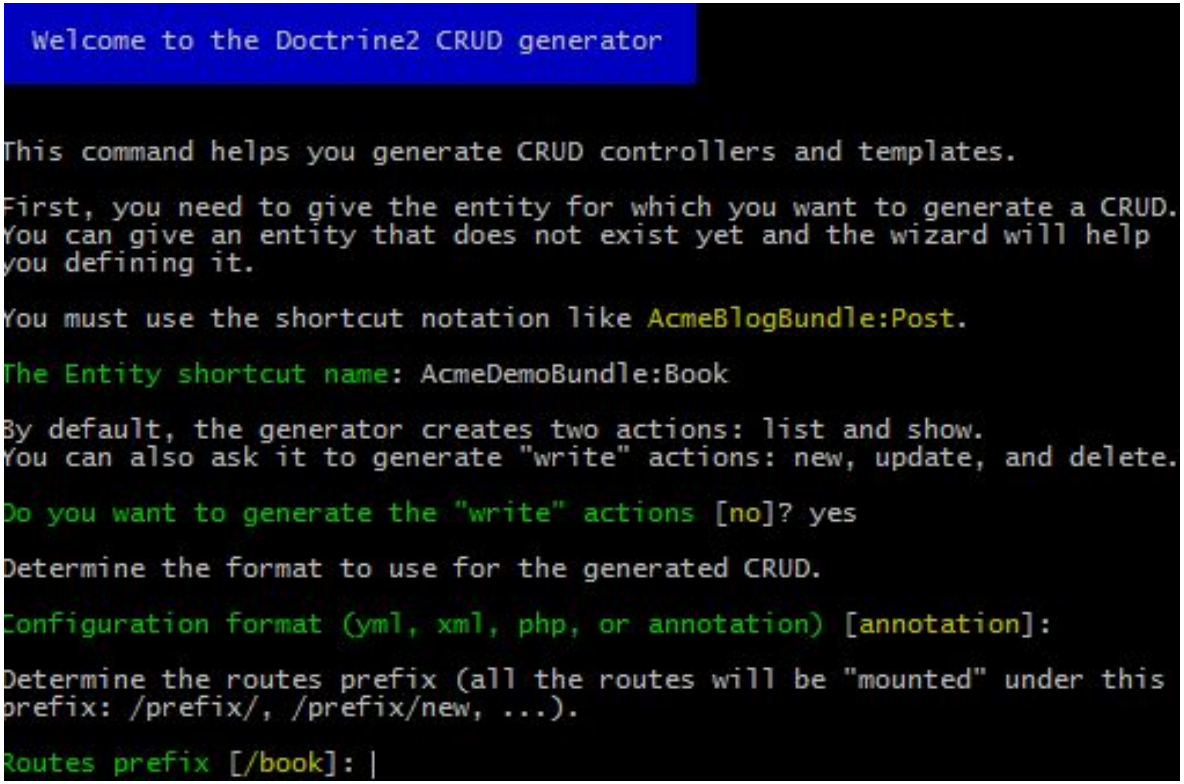
CRUD

Что это?

Генерация CRUD

В консоле необходимо выполнить команду:

php –f app/console generate:doctrine:crud



Реализация таблицы

Создание Action

```php
/**
 * Lists all Book entities.
 *
 * @Route("/", name="book")
 * @Method("GET")
 * @Template()
 */
public function indexAction()
{
    $em = $this->getDoctrine()->getManager();

    $entities = $em->getRepository('AcmeDemoBundle:Book')->findAll();

    return array(
        'entities' => $entities,
    );
}
```

Как ограничить результат

```
$repository = $em->getRepository('AcmeDemoBundle:Book');

$query = $repository->createQueryBuilder('p');

$query->setFirstResult($offset);
$query->setMaxResults($limit);

$books =  $query->execute();
```

```
$qb->select('u')
    ->from('User u')
    ->where('u.id = :identifier')
    ->orderBy('u.name', 'ASC')
    ->setParameter('identifier', 100);
```

Вывод в темплейте

```
{% extends '::base.html.twig' %}

{% block body -%}
    <h1>Book list</h1>

    <table class="records_list">
        <thead>
        <tr>
            <th>Id</th><th>Title</th><th>Pages</th><th>Created</th><th>Updated</th><th>Actions</th>
        </tr>
        </thead>
        <tbody>
        {% for entity in entities %}
            <tr>
                <td><a href="{{ path('book_show', { 'id': entity.id }) }}">{{ entity.id }}</a></td>
                <td>{{ entity.title }}</td>
                <td>{{ entity.pages }}</td>
                <td>{% if entity.created %}{{ entity.created|date('Y-m-d H:i:s') }}{% endif %}</td>
                <td>{% if entity.updated %}{{ entity.updated|date('Y-m-d H:i:s') }}{% endif %}</td>
                <td>
                    <ul>
                        <li>
                            <a href="{{ path('book_show', { 'id': entity.id }) }}">show</a>
                        </li>
                    </ul>
                </td>
            </tr>
        {% endfor %}
        </tbody>
    </table>

{% endblock %}
```

Загрузка шаблона

1. Используется шаблонизатор Twig
2. Соответствие контроллера и шаблона:

```
Resources/views/Demo/hello.html.twig
Controller/DemoController.php , helloAction()
```

3. В шаблоне доступны данные возвращаемые из контроллера.

```php
/**
 * @Route("/hello/{name}", name="_demo_hello")
 * @Template()
 */
public function helloAction($name)
{
    return array('name' => $name);
}
```

```
{% extends "AcmeDemoBundle::layout.html.twig" %}

{% block title "Hello " ~ name %}

{% block content %}
    <h1>Hello {{ name }}!</h1>
{% endblock %}

{% set code = code(_self) %}
```

Использование git

1. Установка git: sudo apt-get install git-core
2. Генерация ключей доступа: ssh-keygen
3. Из папки .ssh залить файл id_rsa.pub на сайт github или gitlab
4. Получить код проекта: git clone git@github.com:eermolaev/test2.git
5. Создаем файлы
6. Добавляем файлы в систему контроля: git add .
7. Коммитаем файлы в локальную ветку: git commit –m ‘My desciption’
8. Отправка изменения на сервер: git push
9. Получить изменения с сервера: git pull
10. Статус: git status

Пример приложения

https://github.com/hiend/simbirsoft_examples/tree/ch06ch07ch08

Домашнее задание

1. Добавить столбцы: price, author, in_stock
2. Вывести в таблице
3. Если книги на складе нет, то строка выделяется цветом
4. Цена должна быть отформатирована (currency)
5. Выводить по 5 записей
6. Добавить ссылка «Next», которая выводит следующие 5 записей.