esprit
Se former autrement
HONORIS UNITED UNIVERSITIES

Contrôleur et Routage

UP Web

AU: 2024/2025

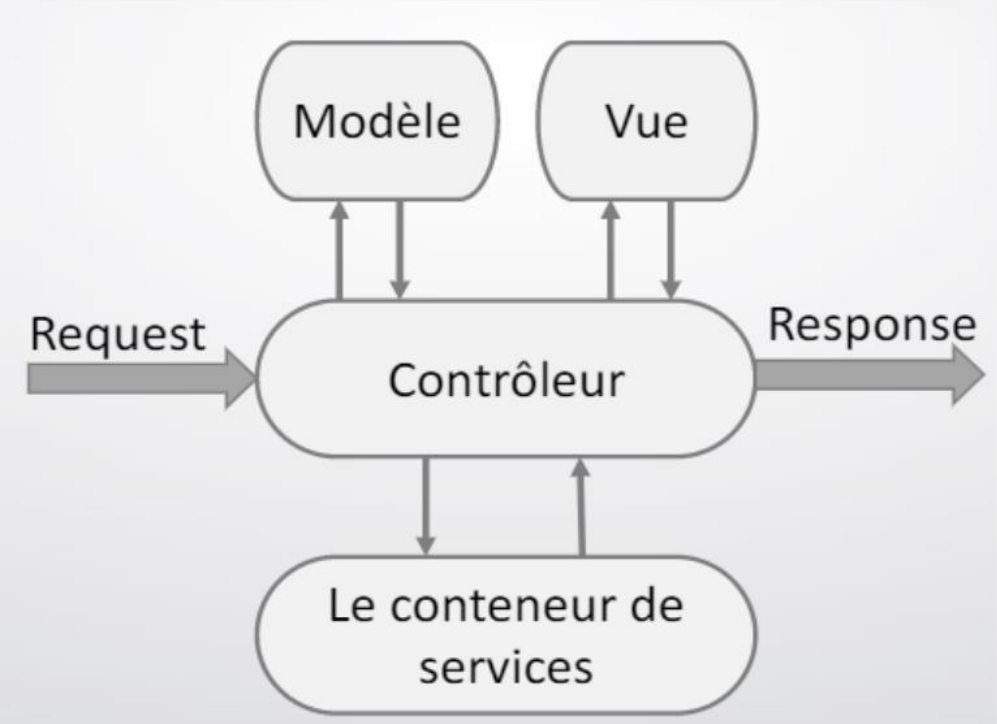
United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
EUR-ACE® Délivrée par la Commission des Titres d'Ingénieur
CONFERENCE DES GRANDES ÉCOLES
cdio CONCEIVE DESIGN IMPLEMENT OPERATE

Plan

- Le Contrôleur
- Le routage

Le Contrôleur

- Un contrôleur est un élément indispensable de l'architecture MVC.
- Il reçoit une requête et il interagit avec les différents composants d'une application Symfony pour renvoyer une réponse (un objet Response Symfony).



Le Contrôleur

- Un contrôleur est une classe PHP responsable de traiter une requête et de retourner une réponse
- Un contrôleur hérite de la classe **AbstractController** pour bénéficier des méthodes pratiques fournies par Symfony.
- Il existe 2 méthodes pour créer un contrôleur: manuellement ou en utilisant la dépendance **MakerBundle**.

1ère méthode:

- **Étape 1** : Créer une nouvelle classe de contrôleur
- **Étape 2** : Étendre **AbstractController**
- **Étape 3** : Ajouter des méthodes publiques pour définir des actions
- **Étape 4** : Ajouter la route pour les lier aux URLs.

