

# Symfony2 Routing

---

AYMEN SELLAOUTI

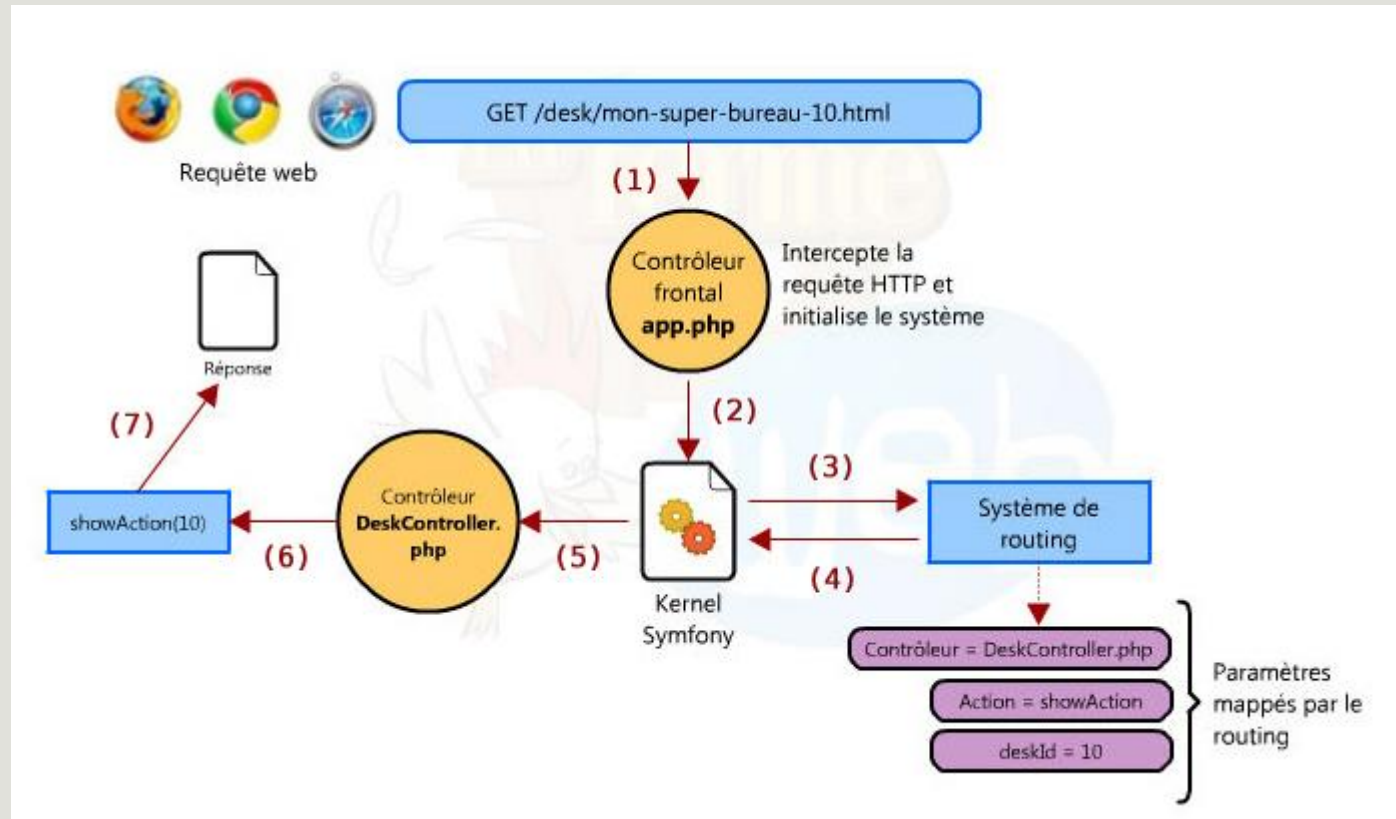
# Introduction

---

Système permettant de

- gérer les liens internes du projet
- avoir des URLs plus explicites
- associer une URL à une action

# Introduction



Cette architecture illustre le fonctionnement de Symfony.

