Frameworks

Chapter 18

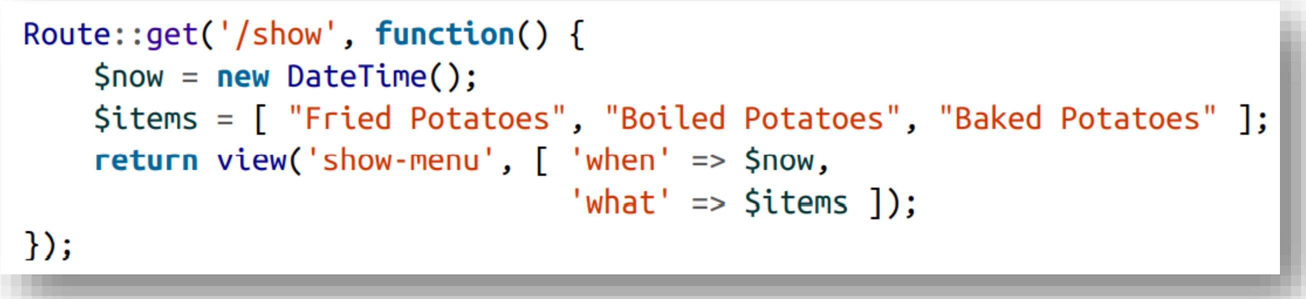
Frameworks

- A framework is a set of functions, classes, and conventions that make it easier to accomplish common tasks.
- By Framework we can save time to implement an APP, but must spend time to learn
- Frameworks generally provide standard ways to accomplish at least the following tasks:
  - **Routing**: Translating user-requested URLs to specific methods or functions
  - **Object-relational mapping:** treat rows in your database as **objects** in your code and **providing methods on those objects**
  - **User management:** creating and maintaining users accounts and permisssions

- Examples of framworks:
  - Laravel
  - Symfony
  - Zend

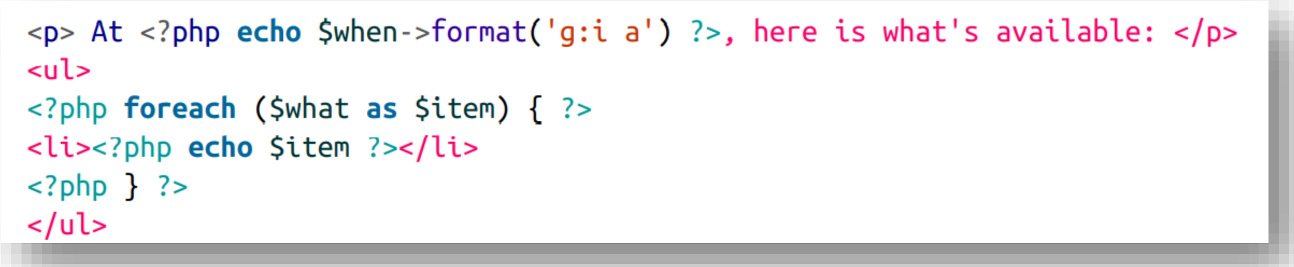
Laravel

- Laravel describes itself as **elegance**, **simplicity**, and **readability**.
- It is based on an **MVC architecture**
- You must install **Composer** then **Laravel** to be able to use it
- An **sample code** of Laravel for routing



```
Route::get('/show', function() {
    $now = new DateTime();
    $items = [ "Fried Potatoes", "Boiled Potatoes", "Baked Potatoes" ];
    return view('show-menu', [ 'when' => $now,
                               'what' => $items ]);
});
```

