Web - TP Symfony n^o 4
G. Subrenat, R. Zrour

Licence Informatique L3 - 2022-2023
UFR SFA - Université de Poitiers

Twig

Table des matières

Le code du TP est disponible à : https://gitlab.com/subrenat/l3-web-2022-23
dans la branche *b-TP4* : https://gitlab.com/subrenat/l3-web-2022-23/-/tree/b-TP4

1 Résumé

Ce TP est consacré aux vues (i.e. templates). Il y a deux aspects :

- organisation hiérarchique (avec notion d'héritage) des templates pour éviter toute duplication de code.
- apprentissage de la syntaxe Twig.

Vous êtes en mode recherche, autrement ce document n'est pas auto-suffisant et il faudra trouver l'information sur différents sites :

- le site de SensioLabs est le site de référence et doit être privilégié.
- des tutoriels peuvent être une autre voie d'exploration.

Liens utiles pour ce TP :

- https://symfony.com/doc/6.2/templates.html (pour la version imposée dans le TP)
- https://symfony.com/doc/current/templates.html (pour la dernière version [1])
- https://openclassrooms.com/fr/courses/5489656-construisez-un-site-web-a-l-aide-du-framework-symfony-5 (non testé et une version antérieure à celle du TP)
- https://www.comment-devenir-developpeur.com/symfony-6-et-symfony-6-1 (non testé)

2 Héritage de templates

Dans cette section nous ferons une architecture à deux niveaux.

Voici les liens vers la documentation officielle :
https://symfony.com/doc/current/templates.html#template-inheritance-and-layouts
https://symfony.com/doc/6.2/templates.html#template-inheritance-and-layouts

2.1 Création du contrôleur de test

Travail à effectuer

Créez dans le répertoire *Sandbox* un nouveau contrôleur nommé *Twig* avec pour l'instant aucun répertoire template associé (option *--no-template*).

N'oubliez pas de mettre un préfixe (*/sandbox/twig* semble adéquat) pour toutes les routes du contrôleur.

Vous avez une proposition de code à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/736c8bf4bb06b19c6e2ebfc9cdb4cafa7dc48d85/src/Controller/Sandbox/TwigController.php

2.2 Préambule

Travail à effectuer

Lisez les explications pour comprendre le principe de l'héritage de templates.

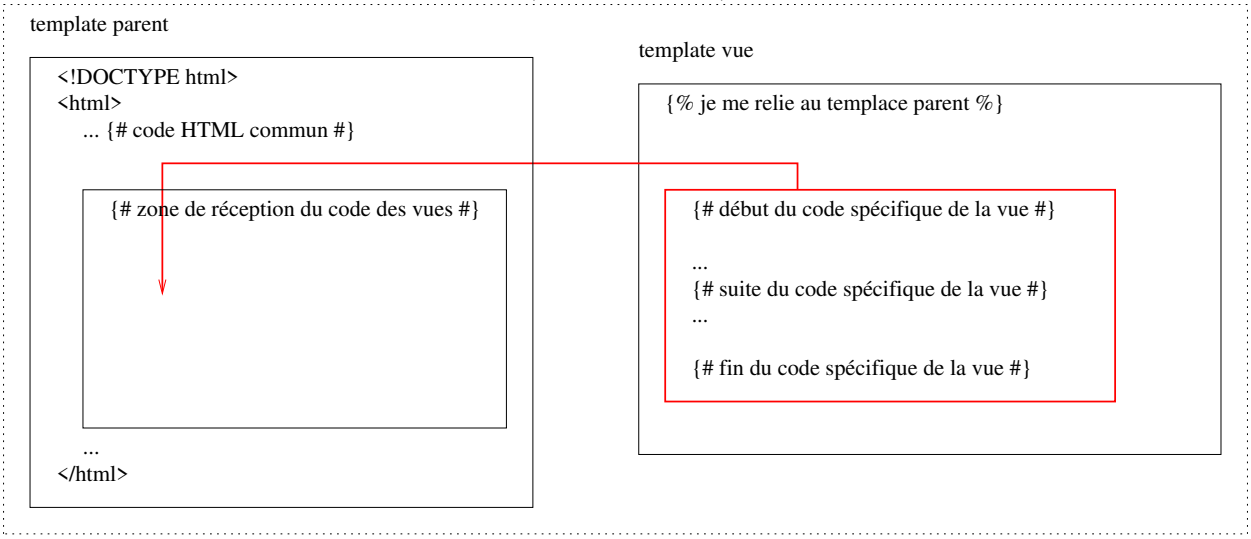
La première crainte que l'on peut avoir en écrivant les vues est de devoir répéter dans chacune l'entête du document HTML (doctype, ...).

La seconde crainte (la même que la première) est de devoir répéter des portions de code communes à toutes les pages (menu, copyright, ...).

Toute cette partie répétitive est mise dans un (ou plusieurs) template dédié, et le code des vues est inséré dans ce template.
Ce template dédié s'appelle le template parent (au sens objet du terme).

1. qui est la version 6.2 si vous faite ce TP en début d'année 2023

Voici un premier schéma illustrant le propos (en syntaxe libre) :

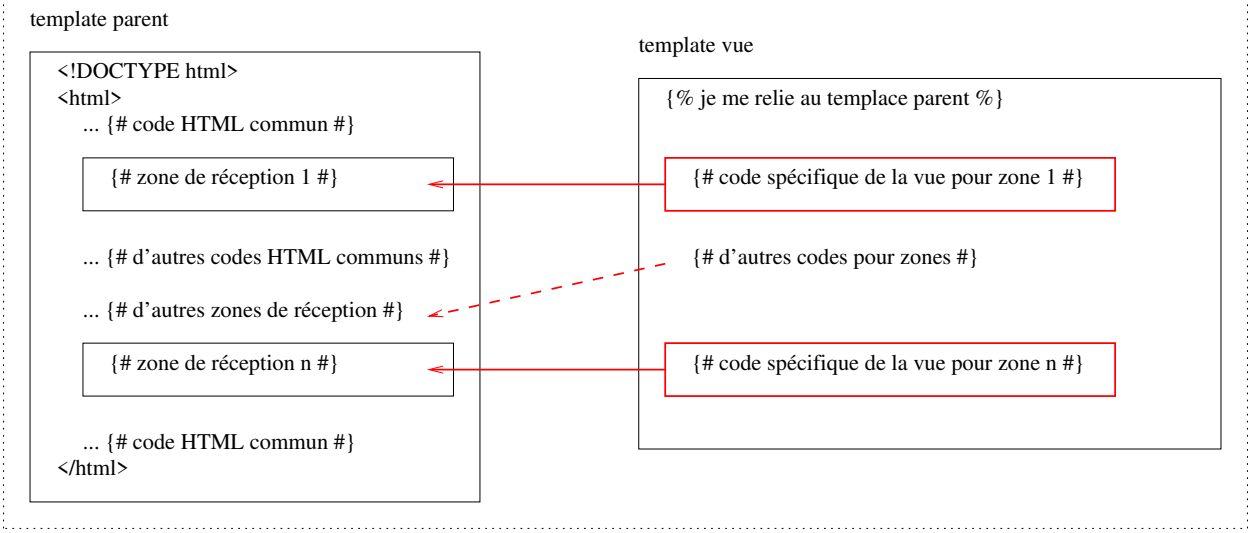


Techniquement parlant le template parent n'est pas modifié (c'est un modèle) : un fichier est créé dynamiquement et contient le code du template parent et le code de la vue.

