



Créez des pages dans Symfony

Lotfi KHEDIRI



Créer une page

La création d'une nouvelle page consiste à

- ▢ **Créer une route**: Une Route est l'URL (par exemple **/about**) de votre page et pointe vers un contrôleur;
- ▢ **Créer un contrôleur** : Un contrôleur est la fonction PHP que vous écrivez qui construit la page.
 - ▢ Une demande entrante (Objet **Request**) >>>> est utilisée pour créer une réponse (objet **Response**)
 - ▢ Un objet Response peut contenir du **contenu HTML**, une **chaîne JSON** ou même un **fichier binaire** comme une image ou un PDF. (voir [Symfony and HTTP Fundamentals](#))

Création d'une page: route et contrôleur

- ▮ Une page `"/lucky/number"` qui génère un nombre chanceux aléatoire et l'imprime
- ▮ Créez une classe `"Controller"` et une méthode `"controller"` à l'intérieur:

```
<?php
// src/Controller/LuckyController.php
namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;

class LuckyController
{
    public function number()
    {
        $number = random_int(0, 100);
        return new Response('<html><body>Lucky number: '.$number.'</body></html>');
    }
}
```



La route

- ▢ Vous devez maintenant associer cette **fonction contrôleur** à une **URL publique** (par exemple /lucky/number) afin que la méthode number() soit exécutée lorsqu'un utilisateur y accède.
- ▢ Cette association est définie en créant une **Route** dans le fichier **config/routes.yaml**
- ▢ C'est le fichier de configuration des routes

La route (en utilisant routes.yaml)

- ▮ Le fichier **config/routes.yaml** : fichier de configuration des routes
- ▮ tester en allant sur: <http://localhost:8000/lucky/number>

```
# config/routes.yaml

# the "app_lucky_number" route name is not important yet
app_lucky_number:
  path: /lucky/number
  controller: App\Controller\LuckyController::number
```

Configuration des routes avec les annotations

- ▢ Au lieu de YAML, Symfony permet d'utiliser des *annotations*
- ▢ Pour ce faire, installer le package d'annotations:

```
$ composer require annotations
```

```
class LuckyController  
{
```

```
    /**
```

```
        * @Route("/lucky/number")
```

```
        */
```

```
    public function number()  
    {
```

```
        // this looks exactly the same
```

```
    }
```

```
}
```



La commande bin/console

C'est la **CLI** (commande line interface) de symfony :
>> c'est un ensemble de commande fournit par symfony qui permet de simplifier plusieurs taches de développement.

```
$ php bin/console
```

```
$ php bin/console debug:router
```

La barre d'outils de débogage Web

- ❑ L'une des fonctionnalités Symfony est la **barre d'outils de débogage Web**:
- ❑ Affiche une *énorme* quantité d'informations de débogage au bas de votre page
- ❑ Tout cela est inclus dans le bundle **symfony/profiler-pack**.



Rendu avec un template

- ❑ Le controleur doit **étendre** la classe de base de Symfony **AbstractController** :
- ❑ utilisez la méthode **render()** pour rendre un **template** (twig) et lui passer une variable number

```
❑ class LuckyController extends AbstractController
❑ {
❑     /**
❑      * @Route("/lucky/number")
❑      */
❑     public function number()
❑     {
❑         $number = random_int(0, 100);
❑         return $this->render('lucky/number.html.twig', ['number' => $number,]);
```

Création du template twig

- ❑ Créez un nouveau répertoire **templates/lucky** avec un nouveau fichier **number.html.twig**

```
{# templates/lucky/number.html.twig #}  
<h1>Your lucky number is {{ number }}</h1>
```

- ❑ Dans la partie templates , vous apprendrez tout sur Twig: comment boucler, rendre d'autres modèles et tirer parti de son puissant système d'héritage de mise en page.

Documentation Twig

□ <https://twig.symfony.com/doc/3.x/>



You are reading the documentation for Twig 3.x. Switch to the documentation for Twig [1.x](#). [2.x](#).

Twig Documentation

Read the online documentation to learn more about Twig.

- [Introduction](#)
- [Installation](#)
- [Twig for Template Designers](#)