Résumé de tout ce qu'il faut savoir AVANT de commencer (Symfony, Contrôleur, Routage, Twig)

1. Symfony, c'est quoi ?

- **Un framework PHP pour faire des sites web modernes plus vite, plus proprement.**
- **Il organise le code en MVC : Modèle (données), Vue (affichage), Contrôleur (chef d'orchestre).**

2. Un contrôleur, c'est quoi ?

- **Un fichier PHP (dans `src/Controller/`), une "classe".**
- **Il reçoit la demande d'une page, décide quoi faire, et prépare la réponse.**
- **Toujours hériter de AbstractController pour avoir tous les outils Symfony.**

3. Le routage, c'est quoi ?

- **Le système qui fait le lien entre une URL (adresse web) et une méthode de contrôleur.**
- **En Symfony moderne, on déclare une route avec l'attribut #[Route(...)] juste au-dessus de la méthode.**

- **Les paramètres dynamiques se mettent entre accolades dans l'URL (`/page/{id}`).**

4. Twig, c'est quoi ?

- **Un langage de templates (modèles HTML) utilisé pour afficher dynamiquement des variables envoyées par le contrôleur.**

- **On affiche une variable avec `{{ variable }}`.**

- **Pour faire des boucles (afficher une liste) : `{% for x in liste %}`**

- **Pour des conditions : `{% if condition %}`**

5. Où ranger chaque fichier ?

- **Contrôleurs : `src/Controller/`**

- **Vues Twig : `templates/` (et sous-dossiers selon tes besoins)**

- **Images et fichiers publics : `public/`**

🌟 Qu’est-ce que Twig ? (La base pour tout comprendre)

1. Qu’est-ce que Twig ?

- **Twig** est un **moteur de templates** (modèles de pages) utilisé par Symfony.
- Il sert à **générer des pages HTML** facilement, en mélangeant du texte, du HTML, et des variables envoyées depuis ton contrôleur PHP.
- **Exemple** :
  Tu veux afficher "Bonjour, Paul !" sur une page, mais tu ne connais le prénom qu'au moment où l'utilisateur visite la page.
  → Twig va insérer la variable {{ name }} à la place du prénom.

2. À quoi ça sert Twig ?

- **Afficher dynamiquement** des informations : variables, listes, conditions.
- **Eviter d’écrire du PHP** directement dans tes pages HTML (plus propre, plus sécurisé).
- Permet d’utiliser des **boucles**, des **filtres** (pour mettre en majuscule, trier, compter…), des **conditions** (if/else).

3. Comment ça marche ?

- Le **contrôleur** (ton code PHP) envoie des variables à Twig.
- Twig **affiche** ces variables dans la page HTML, là où tu mets `{{ variable }}`.

4. Exemple très simple

Dans le contrôleur :

php
CopierModifier
```
public function hello($name) {
    return $this->render('hello.html.twig', ['name' => $name]);
}
```

Dans Twig (`templates/hello.html.twig`) :

twig
CopierModifier
```
<h1>Bonjour, {{ name }} !</h1>
```

→ Si `$name = 'Paul'`, la page affichera "Bonjour, Paul !"

🟩 Exercice 1 : Twig et affichage d'une variable

---

1. Créer un nouveau contrôleur « AuthorController » sous le répertoire Controller

Qu'est-ce qu'on te demande ?
Créer un **fichier PHP** pour gérer les auteurs.

Ce qu'il faut savoir avant

- Les **contrôleurs** sont rangés dans `src/Controller`.
- **Son nom** : il doit finir par "Controller".

Réponse
bash
CopierModifier
```
php bin/console make:controller AuthorController
```

OU à la main :
Créer un fichier `src/Controller/AuthorController.php` et écrire :

php
CopierModifier

```
namespace App\Controller;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
class AuthorController extends AbstractController {}
```

Explication détaillée

- C'est comme un "chef d'orchestre" pour tout ce qui concerne les auteurs : il va répondre aux demandes (par ex, afficher un auteur).
- Symfony "regarde" ce dossier pour trouver tes contrôleurs.

Astuce

- **Toujours dans src/Controller/**, et le nom finit par Controller !

---

2. Créer une nouvelle méthode « showAuthor » qui prend en paramètre la variable « name »

Qu'est-ce qu'on te demande ?
Une fonction dans AuthorController qui reçoit un nom d'auteur.

Ce qu'il faut savoir avant

- **Méthode** = fonction à l'intérieur de la classe contrôleur.
- **Paramètre** = ce que tu veux recevoir dans l'URL (par exemple, `/author/Taha` → `Taha`).