Même s'il est un peu tôt pour en parler, voici le principe :

- Le template parent est écrit une seule fois (disons en début de création du site pour simplifier le propos)
- C'est la vue qui vient s'insérer dans le template parent. Lorsqu'on écrit le code de la vue :
  - on précise le nom du template parent
  - on écrit le code dédié à la vue (affichage d'une liste de produits, formulaire de création d'un compte client, ...)

Soyons plus précis ; ce template parent ne contient pas une zone où doit s'insérer le code des vues, mais plusieurs ; ces zones sont nommées et on parle de blocs.
Une vue précise ce que l'on met ou ce que l'on rajoute dans chacun de ces blocs.
Voici un second schéma illustrant le propos (toujours en syntaxe libre) :



Notez que dans le template parent on peut pré-remplir les blocs ; les vues ont donc le choix entre remplacer ou compléter le code initial.

2.3 Le template parent

Travail à effectuer

Le but est d'écrire un template parent relativement simple qui sera utilisé par les vues.

Lisez d'abord les explications et directives avant de coder le template. Pour la syntaxe, aidez-vous du site de symfony.

En réalité ce template parent ne sera utilisé que par les vues du contrôleur *TwigController*.

Pour information, Symfony fournit un template parent : *templates/base.html.twig*. Vous pouvez vous en inspirer, mais nous allons créer le nôtre qui sera plus simple.

L'architecture du template principal peut être récupérée à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/dbe743b02d1216263636172edd48e3cf65a528bf/templates/Sandbox/Layouts/layout1.html.twig

et est la suivante :

layout1.html.twig (à compléter)

```
1  {# contrôleur racine (niveau 1) pour TwigController #}
2  <!DOCTYPE html>
3  <html lang="fr">
4      <head>
5          <meta charset="UTF-8">
6          <title>{# ici le bloc title #}</title>
7      </head>
8      <body>
9          <div id="vue">
10             <h1>{# ici le bloc titre_principal #}</h1>
11             {# ici le bloc vue #}
12         </div>
13     </body>
14 </html>
```

- *ligne 1* : il faut remplacer "contrôleur" par "template"

En règle générale, ce type de template s'appelle *layout.html.twig* ou *base.html.twig*. Mais comme il y aura plusieurs versions, il s'appellera *layout1.html.twig*.

Il reste à préciser où placer ce fichier, voici quelques indications :

- il est lié exclusivement aux contrôleurs *Sandbox*[2],
- on considère que c'est une vue,

Bref nous pourrions le placer dans le répertoire *templates/Sandbox* à côté des répertoires contenant les vues des contrôleurs *Sandbox*. Mais comme nous allons faire plusieurs tests, nous le placerons dans *templates/Sandbox/Layouts*.

Voici des indications sur le contenu des blocs :

- Bloc title : il est nommé *title* et contiendra le nom de l'onglet dans le navigateur. Il a une valeur par défaut : "Sandbox".
- Bloc titre_principal : il est nommé *titre_principal* et contiendra le titre principal de la vue. Il n'a pas de valeur par défaut.
- Bloc vue : il est nommé *vue* et contiendra le contenu des vues. Il n'a pas de valeur par défaut.

Créez ce template que vous appellerez donc *layout1* en le mettant dans *templates/Sandbox/Layouts*.
Et remplacez les trois commentaires par le code Twig de déclaration de blocs.

Vous pouvez comparer votre code avec celui récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/1ca8ee8c0bb2de7e5bd0d492679ef3fc671ec8e3/templates/Sandbox/Layouts/layout1.html.twig

Ce qui donne :

layout1.html.twig (à compléter)

```
1  {# contrôleur racine (niveau 1) pour TwigController #}
2  <!DOCTYPE html>
3  <html lang="fr">
4      <head>
5          <meta charset="UTF-8">
6          <title>{% block title %}Sandbox{% endblock %}</title>
7      </head>
8      <body>
9          <div id="vue">
10             <h1>{% block titre_principal %}{% endblock %}</h1>
```

2. et même *SandboxTwig* dans notre cas, mais normalement il devrait concerner tous les contrôleurs *Sandbox*

```
11          {% block vue %}{% endblock %}
12      </div>
13  </body>
14 </html>
```

Quelques remarques :

- *ligne 1* : il faut remplacer “contrôleur” par “template”
- Un bloc se déclare : {% *block* <*nom_bloc*> %}{% *endblock* %}
- Le nom du bloc est libre (par exemple *titre_principal*)
- S’il y a un code par défaut, il se situe entre les marqueurs de début et de fin du bloc

Les actions et vues dans les exercices suivants seront à mettre dans le contrôleur *Twig* et dans le répertoire *templates/Sandbox/Twig*.

2.4 Vue 1

Travail à effectuer

Le but est d’écrire une première action avec sa vue qui hérite du template précédemment écrit.

Créez, dans le contrôleur *Twig*, l’action *vue1* avec sa vue. La vue hérite du layout 1, mais sans rajouter aucun code.

Vérifiez que le code source, dans le navigateur (*Ctrl-u* sous firefox), est cohérent.

Vous pouvez comparer votre code avec celui récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/944971a810f6ae5becabc721f8058243ea774dfb/src/Controller/Sandbox/TwigController.php

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/944971a810f6ae5becabc721f8058243ea774dfb/templates/Sandbox/Twig/vue1.html.twig

Ce qui donne pour le contrôleur :

Sandbox/TwigController.php (extrait)

```
12     #[Route(
13         '/vue1',
14         name: '_vue1'
15     )]
16     public function vue1Action(): Response
17     {
18         return $this->render('Sandbox/Twig/vue1.html.twig');
19     }
```

Rien de nouveau par rapport au TP précédent.

Ce qui donne pour la vue :

templates/Sandbox/Twig/vue1.html.twig

```
1 {% extends 'Sandbox/Layouts/layout1.html.twig' %}
```

Explications :

- Le fichier est réduit à une seule ligne.
- Le mot-clé est *extends* et cette directive doit être en début de fichier.
- On précise le nom du template parent via son chemin relatif à partir du répertoire *templates*.
- Le code HTML reçu par le navigateur est bien le template principal où les blocs ont été remplacés par leurs contenus par défaut.
- Si on ne précise pas le contenu d’un bloc, il garde donc son contenu par défaut (comme en objet lorsqu’on ne redéfinit pas une méthode dans une classe fille, on garde le code de la classe mère).

2.5 Vue 2

Travail à effectuer

Le but est d’écrire une seconde action avec sa vue qui hérite du template parent. Cette fois-ci, la vue redéfinit le contenu du template parent.

Créez, dans le contrôleur *Twig*, l’action *vue2* avec sa vue. La vue hérite du layout 1.

Cette vue :

- complète le bloc *title* en lui ajoutant le texte " : vue 2" pour obtenir "Sandbox : vue 2" (attention il ne s'agit pas de réécrire le mot "Sandbox", il doit être hérité du template parent)
- met un titre principal
- met un paragraphe dans le bloc *vue*

Vous pouvez comparer votre code avec celui récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/10316a3f9ab27e7288de4cf37e5cf66e31162b1a/src/Controller/Sandbox/TwigController.php

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/10316a3f9ab27e7288de4cf37e5cf66e31162b1a/templates/Sandbox/Twig/vue2.html.twig

Ce qui donne pour la vue :