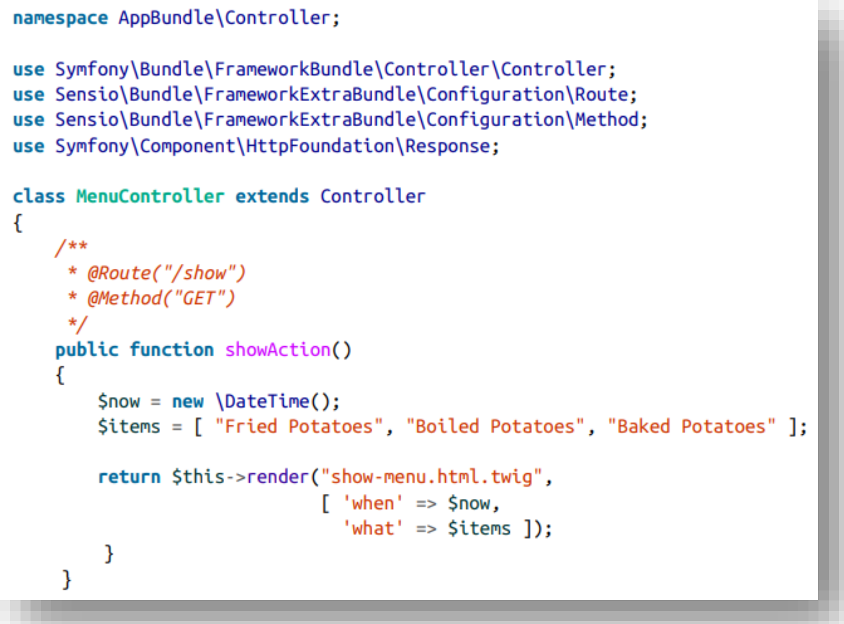
- A **view** is a template for **presentation logic**
- The **view()** in Laravel looks for **a file in a predefined location** and then runs it to generate a response
- The call to **view('show-menu')** tells Laravel to look for a file named **show-menu.php** in the **resources/views** directory



```
<p> At <?php echo $when->format('g:i a') ?>, here is what's available: </p>
<ul>
<?php foreach ($what as $item) { ?>
<li><?php echo $item ?></li>
<?php } ?>
</ul>
```

Symfony

- It describes itself as a set of reusable components and a framework for web projects
- With Symfony, routes are not specified in one central place
- individual classes in the src/AppBundle/Controller directory define the triggered methods
- A special annotation in a comment next to a method indicates what route the method handles
- An Example:

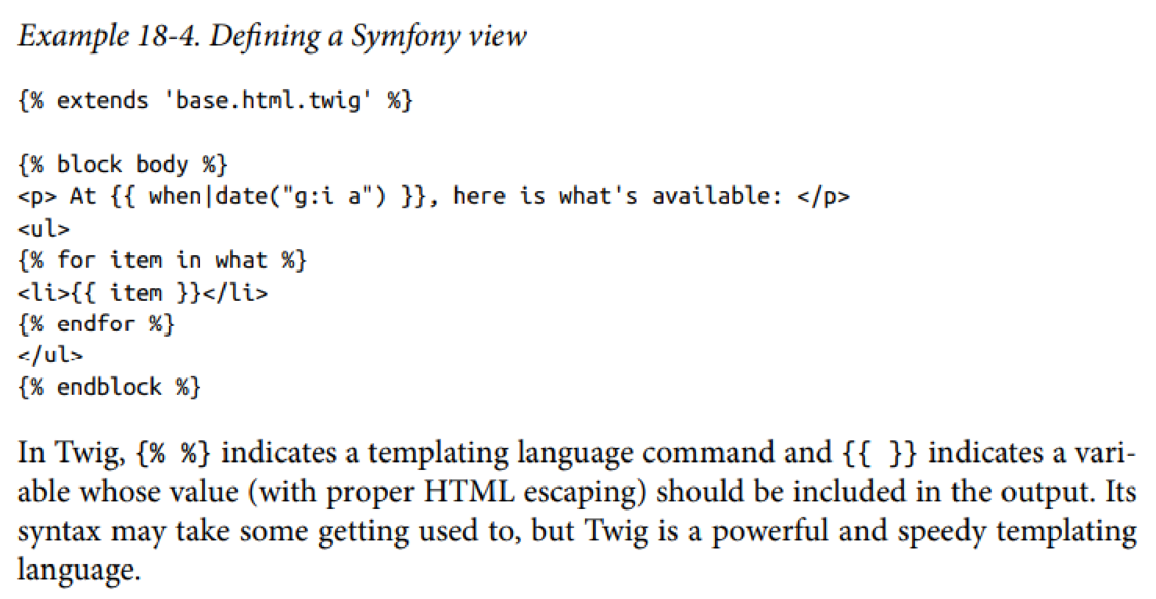


```
namespace AppBundle\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Method;
use Symfony\Component\HttpFoundation\Response;

class MenuController extends Controller
{
    /**
     * @Route("/show")
     * @Method("GET")
     */
    public function showAction()
    {
        $now = new \DateTime();
        $items = [ "Fried Potatoes", "Boiled Potatoes", "Baked Potatoes" ];

        return $this->render("show-menu.html.twig",
                             [ 'when' => $now,
                               'what' => $items ]);
         }
     }
```