Le Contrôleur

2ème méthode:

- Lancer la commande: **symfony console make:controller ou**

  php bin/console make:controller

```
C:\xampp\htdocs\test_me_symfony6>symfony console make:controller

 Choose a name for your controller class (e.g. OrangeChefController):
 > DefaultController

 created: src/Controller/DefaultController.php
 created: templates/default/index.html.twig

  Success!

 Next: Open your new controller class and add some pages!
```

- Cette méthode permet de créer un contrôleur avec une méthode de test et une vue.
- Vous pouvez nommer directement le contrôleur :

  symfony console make:controller controller_name

Le Contrôleur

Exemple d'un contrôleur:

```php
<?php

namespace App\Controller;

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

class DefaultController extends AbstractController
{
    #[Route('/default', name: 'app_default')]
    public function index(): Response
    {
        return $this->render('default/index.html.twig', [
            'controller_name' => 'DefaultController',
        ]);
    }
}
```

Le Contrôleur

Il existe plusieurs types de réponse:

- Un message simple
- une page HTML (Twig)
- un document XML
- un tableau JSON sérialisé
- etc…

Le Contrôleur

Les fonctions de base de la classe AbstractController

Méthode	Fonctionnalité	Valeur de retour
generateUrl()	Génère une URL à partir de la route	String
forward(String Action,array ()$parameters)	Forward la requête vers un autre contrôleur	Response
Redirect(string $url, int $statut	Redirige vers une url	RedirectResponse
RedirectToRoute(string $route, array $parameters)	Redirige vers une route	Rsponse
Render(string $view,array $parameters)	Affiche une vue	Response
Get (string $id)	Retourne un service à travers son id	objet
createNotFoundException(String $message)	Retourne une NotFoundException	NotFoundException

Le Routage

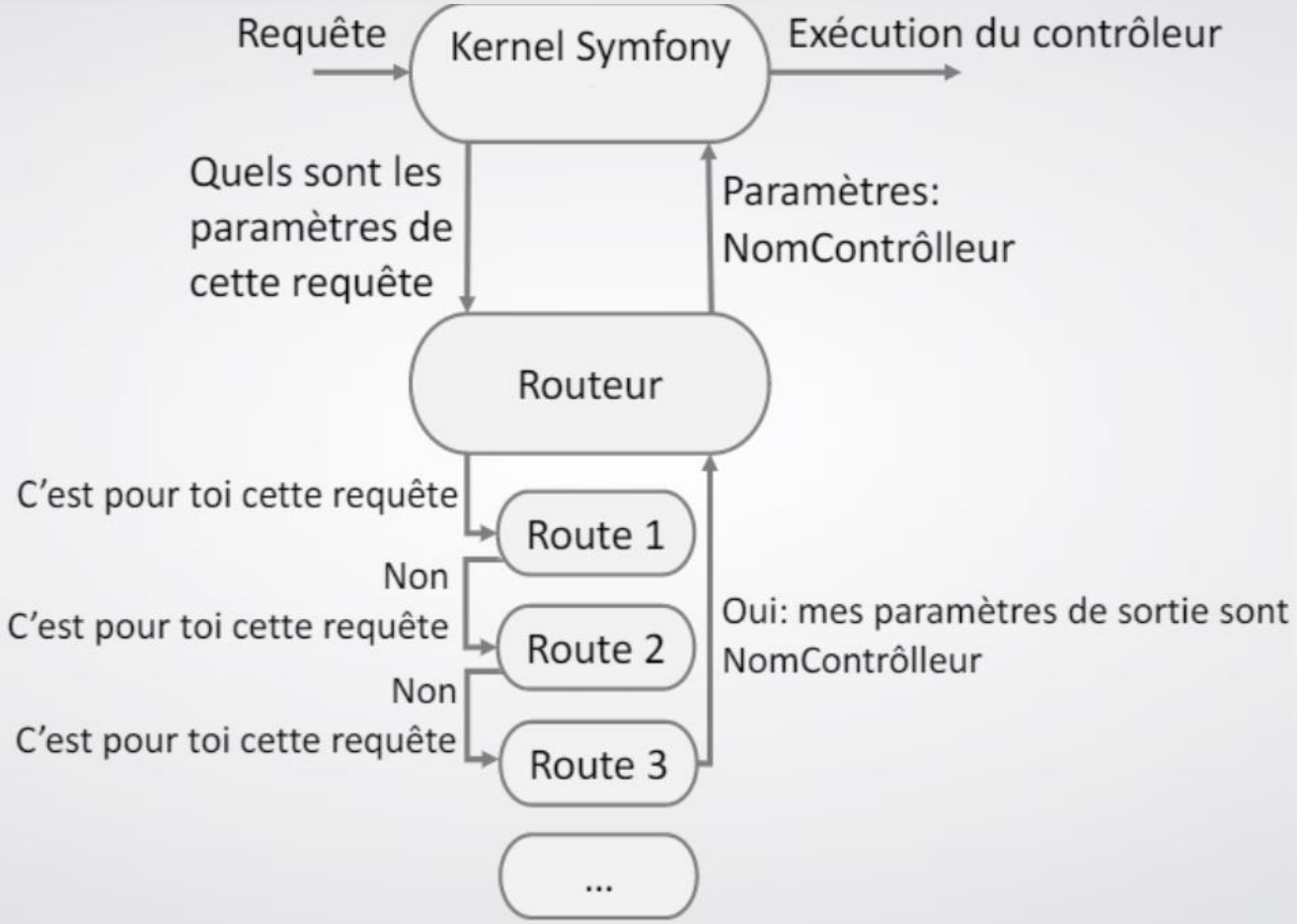
- Le routage est le processus de mappage d'une URL à un contrôleur et une action spécifique.

