Premiers pas avec symfony

Symfony est un framework MVC et qui utilise le paradigme orienté objet de PHP. Il permet de développer des applications complexes mais maintenables.

Controllers & Endpoints

Pour répondre à des requêtes HTTP, votre application a besoin de *Controllers* les controllers sont des classes qui contienent les méthodes endpoints. Une méthode endpoint est une méthode qui va gérer une requête sur une route particulière et avec une méthode HTTP particulière.

Premier Controller

Pour commencer, créez dans le dossier `src/Controller` de votre projet une nouvelle classe PHP :

2020-10-02-13_19_18-tuto-symfony---bas.png

Ensuite, pour configurer la route exécutez la commande suivante afin d'installer un package symfony :

```
composer require annotations
```

Vous pouvez ensuite écrire une méthode enpoint dans votre classe :

```
namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class HelloWorldController
{
   /**
    * @Route("/helloworld")
    */
    public function HelloWorld(){
        return new Response("Hello World from Symfony !");
    }
}
```

L'annotation *@Route* est dans un commentaire mais elle est bien comprise par le framework, elle permet de lier votre méthode endpoint à la route `/helloworld`. Pour essayer tout ça, exécutez :

```
symfony server:start
```

puis rendez vous à : http://127.0.0.1:8000/helloworld

Les Templates

Pour être MVC, il faut dissocier la vue du Controller. Pour cela, nous allons utiliser des Templates afin de décrire les vues.

Dans un premier temps, exécutez la commande suivante pour installer le module de Symfony pour les templates :

```
composer require twig
```

Puis créez dans le dossier `src/templates` un nouveau dossier `hello` et dans ce dossier un fichier `hello.html.twig` avec le contenu suivant :

```
{# templates/lucky/number.html.twig #}
```

Ensuite, changez le code de votre Controller comme ceci, afin de rendre le template :

```
namespace App\Controller;



use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class HelloWorldController extends AbstractController
{
    /**
    * @Route("/helloworld")
    */
    public function HelloWorld(){

        $name = "Shepard";
        return $this->render("hello/hello.html.twig", [
            "name" => $name
```

```
        ]);
    }
}
```

Notre Controller étends désormais la classe `AbstractController` afin de pouvoir accèder aux méthodes de rendu de template. On utilise donc la méthode `render` avec en paramètre le nom de notre template ainsi que des données à afficher via un tableau associatif. Enfin, ajouter cette ligne à votre template pour afin d'afficher les données en question :

```
<h1>Hello {{ name }} !</h1>
```

Lancez ensuite votre application et allez sur l'URL de l'endpoint pour constater le rendu de votre template !

A propos des Routes

Nommage

Afin de faciliter le debuggage et la génération d'URLs, il est recommander de nommer ses URLs avec l'attribut `name` :

```
    /**
    * @Route("/helloworld", name="hello")
    */
    public function HelloWorld(){ ... }
```

Méthode HTTP

En définissant la Route d'une méthode endpoint, vous pouvez filtrer par méthode HTTP de la façon suivante :

```
    /**
    * @Route("/helloworld",name="hello" methods={"GET"})
    */
    public function HelloWorld(){ ... }
```

Cela est très utile car cela aide à séparer automatique la logique de présentation et la logique de modification d'un formulaire par exemple.

Paramètres de Route

Il est possible de passer des paramètres dans une route. Attention à ne pas confondre avec les paramètre d'URL qui se trouve après un `?` et sont séparés par des `&`. Un paramètre de route est une partie dynamique de l'URL, que le client utilise pour passer un paramètre au serveur. Souvent, il s'agit du nom ou identifiant de la ressource concernée par la requête. Par exemple dans une application gèrant un blog, la route /posts/6 concernerait le poste de blog avec l'identifiant 6.

Pour déclarer un paramètre de route, procédez ainsi :

```
 /**
     * @Route("/blog/{id}", name="blog_show", methods = {"GET"})
     */
    public function show(int $id) { ... }
```

Revision #7
Created 2020-10-01 22:16:58 UTC by Arsène Lapostolet
Updated 2021-09-12 13:45:53 UTC by Arsène Lapostolet