TP1 : Contrôleurs et Vues

L'objectif principal de ce premier TP est d'afficher des pages en Twig.

1. Installation des packages

- Vous avez besoin d'avoir plusieurs packages, il faut vérifier s'ils sont présents déjà ou non dans le « squelette » de Symfony dans le fichier composer.json ; sinon il faudra les télécharger via composer.

D'abord, modifier dans le fichier « composer.json » l'option « allow-contrib » sous « extra » puis « symfony » à true : cela nous permet de récupérer des packages issus de collaborations d'utilisateurs.

- A lancer dans le terminal :
  - apache-pack: « ./composer req symfony/apache-pack »
  - framework-extra-bundle : permet de travailler avec des annotations.
  - maker-bundle : aide à créer des controleurs, des entités… de façon automatisée.
  - profiler-pack : permet d'afficher la barre de debug.
  - twig-bundle : pour utiliser le moteur de templates.

2. Création de votre première page

A/ Première Mission : création du contrôleur

Pour cela, nous allons utiliser le MakerBundle, qui permet de générer du code automatiquement. Nous allons créer une page d'accueil qui affiche un certain nombre de données, d'abord de test, puis à terme une liste de concerts.

D'abord, créons notre contrôleur.

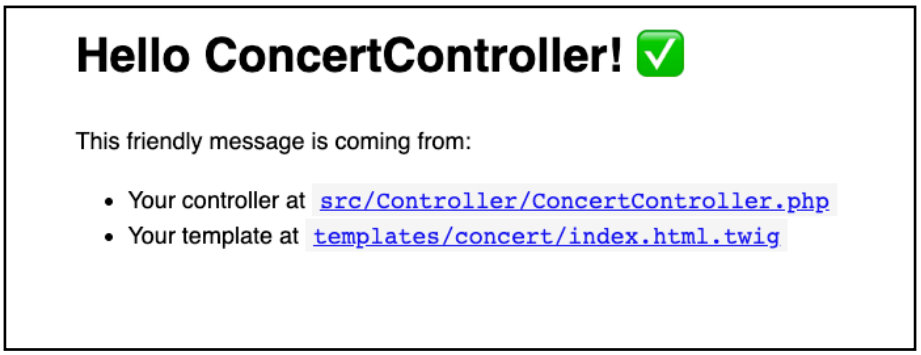
- Dans votre console, saisissez :

```
aka@Air-de-Aka concertProject % php bin/console make:controller

 Choose a name for your controller class (e.g. GentleKangarooController):
 > ConcertController

 created: src/Controller/ConcertController.php
 created: templates/concert/index.html.twig

  Success!

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

Et suivons le conseil : « open your new controller class ! ».

B/ Routing via annotations

- Symfony a crée pas mal de code. Regardez le bloc commenté au-dessus de la fonction index(): ce sont des annotations : @Route(). Ce qui est indiqué entre parenthèses est la route à taper.
- Ouvrez votre navigateur et tapez « http://localhost:8000/concert » soit l'adresse indiquée par le routing.



C/ Twig

- Comment ce contenu est-il affiché ? Via un template Twig : allez voir dans le dossier « Templates ». Nous retrouvons ici le principe d'héritage de templates (si vous avez fait de la Programmation Orientée Objet, c'est la même chose !).
- Héritage via : {% extends 'base.html.twig' %}
- Observer les structures affichées et trouvez d'où vient le 'controller_name'. Essayez avec d'autres paramètres / contenus et actualisez l'affichage navigateur : je veux qu'il s'affiche 'Hello Licence APIDAE' à la place, sans modifier le template.

- Pour mieux visualiser l'héritage, dans le block body de votre base.html.twig, ajoutez un <h1>:

```
{% block body %}
    <h1>Hello</h1>
{% endblock %}
```

- Actualisez la page du navigateur : qu'est-ce qui s'affiche et pourquoi ?

En fait, lorsque vous écrivez dans un block Twig, vous écrasez le contenu du même bloc du fichier parent.
Résultat : Twig va nous faire gagner beaucoup de temps ! Par exemple, imaginer la barre de navigation…. Pas besoin de la dupliquer sur chaque page du site, vous n'aurez qu'à créer un fichier *navbar.html.twig* qui contiendra le code de votre navbar, et vous l'appellerez ensuite sur toute vos pages.
Donc un seul fichier à modifier, pas d'oubli(s) et du coup gain de temps.

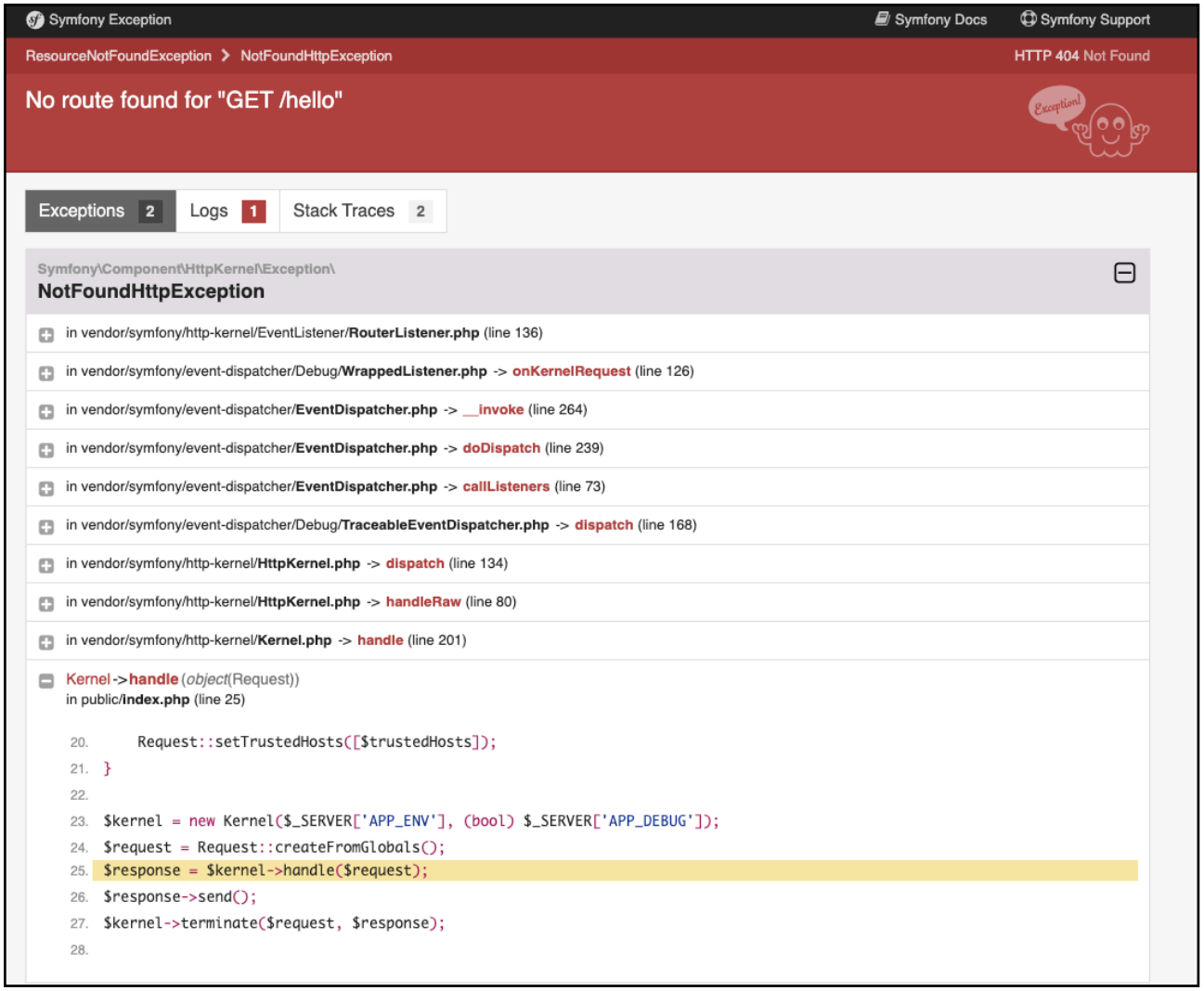
3. Créer une deuxième page et son template

Mission : créer une route « /list » qui renvoie une liste de concerts, affichés en Twig, dans le ConcertController.

- Pensez à aller voir la documentation de Twig, qui vous permettra de faire des fonctions dans le fichier Twig…
- Je vous laisse le choix des groupes, que vous pourrez passez en paramètre.

Bon courage !

En bonus, une erreur que vous rencontrerez souvent, et qui est affiché par le debug Symfony. Je vous conseille fortement de prendre le temps de lire ce qui est affiché, par seulement le texte de l'erreur. Vous avez plusieurs onglets (Exceptions, Logs, Stack Traces) qui vous permettent de suivre le chemin qu'a suivi Symfony avant de rencontrer une erreur.


Symfony Exception
Symfony Docs
Symfony Support
ResourceNotFoundException > NotFoundHttpException
HTTP 404 Not Found
No route found for "GET /hello"
Exceptions 2
Logs 1
Stack Traces 2
Symfony\Component\HttpKernel\Exception\
NotFoundHttpException
in vendor/symfony/http-kernel/EventListener/RouterListener.php (line 136)
in vendor/symfony/event-dispatcher/Debug/WrappedListener.php -> onKernelRequest (line 126)
in vendor/symfony/event-dispatcher/EventDispatcher.php -> __invoke (line 264)
in vendor/symfony/event-dispatcher/EventDispatcher.php -> doDispatch (line 239)
in vendor/symfony/event-dispatcher/EventDispatcher.php -> callListeners (line 73)
in vendor/symfony/event-dispatcher/Debug/TraceableEventDispatcher.php -> dispatch (line 168)
in vendor/symfony/http-kernel/HttpKernel.php -> dispatch (line 134)
in vendor/symfony/http-kernel/HttpKernel.php -> handleRaw (line 80)
in vendor/symfony/http-kernel/Kernel.php -> handle (line 201)
Kernel->handle(object(Request))
in public/index.php (line 25)
20. Request::setTrustedHosts([$trustedHosts]);
21. }
22.
23. $kernel = new Kernel($_SERVER['APP_ENV'], (bool) $_SERVER['APP_DEBUG']);
24. $request = Request::createFromGlobals();
25. $response = $kernel->handle($request);
26. $response->send();
27. $kernel->terminate($request, $response);
28.