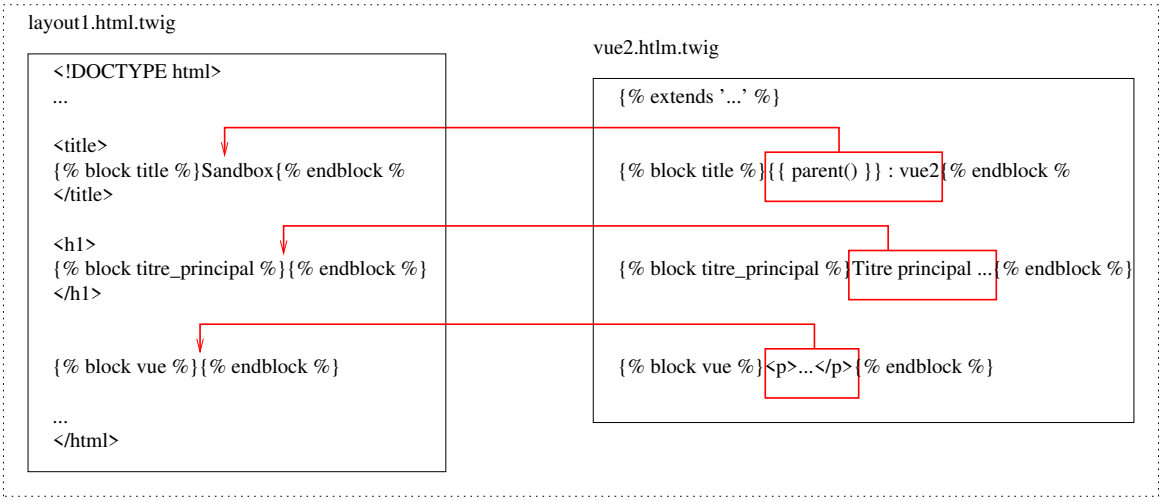
templates/Sandbox/Twig/vue2.html.twig

```
1  {% extends 'Sandbox/Layouts/layout1.html.twig' %}
2
3  {% block title %}{{ parent() }} : vue 2{% endblock %}
4
5  {% block titre_principal %}Titre principal de la vue 2{% endblock %}
6
7  {% block vue %}
8      <p>
9          Le paragraphe du template <code>vue2.html.twig</code>
10     </p>
11 {% endblock %}
```

Explications :

- *ligne 1* : on retrouve la première ligne précisant l'héritage.
- *lignes 3, 5, 7, 11* : on remet les délimiteurs {% *block* <*nom_bloc*> %}{% *endblock* %} pour les blocs dont on veut changer le contenu par rapport au template parent.
- Si on ne veut pas apporter de modification à un bloc, il suffit de ne pas le citer.
- *lignes 5, 7* : par défaut le code du template fils remplace le code du template parent.
- *ligne 3* : si l'on veut récupérer le code du template parent, il faut utiliser la fonction *parent*.
- Attention c'est le template parent (ici *layout1.html.twig*) qui "inclut" les blocs du template fils (ici *vue2.html.twig*), et non pas le contraire.
- C'est le même raisonnement que pour la programmation objets.

Voici le schéma d'inclusions spécifique à *layout1.html.twig* et *vue2.html.twig* :



2.6 Vue 3

Travail à effectuer

Le but est d'écrire une vue incorrecte, ... et de comprendre pourquoi.

Créez, dans le contrôleur *Twig*, l'action *vue3* avec sa vue.

C'est le même code que la vue 2, si ce n'est que :

- on ajoute un commentaire Twig entre les premier et second blocs

- on ajoute un commentaire HTML entre les second et troisième blocs

Que ce passe-t-il ? Et pourquoi ?

Vous pouvez comparer votre code avec celui récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/5cba971142f0c58e4d2bf686018887535f4fbfc9/src/Controller/Sandbox/TwigController.php

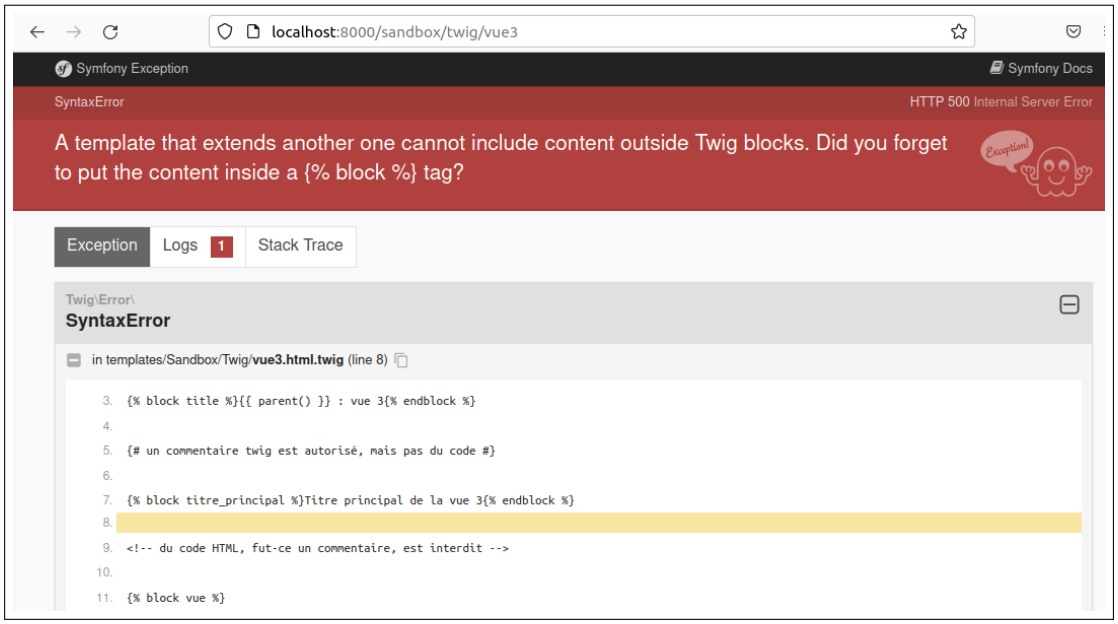
https://gitlab.com/subrenat/l3-web-2022-23/-/blob/5cba971142f0c58e4d2bf686018887535f4fbfc9/templates/Sandbox/Twig/vue3.html.twig

Ce qui donne pour la vue :

templates/Sandbox/Twig/vue3.html.twig (extrait)

```
{% extends 'Sandbox/Layouts/layout1.html.twig' %}

{% block title %}{{ parent() }} : vue 3{% endblock %}

{# un commentaire twig est autorisé, mais pas du code #}

{% block titre_principal %}Titre principal de la vue 3{% endblock %}

<!-- du code HTML, fut-ce un commentaire, est interdit -->

{% block vue %}
    <p>
```

On obtient le message d'erreur suivant dans le navigateur :



Explications :

- Le message est clair : dans un template enfant il n'est pas possible de mettre du code hors des blocs, que ce soit du code HTML ou Twig (sauf les commentaires Twig).
- Un commentaire HTML, du point de vue de Twig, est du HTML non particulier, c'est pourquoi il est refusé.
- En revanche les commentaires Twig sont acceptés n'importe où dans la page, et c'est la seule exception.
- C'est en fait logique, le code écrit dans les templates vues est inséré dans le template parent :
  - si le code est dans un bloc, Symfony sait où l'insérer dans le parent, et il n'y a pas de problème.
  - si le code est hors d'un bloc, Symfony ne sais où l'insérer dans le parent, et donc c'est refusé.

Une remarque pour terminer : un bloc peut contenir d'autres blocs, autrement dit on peut avoir une hiérarchie de blocs (à ne pas confondre avec la hiérarchie de templates).

3 Inclusion de template

Travail à effectuer

Nous allons voir comment répartir le code d'un template dans plusieurs fichiers.

Voici les liens vers la documentation officielle :
https://symfony.com/doc/current/templates.html#including-templates
https://symfony.com/doc/6.2/templates.html#including-templates

Nous continuons avec le template principal, mais cette fois-ci il ne s'agit pas d'avoir un bloc qui sera redéfini dans les vues, mais d'inclure directement un fichier *twig*.
Ce fichier contiendra le bas de page (un texte sera suffisant dans cet exercice).