Example 18-4. Defining a Symfony view

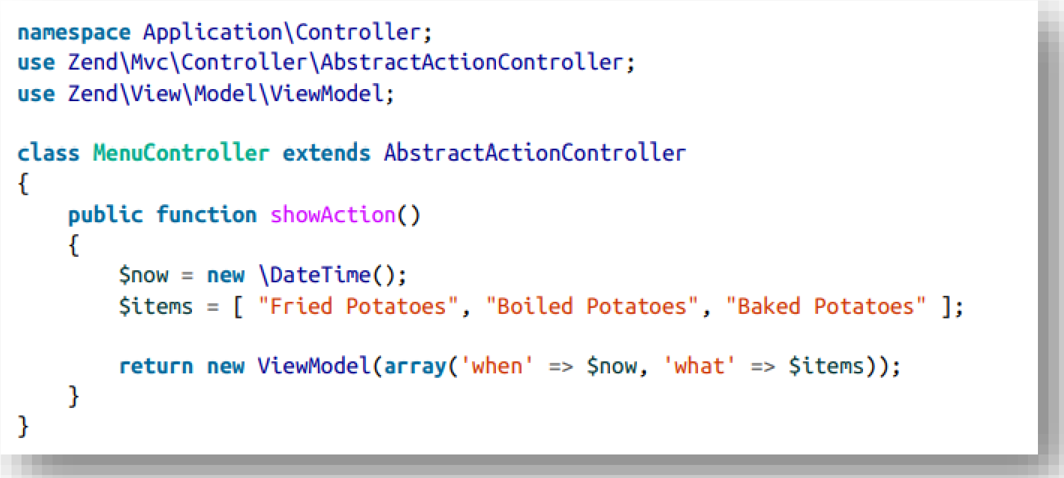
```
{% extends 'base.html.twig' %}

{% block body %}
<p> At {{ when|date("g:i a") }}, here is what's available: </p>
<ul>
{% for item in what %}
<li>{{ item }}</li>
{% endfor %}
</ul>
{% endblock %}
```

In Twig, {% %} indicates a templating language command and {{ }} indicates a variable whose value (with proper HTML escaping) should be included in the output. Its syntax may take some getting used to, but Twig is a powerful and speedy templating language.

Zend

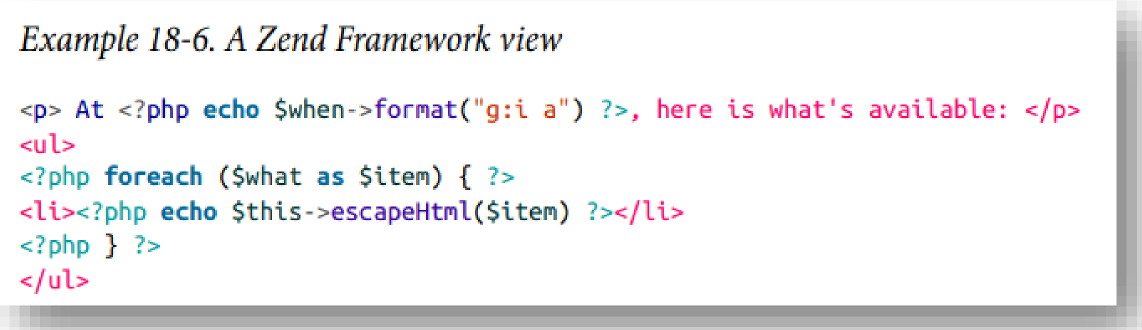
- It describes itself as a collection of components
- Zend organizes application code into modules
  - separate modules for separate high-level parts of program
- An Example:
  - There is base Application module: contains some default routing logic that maps paths to controller classes in a specific place in the filesystem.
  - MenuController.php saved in the module/Application/src/



```
namespace Application\Controller;
use Zend\Mvc\Controller\AbstractActionController;
use Zend\View\Model\ViewModel;

class MenuController extends AbstractActionController
{
    public function showAction()
    {
        $now = new \DateTime();
        $items = [ "Fried Potatoes", "Boiled Potatoes", "Baked Potatoes" ];

        return new ViewModel(array('when' => $now, 'what' => $items));
    }
}
```



Example 18-6. A Zend Framework view

```
<p> At <?php echo $when->format("g:i a") ?>, here is what's available: </p>
<ul>
<?php foreach ($what as $item) { ?>
<li><?php echo $this->escapeHtml($item) ?></li>
<?php } ?>
</ul>
```