- L’objectif du routeur est de chercher la route qui correspond à l'URL appelée, et de retourner les paramètres de cette route.

- le composant de routage analyse l'URL de la requête et exécute le contrôleur approprié en fonction de la route définie.

Le Routage

Principe de fonctionnement du routage

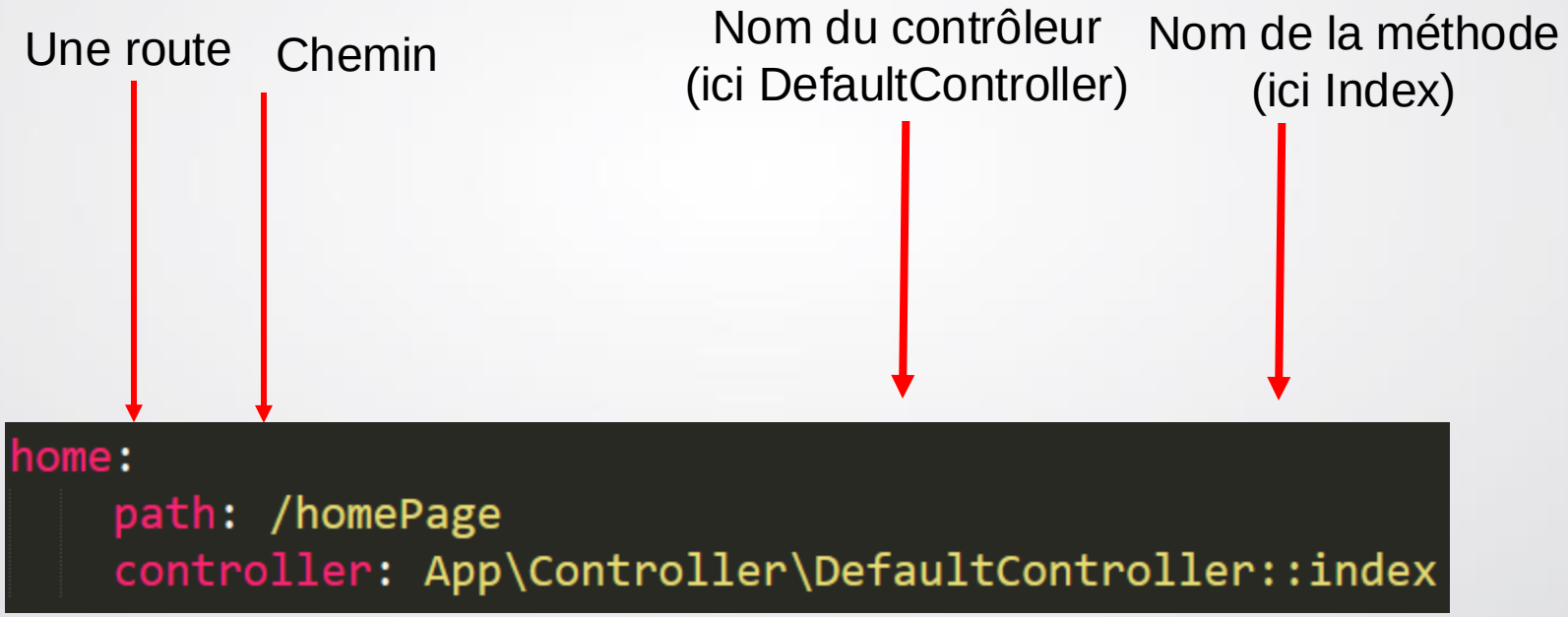


Le Routage

Il existe 3 méthodes pour créer une route:

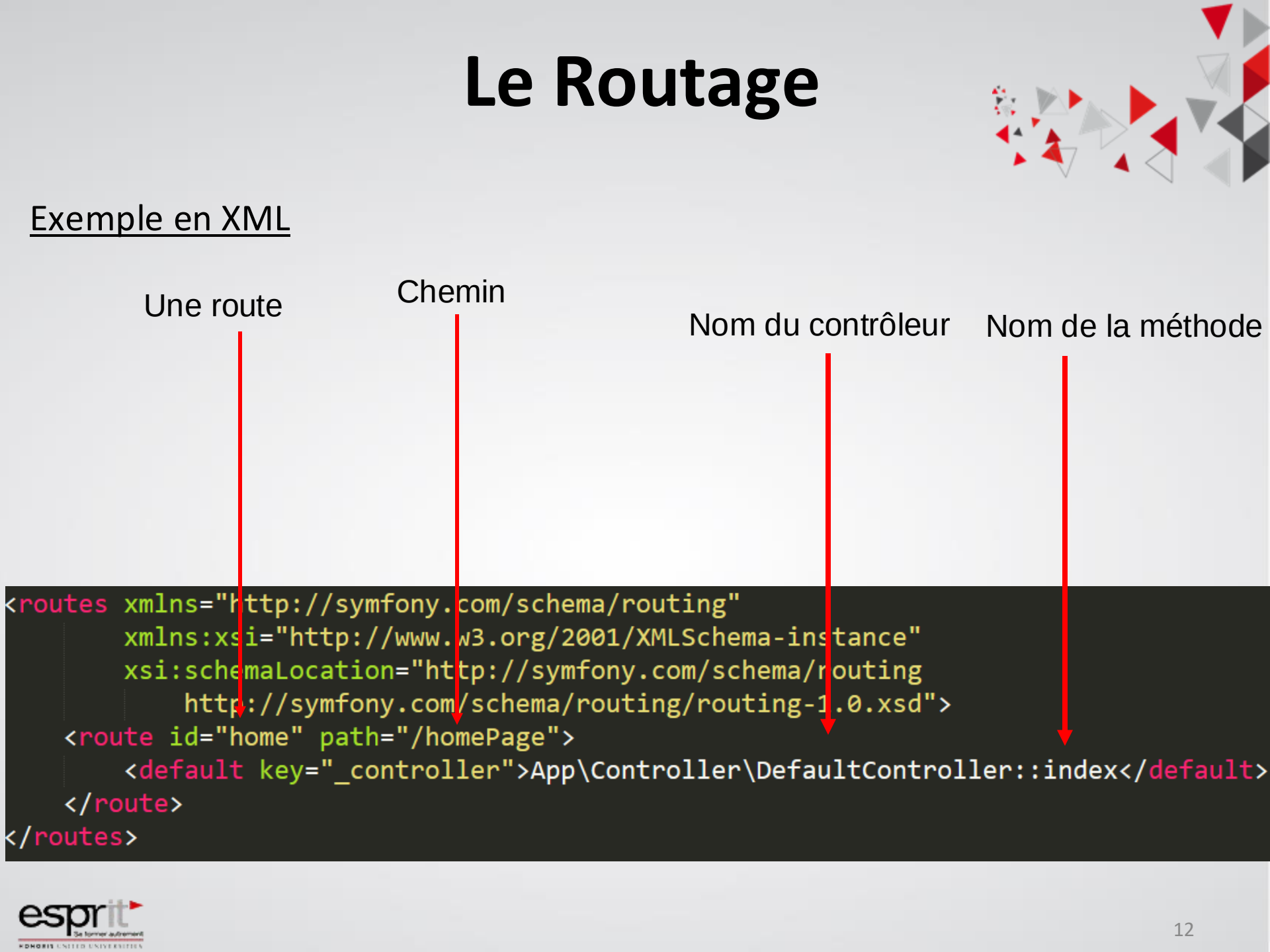
1ère méthode: en utilisant les fichiers de configuration config/routes.yaml ou config/routes.php ou config/routes.xml ,