1- Requête de l'utilisateur

2- La requête est envoyée vers le noyau de Symfony.

3- Le Noyau consulte le routeur afin de connaître quel Action exécuter.

4- Le routeur envoie les informations concernant l'URI

5 6- Le noyau invoque l'action à exécuter.

7- L'action retourne la réponse.

Routing (<http://www.lafermeduweb.net/>)

# Format de gestion du routing

---

Les fichiers de routing peuvent être de quatre formats différents :

➤ YAML

➤ XML

➤ PHP

➤ Annotations

# Routing : fichier principal

---

Emplacement : app/config/routing.yml

Squelette d'une route :

```
lynda_back:
  resource: "@LyndaBackBundle/Resources/config/routing.yml"
  prefix: /

lynda_front:
  resource: "@LyndaFrontBundle/Resources/config/routing.yml"
  prefix: /

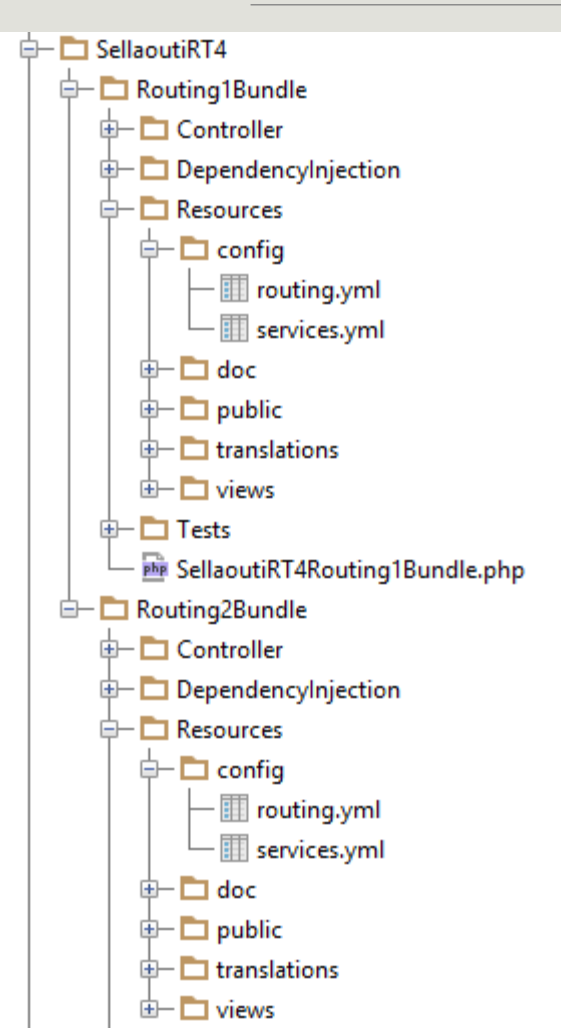
rt4_as:
  resource: "@Rt4AsBundle/Resources/config/routing.yml"
  prefix: /as

rt4_test:
  resource: "@Rt4TestBundle/Resources/config/routing.yml"
  prefix: /test

app:
  resource: "@AppBundle/Controller/"
  type: annotation
```