L'intérêt d'inclure des fichiers twig dans le template principal est d'avoir une meilleure lisibilité du code. En effet le template principal est vite volumineux (menu, copyright, messages flash, demande d'autorisation de déposer des cookies, ...). Et créer des fichiers Twig secondaires contenant des parties du template principal facilite grandement la maintenance du code ; il y a juste à inclure ces fichiers secondaires dans le template principal.

Créez le fichier *twig* contenant le bas de page (on pourra le nommer *footer.html.twig*) avec quelques phrases.
Il faut aussi se poser la question du répertoire d'accueil de ce fichier.

Afin de ne pas toucher au premier layout principal (*layout1.html.twig*), créez le fichier *layout2.html.twig* qui est une copie de *layout1.html.twig* mais ajoutez une nouvelle division juste avant la balise *</body>* qui inclut le fichier bas de page. Basez-vous sur le modèle ci-dessous.

L'architecture de cette deuxième version de template principal peut être récupérée à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7f3abbd0b0bdf11ce23247963b8e8d582dc56f6b/templates/Sandbox/Layouts/layout2.html.twig

et est la suivante dans laquelle il faut remplacer le commentaire twig par une instruction d'inclusion :

layout2.html.twig (à compléter)

```
1  {# contrôleur racine (niveau 1) pour TwigController : version 2 #}
2  <!DOCTYPE html>
3  <html lang="fr">
4      <head>
5          <meta charset="UTF-8">
6          <title>{% block title %}Sandbox{% endblock %}</title>
7      </head>
8      <body>
9          <div id="vue">
10             <h1>{% block titre_principal %}{% endblock %}</h1>
11             {% block vue %}{% endblock %}
12         </div>
13
14         <div id="footer">
15             {# ici inclusion du template footer.html.twig #}
16         </div>
17     </body>
18 </html>
```

- *ligne 1* : il faut remplacer "contrôleur" par "template"

Créez, dans le contrôleur *Twig*, l'action *vue4* avec sa vue. La vue hérite du layout 2 et peut être une copie de *vue2.html.twig*.

Vérifiez que le code source, dans le navigateur (*Ctrl-u* sous firefox), est cohérent avec le nouveau template principal.

Discussion sur une correction du fichier *footer*

Un exemple de fichier *footer.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c34778ad8e0aeadd8ce9a3f82c41a29e739b0635/templates/Sandbox/Layouts/footer.html.twig

et est le suivant :

templates/Sandbox/Layouts/footer.html.twig

```
1  {# pied de page commun pour les pages du contrôleur Twig #}
2  <p>
3      Ceci est le bandeau de bas de page !
4  </p>
```

Explications :

- Le fichier *footer.html.twig* est un fichier général de l'application, une solution est donc de le mettre dans le même répertoire que les layouts (*templates/Sandbox/Layouts*).
- Important : il n'y a pas d'héritage, de *doctype* ou autre balise *head* ; c'est juste du code qui s'insère à un endroit d'une page existante (qui elle a un *doctype*). C'est donc un fichier très simple.

Discussion sur une/la correction du fichier *layout2*

Le fichier complet *layout2.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c34778ad8e0aeadd8ce9a3f82c41a29e739b0635/templates/Sandbox/Layouts/layout2.html.twig

et voici la portion qui nous intéresse :

templates/Sandbox/Layouts/layout2.html.twig extrait)

```
14          <div id="footer">
15              {% include 'Sandbox/Layouts/footer.html.twig' %}
16          </div>
17      </body>
```

- c'est la directive *include* qui effectue une inclusion
- on indique le fichier à inclure via son chemin relatif à partir du répertoire *templates*.
- une inclusion (comme en C ou PHP) revient à remplacer la directive par le contenu du fichier, comme si le code avait été écrit directement.
- il n'est pas nécessaire de définir un bloc pour faire une inclusion. Et même ici ce n'est vraisemblablement pas recommandé : un bloc peut être écrasé/modifié/supprimé par les templates enfants, ce qui n'est a priori pas voulu pour le pied de page d'un site.

Action et vue

Les fichiers contrôleur et vue peuvent être récupérés à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/064e8294d2215b2c0f4e6d08c02447665de0b32b/src/Controller/Sandbox/TwigController.php

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/064e8294d2215b2c0f4e6d08c02447665de0b32b/templates/Sandbox/Twig/vue4.html.twig

Il n'y a aucune différence entre la vue *vue4* et les vues 1, 2 et 3 si ce n'est qu'elle hérite du template *layout2* au lieu de *layout1*. Notamment le *footer* est géré sans avoir à modifier le code de la vue.

4 Inclusion de contrôleur

Voici les liens vers la documentation officielle :
https://symfony.com/doc/current/templates.html#embedding-controllers
https://symfony.com/doc/6.2/templates.html#embedding-controllers

Travail à effectuer

Dans un premier temps lisez la problématique et les explications permettant de la résoudre.

Notre but est d'insérer, dans le template principal, une zone qui affiche les n produits les plus chers de la base de données, où n sera paramétrable.

La question fondamentale est : pourquoi ne pas utiliser l'inclusion de template étudiée dans la section précédente ?

Réponse : ce n'est pas possible car le template à insérer a besoin de données (issues de la base de données dans notre exemple) :

- à chaque page affichée, la base de données a pu être modifiée et donc notre template doit se mettre à jour systématiquement (ce n'est donc pas un affichage statique comme le pied de page).
- un template ne peut pas faire de requête à la base de données (c'est le rôle du contrôleur, le template a pour seul rôle d'afficher les données issues d'un contrôleur).
- il faut donc passer les données à notre nouveau template.
- l'inclusion d'un template (section précédente) ne permet pas de passer des données.

Bref l'inclusion d'un template n'est pas possible.

Voici la démarche utilisée :

- le template principal choisit l'endroit où doit s'afficher l'information (comme pour l'inclusion)
- puis (au lieu d'inclure un .twig) il va appeler une action d'un contrôleur
- cette action fait la requête à la base de données
- puis (comme n'importe quelle action) elle passe le résultat à son template (fonction *render*)
- et ce template s'affichera à l'endroit choisi par le template principal (cf. premier item de cette liste)

De fait on peut dire qu'on a inclus un template, mais c'est indirectement en passant par l'intermédiaire d'un contrôleur.

Note : cela a un coût car l'inclusion d'un contrôleur relance toute la mécanique de Symfony : il ne faut donc pas en abuser.

Travail à effectuer

Après la théorie, il s'agit maintenant de suivre les directives pour insérer ces données dans le template principal.

Dans le contrôleur *Twig* ajoutez une action *palmares* ainsi que sa vue.
Note : c'est une action à caractère générale et ne devrait pas se trouver dans ce contrôleur ; mais en l'occurrence on fait des tests et on met tout le code consacré à Twig au même endroit.

L'action prend un entier n en paramètre qui est le nombre de produits qu'il faut extraire de la base de données.
Puis elle interroge la base de données pour récupérer les n produits les plus chers : comme la base de données n'est pas encore accessible, vous créerez un tableau de n produits arbitraires avec leurs prix (par exemple la suite : *produit1* à 999 euros, *produit2* à 998 euros, ainsi de suite jusqu'à *produitn* à $1000 - n$ euros).
Enfin l'action passe ce tableau à sa vue.

Attention : il ne faut pas associer de route à cette méthode, car l'action n'a de sens que si elle est insérée dans un template complet. Autrement dit cette méthode ne doit pas être routable (i.e. posséder une URL).

La vue affiche les n produit par exemple dans une liste à puces.