Réponse
php
CopierModifier

```
public function showAuthor(string $name) {
    return $this->render('author/show.html.twig', ['name' => $name]);
```

```
}
```

Explication détaillée

- La méthode attend une **variable** `$name`.
- Elle “pousse” cette variable vers Twig, dans la page `show.html.twig`.

Astuce

- **Toujours envoyer les infos à Twig sous forme de tableau associatif** : `['clé' => $valeur]`

---

3. Créer une nouvelle route de la méthode « showAuthor »

Qu’est-ce qu’on te demande ?
Associer une adresse (URL) à la méthode pour qu’on puisse l’appeler dans le navigateur.

Ce qu’il faut savoir avant

- Une **route** relie une URL à une méthode d’un contrôleur.
- Les **paramètres** dans l’URL se mettent entre accolades.

Réponse
php
CopierModifier

```
use Symfony\Component\Routing\Attribute\Route;

#[Route('/author/{name}', name: 'show_author')]
public function showAuthor(string $name) { ... }
```

Explication détaillée

- Quand tu tapes `/author/Taha`, Symfony appelle `showAuthor('Taha')`.

- `name: 'show_author'` sert à faire des liens internes (important plus tard).

Astuce

- **Syntaxe : #[Route('/chemin/{param}', name: 'nom_de_route')]**

---

4. Ajouter une nouvelle vue « show.html.twig » pour afficher la variable « name »

Qu'est-ce qu'on te demande ?
Créer un fichier pour afficher l'auteur reçu.

Ce qu'il faut savoir avant

- Les vues Twig sont dans le dossier **`templates/author/`**
- Pour afficher une variable dans Twig, il faut écrire **`{{ variable }}`**

Réponse

Créer `templates/author/show.html.twig` :

twig
CopierModifier
```
<h1>Auteur : {{ name }}</h1>
```

Explication détaillée

- `{{ name }}` va afficher la valeur passée depuis le contrôleur.
- Si tu vas sur `/author/Paul`, tu verras "Auteur : Paul".

Astuce

- **Toujours utiliser {{ variable }} dans Twig pour afficher une donnée**

---

5. Tester le rendu de la page

Qu'est-ce qu'on te demande ?
Vérifier si tout marche bien dans le navigateur.

Ce qu'il faut savoir avant

- **Lancer le serveur Symfony** avec `symfony serve`
- **Accéder à l'URL** dans le navigateur (par exemple, `/author/Paul`)

Réponse

Va sur `http://localhost:8000/author/Paul`
Tu dois voir : **Auteur : Paul**

Explication détaillée

- Tu viens de faire passer une donnée du navigateur → au contrôleur → à la vue Twig → affichage dans le navigateur !

Astuce

- **Teste toujours ton code dans le navigateur après chaque modification**

🟦 Exercice 2 : Tableaux, filtres, conditions en Twig

---

1. Créer une nouvelle méthode listAuthors qui définit la liste des auteurs

Qu'est-ce qu'on te demande ?
Faire une méthode qui crée un tableau d'auteurs, et l'envoie à Twig.

Ce qu'il faut savoir avant

- Un **tableau associatif** en PHP = une liste d'éléments sous forme clé/valeur (comme un mini-carnet d'adresses)

- Il faut **rendre** ce tableau à Twig.

Réponse

php
CopierModifier

```
#[Route('/authors', name: 'list_authors')]
public function listAuthors() {
    $authors = [
        ['id' => 1, 'picture' => '/images/Victor-Hugo.jpg', 'username'
=> 'Victor Hugo', 'email' => 'victor.hugo@gmail.com', 'nb_books' =>
100],
        ['id' => 2, 'picture' => '/images/william-shakespeare.jpg',
'username' => 'William Shakespeare', 'email' =>
'william.shakespeare@gmail.com', 'nb_books' => 200],
        ['id' => 3, 'picture' => '/images/Taha_Hussein.jpg',
'username' => 'Taha Hussein', 'email' => 'taha.hussein@gmail.com',
'nb_books' => 300],
    ];
    return $this->render('author/list.html.twig', ['authors' =>
$authors]);
}
```

Explication détaillée

- Tu prépares une "liste" d'auteurs (un tableau de tableaux).

- Tu l'envoies à Twig sous le nom `authors`.

Astuce

- **Toujours nommer les variables envoyées à Twig de façon claire, et utiliser ['clé' => valeur]**

---

2. Créer le dossier images sous public et télécharger les photos

Qu'est-ce qu'on te demande ?
Avoir des images accessibles depuis le web.