Référence au fichier routing de LyndaBackBundle

# Externalisation des fichiers de routing



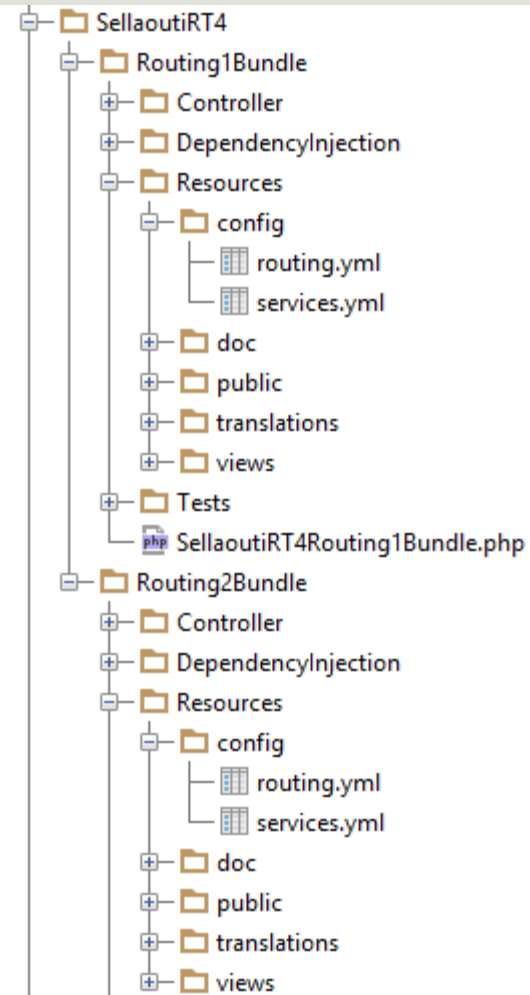
```
app\...\routing.yml x
sellaouti_rt4_routing2:
  resource: "@SellaoutiRT4Routing2Bundle/Resources/config/routing.yml"
  prefix: /
sellaouti_rt4_routing1:
  resource: "@SellaoutiRT4Routing1Bundle/Resources/config/routing.yml"
  prefix: /
app:
  resource: "@AppBundle/Controller/"
  type: annotation
```

```
sellaouti_rt4_routing1_homepage:
  path: /hello/{name}
  defaults: { _controller: SellaoutiRT4Routing1Bundle:Default:index }
```

Routing en utilisant les annotations

```
sellaouti_rt4_routing2_homepage:
  path: /hello/{name}
  defaults: { _controller: SellaoutiRT4Routing2Bundle:Default:index }
```

# Les préfixes



```
sellaouti_rt4_routing2:  
  resource: "@SellaoutiRT4Routing2Bundle/Resources/config/routing.yml"  
  prefix: /root2  
  
sellaouti_rt4_routing1: bundle  
  resource: "@SellaoutiRT4Routing1Bundle/Resources/config/routing.yml"  
  prefix: /root1  
  
app:  
  resource: "@AppBundle/Controller/"  
  type: annotation
```

On ajoute un préfix pour chaque bundle

```
sellaouti_rt4_routing1_homepage:  
  path: /hello/{name}  
  defaults: { _controller: SellaoutiRT4Routing1Bundle:Default:index }
```

```
sellaouti_rt4_routing2_homepage:  
  path: /hello/{name}  
  defaults: { _controller: SellaoutiRT4Routing2Bundle:Default:index }
```

# Préfixe annotation

---

Une annotation Route sur une classe Contrôleur définit un préfixe sur toutes les routes des actions de ce contrôleur

```
1  /**
2   * @Route("/blog")
3   */
4  class PostController extends Controller
5  {
6      /**
7       * @Route("/{id}")
8       */
9      public function showAction($id)
10     {
11     }
12 }
```

<http://symfony.com/fr/doc/current/bundles/SensioFrameworkExtraBundle/annotations/routing.html>



# Squellète d'une route

---

Jusqu'à la version 3, Sensio recommande sans concession l'utilisation du format Yaml au sein des applications. Les bundles développés sont partagés entre deux formats : le XML e le Yaml.

Sensio a décidé de recommander Yaml car celui-ci est plus « *user-friendly* ».

L'utilisation d'un autres format ne changerait rien au fonctionnement de votre application.

Cependant, la documentation de la version 3.4 se focalise essentiellement sur les annotation. Nous allons donc voir ces deux formats.