Pour garder une trace claire (i.e. sans jouer avec les versions dans *git*) des layouts faits jusqu'à présent :

- créez le fichier *layout3.html.twig* qui est une copie de *layout2.html.twig*
- ajoutez une nouvelle division juste après la balise <*body*> qui inclut le contrôleur *palmares* en lui passant le paramètre 3. Vous pouvez vous baser sur le modèle récupérable à :
  https://gitlab.com/subrenat/l3-web-2022-23/-/blob/a958d9247fb807a4ee829742fbeeb1e491866aa7/templates/Sandbox/Layouts/layout3.html.twig
  Ce modèle est le suivant dans lequel il faut remplacer le commentaire twig par une instruction d'inclusion de contrôleur :

layout3.html.twig (à compléter)

```twig
{# contrôleur racine (niveau 1) pour TwigController : version 3 #}
<!DOCTYPE html>
<html lang="fr">
    <head>
        <meta charset="UTF-8">
        <title>{% block title %}Sandbox{% endblock %}</title>
    </head>
    <body>
        <div id="palmares">
            {# inclure ici l'appel àTwigController::palmaresAction avec la paramètre 3 #}
        </div>

        <div id="vue">
            <h1>{% block titre_principal %}{% endblock %}</h1>
            {% block vue %}{% endblock %}
        </div>

        <div id="footer">
            {% include 'Sandbox/Layouts/footer.html.twig' %}
        </div>
    </body>
</html>
```

- *ligne 1* : il faut remplacer “contrôleur” par “template”

Enfin créez l'action *vue5* avec sa vue qui dérive du nouveau template *layout3.html.twig*. La liste des 3 produits doit apparaître et vous pouvez regarder le code source de la page dans le navigateur.

Discussion sur la correction de l'action *palmares*

Le fichier complet *TwigController.php* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c6f23cd28db053176a7dd80faf7189fb8f801eb7/src/Controller/Sandbox/TwigController.php

et voici la portion qui nous intéresse :

Sandbox/TwigController.php (extrait)

```
48      // surtout pas d'attribut Route
49      // note : cette action devrait être dans un contrôleur dédié
50      public function palmaresAction(int $n): Response
51      {
52          // normalement devrait être extrait de la base de données
53          $produits = [];
54          for ($i = 1; $i <= $n; $i ++)
55              $produits[] = ['denomination' => 'produit'.$i, 'prix' => 1000-$i];
56          return $this->render('Sandbox/Layouts/palmares.html.twig', [ 'produits' => $produits ]);
57      }
```

Explications :

- il n'y a pas d'annotation *Route*, ainsi l'action n'est pas directement accessible par l'internaute
- sinon c'est une action tout à fait classique
- comme il n'y a pas d'annotation, on ne peut pas mettre de règle pour vérifier le paramètre *n*. Mais cette action est appelée uniquement en interne par le programme (i.e. pas de risque de manipulation d'URL par l'internaute), donc cet inconvénient est très léger.

Discussion sur la correction de la vue *palmares*

Le fichier complet *palmares.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c6f23cd28db053176a7dd80faf7189fb8f801eb7/templates/Sandbox/Layouts/palmares.html.twig

et voici le listing complet de ce fichier :

Sandbox/Layouts/palmares.html.twig

```
1  <h2>Les {{ produits|length }} produit(s) les plus chers</h2>
2  <ul>
3      {% for produit in produits %}
4          <li>{{ produit.denomination }} : {{ produit.prix }} euro(s)</li>
5      {% endfor %}
6  </ul>
```

Explications :

- cette vue est générale au site (*Sandbox* pour nous) et on la met dans le même répertoire que les layouts (et comme il a été dit, c'est l'action qui n'est théoriquement pas dans le bon contrôleur)
- de fait ce template va être inséré dans le template principal, donc il ne faut pas :
  - d'héritage : sinon il y a une récursion infinie d'héritage
  - de *doctype* : sinon il y aurait deux fois un *doctype* dans la page

Discussion sur la correction du template *layout3*

Le fichier complet *layout3.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c6f23cd28db053176a7dd80faf7189fb8f801eb7/templates/Sandbox/Layouts/layout3.html.twig

et voici la portion nouvelle de ce fichier :

Sandbox/Layouts/layout3.html.twig

```
8      <body>
9          <div id="palmares">
10             {{ render(controller('App\\Controller\\Sandbox\\TwigController::palmaresAction', {n : 3})) }}
11         </div>
```

Explications :

- la syntaxe est plus compliquée : un double appel de fonctions
- on précise le nom complet de l'action via son *namespace* (on rappelle qu'il n'y a pas de nom interne)
- on note le double backslash pour séparer les *namespaces*
- note : il est possible de passer des paramètres si l'action le nécessite (format JSON)

- rappel du fonctionnement : toute la mécanique des Symfony est exécutée (contrôleur, action puis vue), puis le code (HTML dans notre cas) généré par l'exécution du Twig est insérée dans le template principal

Action *vue5* et vue correspondante

Les fichiers contrôleur et vue peuvent être récupérés à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c6f23cd28db053176a7dd80faf7189fb8f801eb7/src/Controller/Sandbox/TwigController.php

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/c6f23cd28db053176a7dd80faf7189fb8f801eb7/templates/Sandbox/Twig/vue5.html.twig

Il n'y a aucune différence entre la vue *vue5* et les vues 1, 2, 3 et 4 si ce n'est qu'elle hérite du template *layout3* au lieu de *layout2*. Notamment le palmarès est géré sans avoir à modifier le code de la vue.

5 Syntaxe Twig

Voici les liens vers la documentation officielle :

- https://symfony.com/doc/current/templates.html#twig-templating-language
- https://symfony.com/doc/5.2/templates.html#twig-templating-language
- https://twig.symfony.com
- https://twig.symfony.com/doc

Vers des parties plus spécifiques :

- *tags* : https://twig.symfony.com/doc/tags/index.html
  par exemple pour une boucle sur une collection (tag *for*) :

```
{% for produit in produits %}
```

- *filtres* : https://twig.symfony.com/doc/filters/index.html
  par exemple pour mettre du texte en majuscules (filtre *upper*) :

```
{{ 'texte'|upper }}
```

- *fonctions* : https://twig.symfony.com/doc/functions/index.html
  par exemple pour un calcul de minimum (fonction *min*) :

```
{{ min(3, 1, 3) }}
```

- *tests* : https://twig.symfony.com/doc/tests/index.html
  par exemple pour savoir si la variable *n* est paire (test *even*) :

```
{% if n is even %}
```

- *extensions spécifiques à Symfony*[3] :
  https://symfony.com/doc/current/reference/twig_reference.html
  https://twig.symfony.com/doc#reference

Travail à effectuer

Le but est de se familiariser avec les bases de Twig au fil des exercices qui suivent.

5.1 Action et vue de travail

Travail à effectuer

Il s'agit de créer la vue et l'action pour tester Twig, et également de passer un ensemble de données à la vue.

3. Twig n'est pas un produit de Symfony, mais a une existence propre et est utilisé par Symfony

Créez, dans le contrôleur *Twig*[4], l'action *vue6* avec sa vue.

L'action ne prend pas de paramètres, en revanche elle passe des variables à sa vue :

- votre prénom
- votre mail
- le tableau à récupérer sur Updago

Il est conseillé d'écrire dans le contrôleur une méthode privée qui renvoie ce tableau afin de ne pas rendre l'action *vue6* illisible.