Ce qu'il faut savoir avant

- Le dossier `public/` sert à stocker ce qui doit être visible depuis le navigateur.

Réponse

Dans le dossier `public/`, créer un dossier `images/` et ajouter les trois photos données.

Explication détaillée

- Ces images sont appelées via leur chemin (ex : `/images/Victor-Hugo.jpg`) dans Twig.

Astuce

- **Tout ce qui est à voir sur le site (images, css, js) va dans le dossier `public/`**

---

3. Créer l'interface list.html.twig pour afficher la liste des auteurs

Qu'est-ce qu'on te demande ?
Afficher la liste envoyée à Twig, dans une page HTML.

Ce qu'il faut savoir avant

- En Twig, on parcourt un tableau avec une boucle `{% for ... in ... %}`

Réponse

Dans `templates/author/list.html.twig` :

twig
CopierModifier

```
{% for author in authors %}
    <div>
```

```
        <img src="{{ author.picture }}" alt="{{ author.username }}"
width="100">
        <h2>{{ author.username }}</h2>
        <p>Email : {{ author.email }}</p>
        <p>Livres publiés : {{ author.nb_books }}</p>
    </div>
{% endfor %}
```

Explication détaillée

- Chaque auteur est affiché avec son image, son nom, son email, et son nombre de livres.

Astuce

- **Toujours pour parcourir un tableau en Twig : {% for element in liste %} ... {% endfor %}**

💡 **Maintenant, continuons avec chaque question de l'Exercice 2

---

4. Ajouter une condition dans le cas où le tableau n'est pas défini et/ou vide

Qu'est-ce qu'on te demande ?
Afficher un message spécial si la liste des auteurs est vide ou n'existe pas.

Ce qu'il faut savoir avant

- En Twig, on fait une condition avec **`{% if ... %}`**.
- Un tableau vide est "faux" (`not authors` est vrai si la liste est vide ou n'existe pas).

Réponse

Dans `list.html.twig` :

twig
CopierModifier
```
{% if authors is not defined or authors is empty %}
    <p>Aucun auteur trouvé.</p>
{% else %}
    <!-- ici la boucle for pour afficher les auteurs -->
{% endif %}
```

Explication détaillée

- `is not defined` vérifie si la variable existe.
- `is empty` vérifie si la liste est vide.
- Si la liste existe et n'est pas vide, on affiche les auteurs.

Astuce

- **Toujours mettre une condition de secours dans Twig pour éviter les erreurs d'affichage**

---

5. Afficher les noms des auteurs en majuscule

Qu'est-ce qu'on te demande ?
Afficher chaque nom d'auteur en toutes lettres majuscules.

Ce qu'il faut savoir avant

- Twig a des **filtres** pour transformer les variables, par exemple `|upper` pour mettre en majuscules.

Réponse
twig
CopierModifier
```
<h2>{{ author.username|upper }}</h2>
```

Explication détaillée

- `|upper` transforme la variable en majuscule (“Taha Hussein” → “TAHA HUSSEIN”).

Astuce

- **Les filtres Twig s’utilisent avec `|`, et il en existe beaucoup (`|lower`, `|capitalize`, `|date`, etc.)**

---

6. Afficher le nombre des auteurs

Qu’est-ce qu’on te demande ?
Afficher combien d’auteurs il y a dans la liste.

Ce qu’il faut savoir avant

- Twig a un filtre **`length`** pour compter le nombre d’éléments d’une liste.

Réponse
twig
CopierModifier

```
<p>Nombre d'auteurs : {{ authors|length }}</p>
```

Explication détaillée

- `|length` compte combien il y a d’éléments dans le tableau `authors`.

Astuce

- **Pour compter dans Twig, toujours penser à `|length`**

---

7. Afficher le nombre total des livres de tous les auteurs

Qu’est-ce qu’on te demande ?
Additionner le champ `nb_books` de chaque auteur et afficher le total.

Ce qu'il faut savoir avant

- Twig ne sait pas additionner directement toute une colonne, donc il faut faire une petite boucle manuelle.

Réponse

Dans Twig, avant d'afficher :

twig
CopierModifier

```
{% set totalBooks = 0 %}
{% for author in authors %}
    {% set totalBooks = totalBooks + author.nb_books %}
{% endfor %}
<p>Nombre total de livres : {{ totalBooks }}</p>
```

Explication détaillée

- On crée une variable `totalBooks`, on l'incrémente dans la boucle.
- Après la boucle, on affiche le total.