# Squelette d'une route YAML

YAML

XML

PHP

```
1  blog:
2    path:  /blog/{page}
3    defaults: { _controller: AcmeBlogBundle:Blog:index, page: 1 }
4    requirements:
5      page: \d+
```

YAML

XML

PHP

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2
3  <routes xmlns="http://symfony.com/schema/routing"
4    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5    xsi:schemaLocation="http://symfony.com/schema/routing http://symfony.com/schema/routing/rout
6
7    <route id="blog" path="/blog/{page}">
8      <default key="_controller">AcmeBlogBundle:Blog:index</default>
9      <default key="page">1</default>
10     <requirement key="page">\d+</requirement>
11   </route>
12 </routes>
```

<http://symfony.com/fr/doc/current/book/routing.html>

# Squelette d'une route PHP

---

YAML

XML

PHP

```
1 use Symfony\Component\Routing\RouteCollection;
2 use Symfony\Component\Routing\Route;
3
4 $collection = new RouteCollection();
5 $collection->add('blog', new Route('/blog/{page}', array(
6     '_controller' => 'AcmeBlogBundle:Blog:index',
7     'page' => 1,
8 ), array(
9     'page' => '\d+',
10 )));
11
12 return $collection;
```

<http://symfony.com/fr/doc/current/book/routing.html>

# Squelette d'une route Annotation

---

```
class DefaultController extends Controller
{
    /**
     * @Route("/app/example", name="homepage")
     */
    public function indexAction()
    {
        return $this->render('default/index.html.twig');
    }
}
```

Exemple de routing utilisant les annotations

<http://symfony.com/fr/doc/current/book/routing.html>

# Paramétrage de la route : Yaml

---

Nous pouvons ajouter autant de paramètre dans la route

Les séparateurs entre les paramètres sont ‘/’ et ‘.’

## Exemple

front\_article:

```
path:    /article/{year}/{langue}/{slug}.{format}
defaults: { _controller: LyndaFrontBundle:Default:showArticle }
```

Ici l'url doit contenir l'année de l'article {year}, la langue de l'article {langue} les mots clefs {slug} ainsi que le format {format}

## Astuce

Pour la langue utiliser la variable fourni par symfony2 [\\_locale](#) permettant ainsi de modifier en même temps la locale de la page

Pour le format utiliser le [\\_format](#) qui modifie le content type envoyé au navigateur

# Paramétrage de la route : Annotation

---

```
/**
 * @Route("/hello/{name}", name="front_homepage")
 */
public function testAction($name) {

}

/**
 * @Route("/article/{year}/{_locale}/{slug}.{_format}", name="front_article")
 */
public function showArticleAction($year, $_locale, $slug, $_format) {

}
```

# Paramétrage de la route : variables spécifiques

---

# Assesment

---

Quels sont les URL valides si on a cette route :

```
/**
 * @Route("/article/{year}/{_locale}/{slug}.{_format}", name="front_article")
 */
public function showArticleAction($year, $_locale, $slug, $_format) {

}
```

1- <http://127.0.0.1:8000/article/2005/fr/page.twig-symfony-routing.twig.html>

2- <http://127.0.0.1:8000/article.a/2005/fr/page.twig-symfony-routing.twig.html>

3- <http://127.0.0.1:8000/article/2005/fr/twig-symfony/-routing.twig.html>



# Paramétrage de la route : valeurs par défaut Yaml

---

Afin d'avoir des valeurs par défauts nous utilisons la syntaxe suivante :

## Syntaxe

front\_article:

```
path:    /article/{year}/{_locale}/{slug}.{_format}  
defaults: { _controller: LyndaFrontBundle:Default:index, variable:valeurParDefaut }
```

## Exemple

front\_article:

```
path:    /article/{year}/{langue}/{slug}.{format}  
defaults: { _controller: LyndaFrontBundle:Default:index, _format: html }
```

Important : Seul les paramètres optionnels terminant la route pourront être absents de l'URL

# Paramétrage de la route : valeurs par défaut Annotations

---

En utilisant les annotations, nous ajoutons un champ `defaults` qui contiendra les valeurs par défaut.