Exemple en YAML



Le Routage

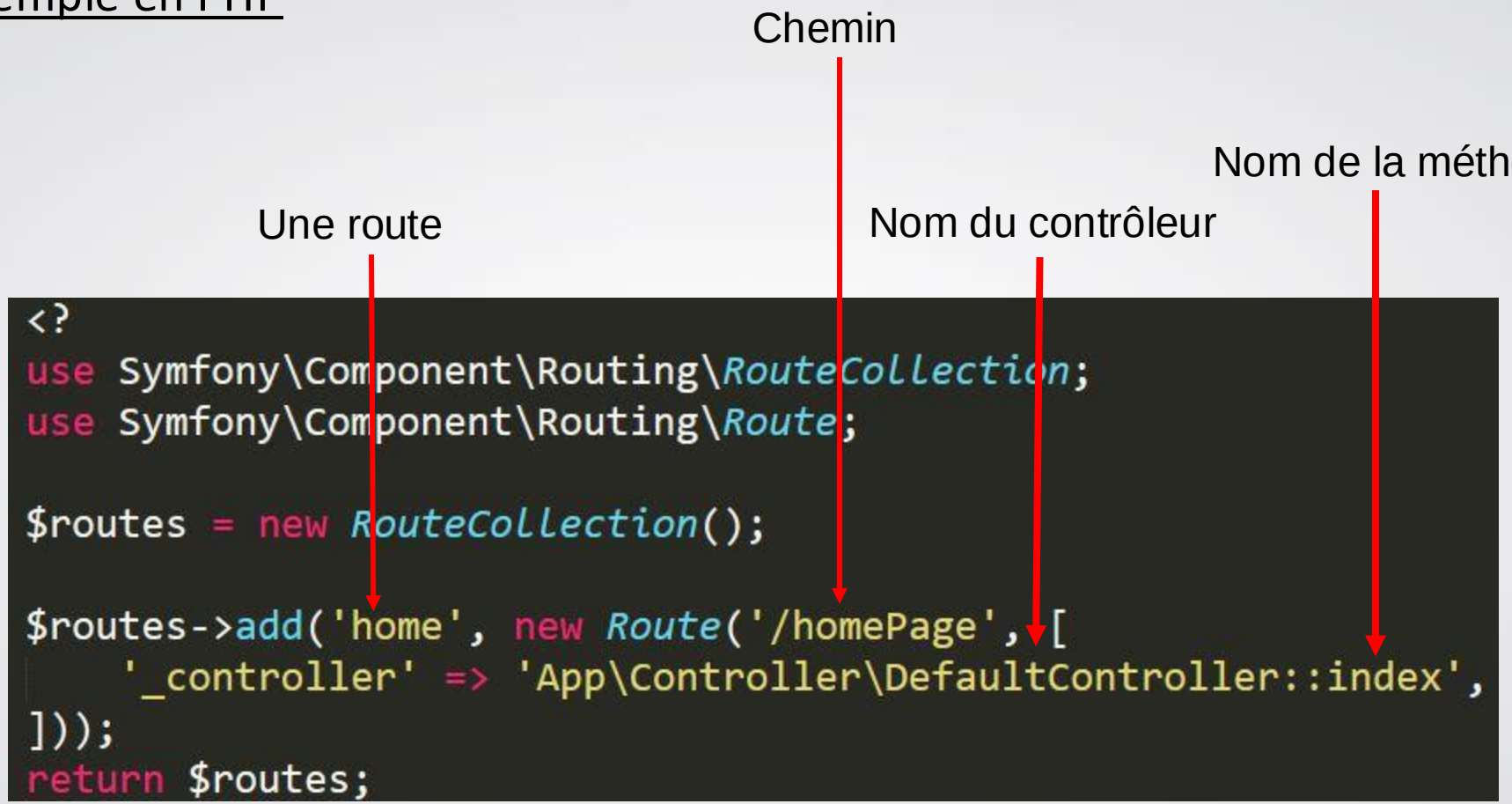
Exemple en XML




```xml
<routes xmlns="http://symfony.com/schema/routing"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://symfony.com/schema/routing
            http://symfony.com/schema/routing/routing-1.0.xsd">
    <route id="home" path="/homePage">
        <default key="_controller">App\Controller\DefaultController::index</default>
    </route>
</routes>
```