Dans un premier temps, l'action :

- hérite de *layout3.html.twig*
- affiche un paragraphe avec le prénom et le mail
- fait un *dump* du tableau

Voici le tableau (qui est fourni, surtout pas de copier-coller) :

```php
$tab = array (
      'mentions' => array(                    // tableau des mentions -> $tab['mentions']
            'Info' => array(                  //     mention Info      -> $tab['mentions']['Info']
                  'nom' => 'Informatique',    //         nom           -> $tab['mentions']['Info']['nom']
                  'parcours' => array(        //         parcours      -> $tab['mentions']['Info']['parcours']
                        'Informatique',       //             1er       -> $tab['mentions']['Info']['parcours'][0]
                        'Image',              //             2me       -> $tab['mentions']['Info']['parcours'][1]
                    ),                        //
                  'responsable' => 'XS',      //         responsable   -> $tab['mentions']['Info']['responsable']
               ),
            'PC' => array(                    //     mention PC        -> $tab['mentions']['PC']
                  'nom' => 'Physique-Chimie', //         ...
                  'parcours' => array(
                        'Physique',
                        'Chimie minérale',
                    ),
                  'responsable' => 'GA',
               ),
            'Bio' => array(                   //     mention Bio       -> $tab['mentions']['Bio']
                  'nom' => 'Biologie',
                  'parcours' => array(
                        'Géologie',
                        'Biologie végétale',
                        'Biologie animale',
                    ),
                  'responsable' => 'MN',
               ),
         ),
      'ues' => array(                         // tableau des UEs      -> $tab['ues']
            array(                            //     1re UE            -> $tab['ues'][0]
                  'nom' => 'Algo 1',          //         nom           -> $tab['ues'][0]['nom']
                  'volume' => 54,             //         volume        -> $tab['ues'][0]['volume']
               ),
            array(                            //     2me UE            -> $tab['ues'][1]
                  'nom' => 'Maths discrètes', //         ...
                  'volume' => 40,
               ),
            array(                            //     3me UE            -> $tab['ues'][2]
                  'nom' => 'Anglais S1',
                  'volume' => 20,
               ),
            array(                            //     4me UE            -> $tab['ues'][3]
                  'nom' => 'Anglais S2',
                  'volume' => 20,
               ),
            array(                            //     5me UE            -> $tab['ues'][4]
                  'nom' => 'Projet',
                  'volume' => 70,
               ),
         ),
   );
```

4. Dans un élan de générosité, on rappele que le nom complet du contrôleur est *TwigController*

Vous pouvez comparer votre code avec celui récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/01a620a51effe2eec28df07ff31da45b9927f321/src/Controller/Sandbox/TwigController.php

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/01a620a51effe2eec28df07ff31da45b9927f321/templates/Sandbox/Twig/vue6.html.twig

Ce qui donne pour le contrôleur :

TwigController.php (extrait)

```php
    #[Route(
        '/vue6',
        name: '_vue6'
    )]
    public function vue6Action(): Response
    {
        $args = array(
            'prenom' => 'Gilles',
            'mail' => 'gilles@l3.fr',
            'offre' => $this->getOffreFormations(),
        );
        return $this->render('Sandbox/Twig/vue6.html.twig', $args);
    }
```

Explications :

- passage de paramètres classique à la vue. Le tableau est fabriqué dans une méthode (privée) annexe.

Ce qui donne pour la vue :

templates/Sandbox/Twig/vue6.html.twig

```twig
{% extends 'Sandbox/Layouts/layout3.html.twig' %}

{% block title %}{{ parent() }} : vue 6{% endblock %}

{% block titre_principal %}Vue 6 : exercices Twig{% endblock %}

{% block vue %}
    <h2>Qui suis-je ?</h2>
    <p>
        Je suis {{ prenom }} et je suis joignable à<em>{{ mail }}</em>
    </p>

    <h2>Dump du tableau des formations</h2>
    <p>Tips : Ctrl-clic sur les petites flèches</p>
    {{ dump(offre) }}
    <p>
        Tips : lorsqu'on manipule des structures complexes, il peut être utile d'avoir
        un <code>dump</code> sous les yeux pour visualiser l'organisation des données.
    </p>
{% endblock %}
```

Explications :

- rien de spécial par rapport à ce qui a déjà été vu

5.2 Exercices Twig

Travail à effectuer

Il s'agit de rentrer dans le vif du sujet avec des exercices sur Twig.

5.2.1 Noms des mentions

Dans *vue6.html.twig*, affichez la liste des noms longs des mentions en majuscules avec le nom court entre parenthèses.

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/eb45d3004f6750e95b1bcd19aa7258a28a3ac7dd/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```twig
    <h2>Noms des mentions</h2>
```

```
15      <ul>
16          {# {% for nomCourt, mention in offre['mentions'] %} #}      {# syntaxe orientée "tableau" #}
17          {% for nomCourt, mention in offre.mentions %}                {# syntaxe orientée "objet" #}
18              {# <li>{{ mention['nom']|upper }} ({{ nomCourt }})</li> #}
19              <li>{{ mention.nom|upper }} ({{ nomCourt }})</li>
20          {% endfor %}
21      </ul>
```

Explications :

- *offre* (ainsi que *prenom* et *mail*) est la variable injectée par le contrôleur.
- *nomCourt* et *mention* sont des variables locales au script (variables de boucle en l'occurrence).
- Pour accéder à une case d'un tableau associatif, il y a deux syntaxes : la notation pointée/objet (lignes 17 et 19) et la notation crochets/tableau (lignes 16 et 18). Elles sont équivalentes et la notation objet me semble plus lisible.
- Notez la syntaxe *for* pour parcourir un tableau associatif en récupérant à la fois le nom des cases et le contenu des cases.
- Notez le filtre *upper* (il en existe beaucoup d'autres [5]) pour passer en majuscules.

5.2.2 Noms et volume des UEs

Dans *vue6.html.twig*, affichez la liste des noms des UEs avec leurs volumes horaires (utilisez la concaténation avec l'opérateur ~).

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/8eab48103ce0b6090afe2d74c77af70ebf2a3585/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
24      <h2>Noms et volumes des UEs</h2>
25      <ul>
26          {% for ue in offre.ues %}
27              {# <li>{{ loop.index }}. {{ ue.nom }} : {{ ue.volume }}</li> #}    {# manière "normale" #}
28              <li>{{ loop.index }}. {{ ue.nom ~ ' : ' ~ ue.volume }}</li>       {# manière avec concaténation #}
29          {% endfor %}
30      </ul>
```

Explications :

- On note l'objet *loop* qui contient des informations sur l'itération courante.
- Une concaténation se fait à l'intérieur des double-accolades grâce à l'opérateur ~.
- On aurait pu mettre à la place le code en commentaire [6].

5.2.3 Tableau complet

Dans *vue6.html.twig*, affichez le tableau complet sous forme de listes et sous-listes à la façon de la méthode *print_r*.

Il ne s'agit pas de faire un affichage universel (i.e. qui s'applique à n'importe quel tableau) et récursif. Il y a autant de boucles imbriquées qu'il y a de niveau dans le tableau.