```
/**
 * @Route(
 *     "/article/{year}/{_locale}/{slug}.{_format}",
 *     name="front_article",
 *     defaults={"_format":"html", "slug":"Symfony"}
 * )
 */
public function showArticleAction($year, $_locale, $slug, $_format) {
    var_dump($year, $_locale, $slug, $_format);
    die();
}
```

Maintenant l'URL suivante devient correcte : <http://127.0.0.1:8000/article/2005/fr> et les variables `slug` et `_format` prendront leur valeur par défaut

# Paramétrage de la route : Requirements Yaml

---

Pour contrôler les paramètres de la route et ne pas y accepter n'importe quel valeur, on utilise les [requirements](#). Elles permettent de restreindre la valeur du paramètre à un certain type ou une liste de valeurs. Par exemple si votre site ne contient que la version française ou anglaise la variable `_locale` ne peut être que `fr` ou `en`.

## Syntaxe

`front_article:`

```
path:    /article/{year}/{_locale}/{slug}.{_format}
defaults: { _controller: LyndaFrontBundle:Default:index, variable:valeurParDefaut }
requirements :
    nomParamètre: condition
```

## Exemple

`front_article:`

```
path:    /article/{year}/{langue}/{slug}.{format}
defaults: { _controller: LyndaFrontBundle:Default:index, _format: html}
requirements:
    year:\d{4} // l'année sera obligatoirement un chiffre composé de 4 entiers
    langue: fr|en // ici la langue ne pourra être qu'anglais ou français
```

# Paramétrage de la route (7)

---

`\d` équivalente à `\d{1}`

`\d+` ensemble d'entiers

Gestion des méthodes http :

`_method: POST` ceci signifie que la route ne s'exécutera qu'à travers la méthode POST

options les plus utilisées :

- GET
- POST

On peut aussi mettre une expression régulière.

# Paramétrage de la route : Requirements Annotation

---

En utilisant les annotations, nous ajoutons un champ `requirements` qui contiendra les différentes contraintes.

```
/**
 * @Route(
 *     "/article/{year}/{_locale}/{slug}.{_format}",
 *     name="front_article",
 *     defaults={"_format":"html", "slug":"Symfony"},
 *     requirements={
 *         "_locale" : "fr|en",
 *         "year" : "\d{4}"
 *     }
 * )
 */
public function showArticleAction($year, $_locale, $slug, $_format) {

}
```

# Ordre de traitement des routes (1)

---

Le traitement des routes se fait de la première route vers la dernière.

Attention à l'ordre d'écriture de vos routes.

Exemple

front:

```
path: /front/{page}
```

```
defaults: { _controller: FrontBundle:Default:index }
```

front\_pages:

```
path: /front/{Keys}
```

```
defaults: { _controller: FrontBundle:Default:show }
```

Comment accéder au path front\_pages ? Quel est le problème avec ces 2 routes

# Ordre de traitement des routes (2)

---

front:

path: /front/{page}

defaults: { \_controller: FrontBundle:Default:index }

front\_pages:

path: /front/{Keys}

defaults: { \_controller: FrontBundle:Default:show }

Les deux routes sont de la forme /front/\* donc n'importe quel route de cette forme sera automatiquement transféré au Default controller pour exécuter l'index action.

Proposer une solution

# Ordre de traitement des routes (3)

---

front:

path: /front/{page}

defaults: { \_controller: FrontBundle:Default:index }

requirements:

page: \d+

front\_pages:

path: /front/{Keys}

defaults: { \_controller: FrontBundle:Default:show }

Tester le fonctionnement des routes suivantes : /front/1234

/front/test-ordre-de-routeing



# Ordre de traitement des routes (4)

---

Que se passe t-il si on inverse l'ordre des deux routes ? Est-ce que la solution persiste?

front\_pages:

path: /front/{Keys}

defaults: { \_controller: FrontBundle:Default:show }

front:

path: /front/{page}

defaults: { \_controller: FrontBundle:Default:index }

requirements:

page: \d+

Tester le fonctionnement des routes suivantes : /front/1234

/front/test-ordre-de-routeing

# Ordre de traitement des routes (5)

---

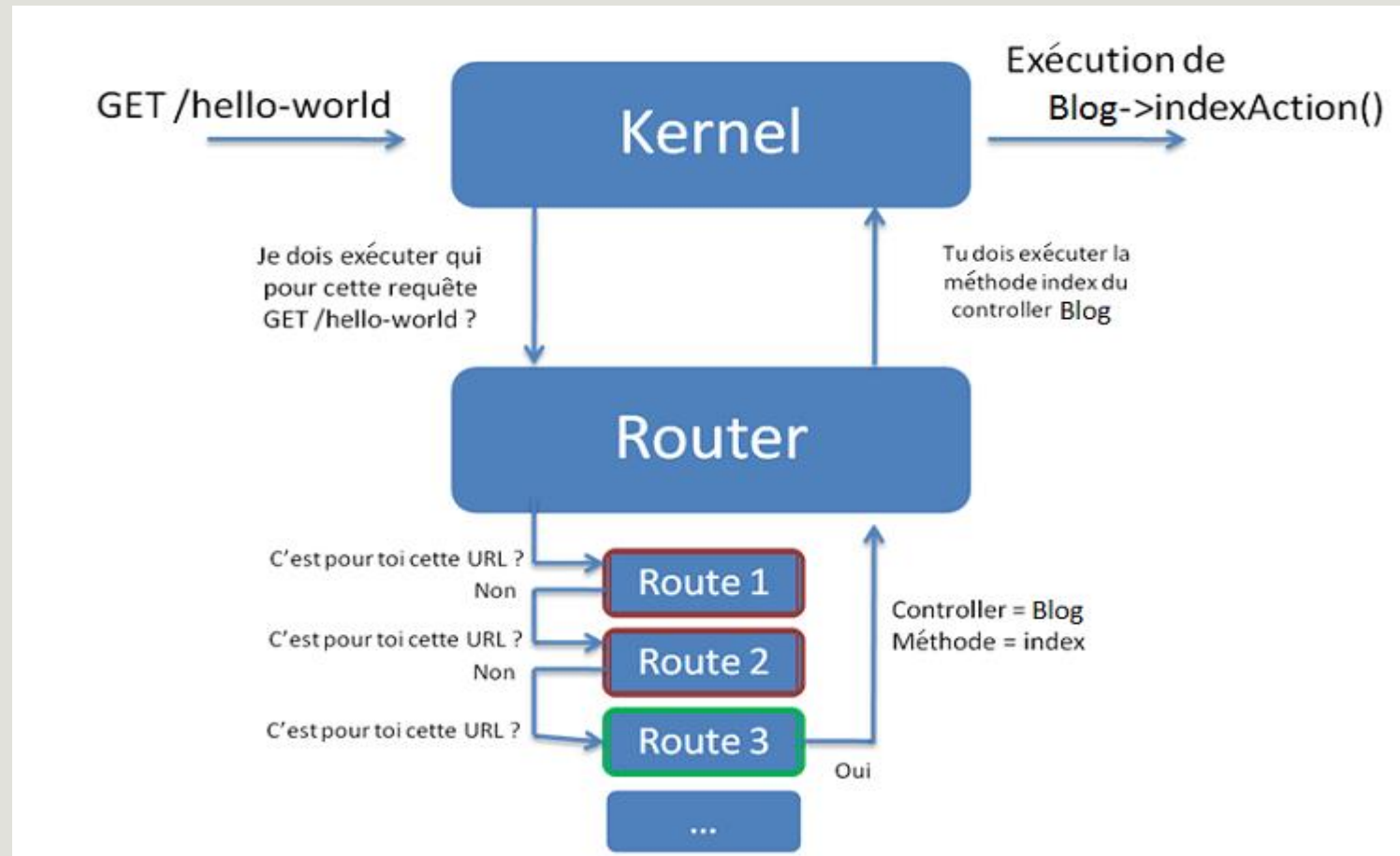
Les Routes précédentes Gagnent toujours

L'ordre des routes est très important.

En utilisant un ordre clair et intelligent, vous pouvez accomplir tout ce que vous voulez.

<http://symfony.com/fr/doc/current/book/routing.html>

# Ordre de traitement des routes (6)



# Débogage des routes

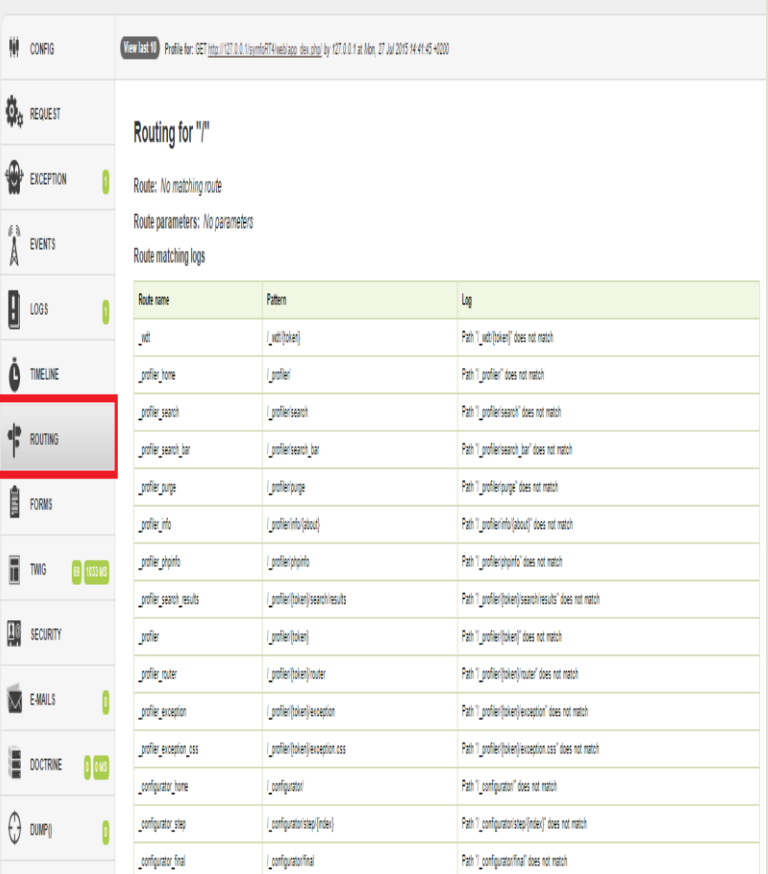