Astuce

- **Toujours faire un `set` pour initialiser la variable avant la boucle si tu veux additionner/sommer en Twig**

---

8. Ajouter un lien « details » devant chaque auteur

Qu'est-ce qu'on te demande ?
Mettre un bouton/lien pour accéder à la fiche de chaque auteur.

Ce qu'il faut savoir avant

- Pour créer un lien vers une route Symfony avec paramètre, on utilise la fonction Twig `path('nom_de_route', {param: valeur})`.

Réponse
twig
CopierModifier
```
<a href="{{ path('author_details', {'id': author.id}) }}">Details</a>
```

Dans ce cas, il faut créer une route appelée `author_details` qui prend `id` en paramètre (voir prochaine question).

Explication détaillée

- Le lien va pointer sur l'URL `/author/details/1` (si l'auteur a l'id 1, par exemple).

Astuce

- **Toujours nommer tes routes dans les contrôleurs pour pouvoir les utiliser avec `path()` dans Twig**

---

9. Créer une nouvelle méthode « authorDetails » qui prend en paramètre la variable « id »

Qu'est-ce qu'on te demande ?
Ajouter dans le contrôleur une fonction qui va recevoir un id d'auteur.

Ce qu'il faut savoir avant

- Le paramètre dans la route se met entre accolades, la méthode prend le même nom en paramètre.

Réponse
php
CopierModifier
```
#[Route('/author/details/{id}', name: 'author_details')]
public function authorDetails($id) {
    // code pour chercher l'auteur par id
}
```

Explication détaillée

- Quand tu visites `/author/details/2`, la méthode reçoit `$id = 2`.

Astuce

- **Toujours faire correspondre le nom entre accolades dans la route et le nom de la variable en paramètre de la méthode**

---

10. Créer une nouvelle route de la méthode « authorDetails »

Qu'est-ce qu'on te demande ?
Associer une URL à cette méthode (fait juste avant avec l'attribut Route).

Ce qu'il faut savoir avant

- C'est exactement la ligne avec #[Route(...)] vue juste au-dessus.

Réponse

Déjà fait :

php
CopierModifier
```
#[Route('/author/details/{id}', name: 'author_details')]
```

Explication détaillée

- Permet d'accéder à la fiche détaillée d'un auteur en passant son id dans l'URL.

Astuce

- **Nommer tes routes de façon logique, pour les retrouver facilement**

---

11. En cliquant sur le lien « details », l’utilisateur sera redirigé vers une nouvelle vue « showAuthor.html.twig » affichant le nom, photo, email et le nombre des livres de l’auteur

Qu’est-ce qu’on te demande ?
Quand on clique sur "details", afficher la fiche complète de l’auteur (nom, photo, email, nb livres).

Ce qu’il faut savoir avant

- Il faut **retrouver l’auteur** correspondant à l’id reçu (soit en base, soit dans le tableau d’auteurs si tu restes simple).

- Il faut envoyer ces infos à la vue Twig, puis les afficher dans la page.

Réponse

Dans AuthorController.php :

php
CopierModifier

```
#[Route('/author/details/{id}', name: 'author_details')]
public function authorDetails($id) {
    $authors = [...]; // même tableau qu'avant, ou bien le chercher ailleurs
    $author = null;
    foreach ($authors as $a) {
        if ($a['id'] == $id) {
            $author = $a;
            break;
        }
    }
    return $this->render('author/showAuthor.html.twig', ['author' => $author]);
}
```

Dans `templates/author/showAuthor.html.twig` :

twig
CopierModifier

```
{% if author %}
    <img src="{{ author.picture }}" alt="{{ author.username }}"
width="100">
    <h2>{{ author.username }}</h2>
    <p>Email : {{ author.email }}</p>
    <p>Livres publiés : {{ author.nb_books }}</p>
{% else %}
    <p>Auteur non trouvé.</p>
{% endif %}
```

Explication détaillée

- On cherche l'auteur avec l'id voulu, on envoie ses infos à Twig.
- Twig les affiche dans une page dédiée.

Astuce

- **Toujours vérifier si la variable existe dans Twig avant d'afficher pour éviter les erreurs !**

À apprendre absolument et à relire souvent

- Contrôleur = classe PHP, hérite de AbstractController, méthodes publiques
- Routage = #[Route('/chemin', name: 'route_name')]
- Twig = `{{ variable }}` pour afficher, `{% for ... %}` pour les boucles, `{% if ... %}` pour les conditions
- Pour les listes : boucle for, filtre `|upper`, `|length`
- Pour les routes dynamiques : accolade dans l'URL + même nom en paramètre

- Pour les liens Twig : `path('nom_route', {'param': valeur})`