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/626cbb394151d5652244095c860ada5fa01b6ead/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
33      <h2>Tableau complet</h2>
34      <ul>
35          {#  partie 1 : les mentions #}
36          <li>Mentions
37              <ul>
38                  {% for nomCourt, mention in offre.mentions %}
```

5. on peut également créer ses propres filtres
6. qui me semble plus clair

```
                    {# traitement d'une mention #}
                    <li>{{ nomCourt }}
                        <ul>
                            <li>nom : {{ mention.nom }}</li>
                            <li>parcours :
                                <ul>
                                    {% for parcour in mention.parcours %}
                                        <li>parcours {{ loop.index }} : {{ parcour }}</li>
                                    {% endfor %}
                                </ul>
                            </li>
                            <li>responsable : {{ mention.responsable }}</li>
                        </ul>
                    </li>
                {% endfor %}
            </ul>
        </li>
        {#  partie 2 : les UEs #}
        <li>UEs
            <ul>
                {% for ue in offre.ues %}
                    {# traitement d'une UE #}
                    <li>UE {{ loop.index0 }}
                        <ul>
                            <li>nom : {{ ue.nom }}</li>
                            <li>volume : {{ ue.volume }} heure(s)</li>
                        </ul>
                    </li>
                {% endfor %}
            </ul>
        </li>
    </ul>
```

Explications :

- Si ce n'est que c'est pénible à écrire, il n'y a pas de difficulté particulière.
- Pour être d'accord sur la terminologie, le tableau n'a "que" deux cases : la case *mentions* et la case *ues*.
- On aurait pu faire un code automatique avec une fonction récursive. Mais en général on fait des ajustements selon l'endroit où on est dans le tableau (ajout de mot "heures" après le volume horaire par exemple).

5.2.4 Nombre de mentions

Dans *vue6.html.twig*, affichez le nombre de mentions.

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/970c131fe2c5b4550692d5572654cf801cf71409/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
    <h2>Nombre de mentions</h2>
    <p>Il y a {{ offre.mentions|length }} mention{{ offre.mentions|length > 1 ? 's' : '' }}</p>
```

Explications :

- donc c'est le filtre *length* pour avoir la taille d'un tableau
- on en profite pour montrer l'opérateur de test ternaire

5.2.5 Formatage de texte

Dans *vue6.html.twig*, affichez ce que vous voulez avec le filtre *format*.

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/2fa6a2b356881eada3290005031b113debbb2f14/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
77      <h2>Filtre <code>format</code></h2>
78      {% set ville = 'Poitiers' %}
79      <p>{{ 'Il y a %d habitants à%s en 1990' | format(78894, ville) }}</p>
```

Explications :

- Le filtre *format* permet de faire un affichage comme *printf* en langage C

5.2.6 UEs sous conditions

Dans *vue6.html.twig*, affichez la liste des noms des UEs avec leurs volumes horaires à condition que ce volume soit supérieur à 30.

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/f3ffb0da547db8ed5af4516adf23f642d2736d51/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
82      <h2>UEs dont le volume horaire est &ge; 30</h2>
83      <ul>
84          {% for ue in offre.ues %}
85              {% if ue.volume >= 30 %}
86                  <li>{{ ue.nom }} ({{ ue.volume }} heure(s))</li>
87              {% endif %}
88          {% endfor %}
89      </ul>
```

Explications :

- juste pour donner un exemple de test

5.2.7 Variable globale *app*

Dans *vue6.html.twig*, affichez les différents attributs de la variable *app* : *session*, *request*, *environment*, *debug* et *user*. Vous pouvez utiliser la fonction *dump* pour un joli affichage (qui ne marche pas en mode "prod" rappelons-le).

Discussion

Le fichier complet *vue6.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/f1605ed490403963a3673a1625a75f147d13f146/templates/Sandbox/Twig/vue6.html.twig

et voici la partie qui nous intéresse :

templates/Sandbox/Twig/vue6.html.twig

```
92      <h2>Variable globale <code>app</code></h2>
93      <p>
94          <code>app.session</code> : {{ dump(app.session) }}
95          <code>app.request</code> : {{ dump(app.request) }}
96          <code>app.environment</code> : {{ dump(app.environment) }}
97          <code>app.debug</code> : {{ dump(app.debug) }}
98          <code>app.user</code> : {{ dump(app.user) }}
99      </p>
```

Explications :

- on n'utilise pas directement le contenu de ces variables : on utilise leurs méthodes
- *app.session* : contient toutes les variables de session (*$_SESSION*). Cet objet est également contenu dans *app.request*.
- *app.request* : contient toutes les variables super-globales (session, cookies, ...)
- *app.environment* : indique si Symfony est en mode production (prod), développement (dev) ou autre
- *app.debug* : indique si on est en mode debug ce qui est généralement le cas lorsqu'on est en environnement de développement
- *app.user* : sera utilisé lors de l'authentification

6 Application Vente

Travail à effectuer

Il s'agit d'appliquer ce qui a été vu (héritage de templates et Twig) à l'application de gestion de produits.

Nous revenons au niveau de l'application principale (on quitte les contrôleurs *Sandbox*).

Nous allons avoir une hiérarchie à 3 niveaux :

- le template parent est placé dans le répertoire *templates/Layouts* et se nomme *layout.html.twig*
- le template de second niveau est dans le même répertoire et se nomme *layout_vente.html.twig*
- les templates de troisième niveau sont les vues classiques

6.1 Template racine

L'architecture du template principal peut être récupérée à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/5954835f3b8f8e5c756964f362b069b91ffc39c0/templates/Layouts/layout.html.twig

et est la suivante :

layout.html.twig (à compléter)

```twig
{# contrôleur racine (niveau 1) générique #}
<!DOCTYPE html>
<html lang="fr">
    <head>
        <meta charset="UTF-8">
        <title>{# ici le bloc title #}</title>
        {# ici le bloc css #}
        {# ici le bloc javascript #}
    </head>
    <body>
        {# ici le bloc body #}
    </body>
</html>
```

Voici des indications sur le contenu des blocs du template principal :

- *ligne 1* : il faut remplacer "contrôleur" par "template"
- il se nomme *layout.html.twig*
- bloc title : il contiendra le nom de l'onglet dans le navigateur. Il n'a pas de valeur par défaut.
- bloc css : il est vide par défaut (mais cela pourrait changer) et contiendra les inclusions des fichiers css.
- bloc javascript : idem mais pour le javascript.
- bloc body : vide par défaut, ce sera le corps de la page.

En fait c'est un template passe-partout valable pour presque n'importe quel site web.

Créez ce template.

Discussion

Le fichier complet *layout.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7d50f6adc354d17bbdbe804661964eba8acdb9bc/templates/Layouts/layout.html.twig

et voici le listing :

layout.html.twig

```twig
{# contrôleur racine (niveau 1) générique #}
<!DOCTYPE html>
<html lang="fr">
    <head>
        <meta charset="UTF-8">
        <title>{% block title %}{% endblock %}</title>
        {% block css %}{% endblock %}
        {% block javascript %}{% endblock %}
    </head>
    <body>
        {% block body %}{% endblock %}
```

```
12      </body>
13  </html>
```

Explications :

- *ligne 1* : il faut remplacer “contrôleur” par “template”
- rien de particulier dans ce template

6.2 Template intermédiaire

Il se situe entre le template principal et les vues. Il spécialise (au sens objet) le template principal pour avoir un template propre au site de vente.

Voici les directives pour ce template secondaire :

- il se nomme *layout_vente.html.twig*
- il indique le titre par défaut de l’onglet avec la valeur “Vente”
- il inclut un fichier css : *defaut.css*, en plus des inclusions faites dans le template racine (inutile de créer le fichier css, mais si vous le faites alors utilisez la fonction twig *asset*).
- il inclut deux fichiers javascript : *global.js* et *macros.js*, en plus des inclusions faites dans le template racine (inutile de créer les fichiers js, mais si vous le faites alors utilisez la fonction twig *asset*).
- il précise l’architecture du bloc *body*. Autrement dit il y a des ajouts de code HTML et des créations de nouveaux blocs à l’intérieur du bloc *body*[7], et ces nouveaux blocs pourront être redéfinis dans les vues. Voici les directives :
  - une première division affiche les messages flash de type “info” (pas de définition de bloc ici) ; le code sera dans un template inclus pour ne pas surcharger le code du template secondaire.
  - une seconde division inclut un contrôleur *menu* : la méthode *menu* se met dans le contrôleur *Accueil* et n’a pas de route associée (pas de définition de bloc ici) : pour l’instant l’action *menu* ne transmet aucune donnée à sa vue, et mettez un menu en dur dans le template.
  - une division inclut un template représentant le haut de la page (pas de bloc) ; le code de l’entête est laissé à votre appréciation.
  - une division qui définit un titre *h1* contenant un bloc avec comme valeur par défaut “Site de vente”.
  - une division contient un bloc qui inclura le contenu des vues.
  - une division inclut un template représentant le bas de la page (pas de bloc) ; le code du pied de page est laissé à votre appréciation.