Afin de visualiser l'ensemble des routes deux solutions sont possibles :

Par ligne de commande : `php app/console router:debug`

En utilisant la debug toolbar

On peut aussi vérifier quelle route correspond à une URL spécifique

`Php app/console router:match URI`



View last 10 Profile for: GET [https://127.0.0.1/symfony/webapp\\_des.php](https://127.0.0.1/symfony/webapp_des.php) by 127.0.0.1 at Mon, 17 Jul 2015 14:41:45 +0200

**ROUTING**

Routing for "/"

Router: No matching route

Route parameters: No parameters

Route matching logs

Route name	Pattern	Log
_wdt	/{_wdt{token}}	Path "{_wdt{token}}" does not match
_profile_home	/{_profile{}}	Path "{_profile}" does not match
_profile_search	/{_profilesearch}	Path "{_profilesearch}" does not match
_profile_search_bar	/{_profilesearch_bar}	Path "{_profilesearch_bar}" does not match
_profile_purge	/{_profilepurge}	Path "{_profilepurge}" does not match
_profile_info	/{_profileinfo{about}}	Path "{_profileinfo{about}}" does not match
_profile_photo	/{_profilephoto}	Path "{_profilephoto}" does not match
_profile_search_results	/{_profile{token}/search/results}	Path "{_profile{token}/search/results}" does not match
_profile	/{_profile{token}}	Path "{_profile{token}}" does not match
_profile_router	/{_profile{token}/router}	Path "{_profile{token}/router}" does not match
_profile_exception	/{_profile{token}/exception}	Path "{_profile{token}/exception}" does not match
_profile_exception_css	/{_profile{token}/exception.css}	Path "{_profile{token}/exception.css}" does not match
_configurator_home	/{_configurator/}	Path "{_configurator}" does not match
_configurator_step	/{_configurator/step/{index}}	Path "{_configurator/step/{index}}" does not match
_configurator_final	/{_configuratorfinal}	Path "{_configuratorfinal}" does not match