Le Routage

Exemple en PHP



Le Routage

2ème méthode: En utilisant les annotations

Au lieu de placer toutes les routes de l'application dans un seul fichier, il peut-être plus souple d'utiliser les annotations dans le même code du contrôleur

Chemin de la route **Nom de la route**

```
/**
 * @Route("/homePage", name="home")
 */
public function index(): Response
{

    return new Response ("hello world");
    return $this->render('default/index.html.twig', [
        'controller_name' => 'DefaultController',
    ]);
}
}
```

Le Routage

3ème méthode: En utilisant les attributs

Les annotations de route peuvent être définies à l'aide des attributs de PHP 8, ce qui est plus moderne et lisible.

Chemin de la route **Nom de la route**

```
    #[Route('/homePage', name: 'home')]
    public function index(): Response
    {

        return new Response ("hello world");
        return $this->render('default/index.html.twig', [
            'controller_name' => 'DefaultController',
        ]);
    }
}
```

Le Routage

Préfixe de Route

On utilise des préfixes de route pour réduire la redondance et organiser les routes.

```php
#[Route('/admin')]
class DefaultController extends AbstractController
{
    #[Route('/dashboard', name: 'dashboardAdmin')]
    public function dashboardAdmin()
    {
        return new Response('Tableau de bord Admin');
    }
}
```

Préfixe

Le Routage

Le routage avec paramètres

- Nous utilisons une annotation avec un paramètre variable « id » comme ci-dessous

```php
/**
 * @Route("/view/{id}", name="view")
 */
public function view($id)
{
    // $id vaut 5 si l'URL appelée est /view/5
    return new Response("Affichage d'id : ".$id);
}
```

- "{id}" est la variable qui permettra d'avoir une route dynamique.
- La méthode view() prend en paramètre la variable « $id» pour l'afficher

Atelier 2:
Contrôleur + Routage

Références

https://symfony.com/doc/current/controller.html

https://symfony.com/doc/current/controller.html#generating-controllers

https://symfony.com/doc/6.4/routing.html

https://symfony.com/doc/6.4/bundles/best_practices.html#routing

Merci pour votre attention