Discussion sur le layout secondaire

Le fichier complet *layout_vente.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/templates/Layouts/layout_vente.html.twig

et voici le listing :

layout_vente.html.twig

```
1  {# contrôleur intermédiaire (niveau 2) spécifique au site de vente #}
2  {% extends 'Layouts/layout.html.twig' %}
3
4  {% block title %}Vente{% endblock %}
5
6  {% block css %}
7      {{ parent() }}
8      <link rel="stylesheet" media="screen" type="text/css" title="default" href="{{ asset('css/default.css') }}" />
9  {% endblock %}
10
11 {% block javascript %}
12     {{ parent() }}
13     <script type="text/javascript" src="{{ asset('js/global.js') }}"></script>
14     <script type="text/javascript" src="{{ asset('js/macros.js') }}"></script>
15 {% endblock %}
16
17 {% block body %}
18     <div id="flash">
19         {% include 'Layouts/flash.html.twig' %}
20     </div>
```

7. En revanche il est impossible de créer de nouveaux bloc en dehors des blocs présent dans le le template principal

```
21
22      <div id="menu">
23          {{ render(controller('App\\Controller\\AccueilController::menuAction')) }}
24      </div>
25
26      <div id="header">
27          {% include 'Layouts/header.html.twig' %}
28      </div>
29
30      <div id="titre">
31          <h1>{% block titre_principal %}Site de vente{% endblock %}</h1>
32      </div>
33
34      <div id="vue">
35          {% block vue %}{% endblock %}
36      </div>
37
38      <div id="footer">
39          {% include 'Layouts/footer.html.twig' %}
40      </div>
41  {% endblock %}
```

Explications :

- *ligne 1* : il faut remplacer "contrôleur" par "template"
- *ligne 2* : donc le template secondaire hérite du template principal, et les vues hériteront du template secondaire.
- *ligne 4* : on écrase une éventuelle valeur par défaut présente dans le template principal.
- *ligne 7* : on appelle la fonction *parent* pour récupérer les éventuelles inclusions de *.css* faites dans le template principal. Le fait qu'il n'y ait pas d'inclusion de *.css* dans le template principal n'entre pas en ligne de compte : ce dernier pourrait évoluer dans le futur.
- *ligne 8* : on note l'utilisation de la fonction *asset* qui permet de désigner un chemin à partir du répertoire *public* du site. Cet aspect sera vu ultérieurement.
- *lignes 11 à 15* : même remarques que pour le bloc *css*.
- *ligne 19* : rien de particulier, on déporte le code affichant les messages flash dans un template dédié.
- *ligne 23* : le choix a été fait d'inclure un contrôleur (et non un template) car à terme le code affichant le menu aura besoin de données issues de la base de données, notamment pour avoir un menu qui s'adapte à l'utilisateur connecté.
- *lignes 27* : rien de particulier par rapport à ce qui a déjà été vu.
- *ligne 31* : comme pour le bloc *title*, on impose un titre par défaut pour le site. Cet titre pourra être complété ou écrasé par les vues.
- *ligne 35* : c'est ici que se mettra la quasi-totalité du code des vues. On voit bien qu'une vue peut alors se concentrer sur son travail sans se préoccuper de tout ce qui est autour (messages flash, entête, ...).
- *lignes 39* : rien de particulier par rapport à ce qui a déjà été vu.

Discussion sur les *.css* et *.js*

Les fichiers peuvent être récupérés à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/public/css/default.css

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/public/js/global.js

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/public/js/macros.js

Ils n'ont rien de passionnant. Ce qui est important c'est qu'ils se trouvent dans le répertoire *public* ; en effet ce répertoire est le seul à être visible sur le web, et donc tous les fichiers de style, javascript et images doivent être à l'intérieur.

Discussion sur les messages flash

Le fichier complet *flash.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/templates/Layouts/flash.html.twig

et voici le listing :

layout_vente.html.twig

```
1  {# affichage des messages flash #}
2
3  {% if app.session.flashBag.has('info') %}
```

```
4       <strong>Messages d'information</strong>
5       <ul>
6           {% for msg in app.session.flashBag.get('info') %}
7               <li>{{ msg }}</li>
8           {% endfor %}
9       </ul>
10  {% endif %}
```

Explications :

- *ligne 3* : c'est l'occasion de montrer que l'objet *flashBag* a une méthode qui indique s'il y a des messages à disposition.
- le reste est inchangé par rapport à ce qui a déjà été vu.

Discussion sur l'entête et le pied de page

Les fichiers peuvent être récupérés à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/templates/Layouts/header.html.twig

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/templates/Layouts/footer.html.twig

Ils n'ont rien de spécial.

Discussion sur le menu

Le fichier complet *AccueilController.php* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/src/Controller/AccueilController.php

et voici le listing de l'action qui a été rajoutée :

AccueilController.php (extrait)

```
17      // pour inclusion de contrôleur dans le template secondaire : action non routable
18      public function menuAction(): Response
19      {
20          $args = array(
21          );
22          return $this->render('Layouts/menu.html.twig', $args);
23      }
```

Explications :

- *ligne 17* : on insiste à nouveau sur le fait que cette action n'est pas routable, c'est à dire n'est pas accessible via une URL saisie dans le navigateur. Cette action n'a de sens que si elle est incluse dans un template complet.
- *ligne 20* : on prépare juste le terrain lorsqu'il y aura des données à envoyer à la vue.
- *ligne 22* : le choix a été fait de mettre le template dans le répertoire *Layouts* au lieu de *Accueil* car le menu est un code global à toute l'application.
  Il y a donc un manque de cohérence. Vraisemblablement il aurait fallu un contrôleur dédié (appelé *LayoutsController*) pour le menu.

Le fichier *menu.html.twig* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/7381a9967c60ee488a36e115f0fbebbe52c6c9ac/templates/Layouts/menu.html.twig

Pour l'instant ce n'est qu'une simple liste non ordonnée.

6.3 Vues

Enfin modifiez les vues existantes (normalement les actions liées ne doivent pas être modifiées) et faites les dériver du template secondaire.

Discussion sur le fichier *Accueil/index.html.twig*

Le fichier complet *Accueil/index.php* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/3ee3287e60ee5f2856611be898f343da7a2211a9/templates/Accueil/index.html.twig

et voici le listing :

Accueil/index.html.twig

```
1   {% extends 'Layouts/layout_vente.html.twig' %}
2
3   {% block title %}{{ parent() }} : bienvenue{% endblock %}
4
5   {#
6       note pédagogique :
```

```
    - on ne redéfinit pas le bloc "body" sinon on perd tout ce qu'il y a dedans
    - on ne redéfinit que les blocs internes à"body"
#}

{% block titre_principal %}{{ parent() }} : bienvenue{% endblock %}

{# on ne redéfinit pas le bloc "vue" car on n'y ajoute rien #}
```

Explications :

- *ligne 3* : on rappelle la manière de récupérer le contenu du bloc dans le template parent.
- cf. commentaires dans le code

Discussion sur le fichier *Magasin/valeurStock.html.twig*

Le fichier complet *Accueil/index.php* peut être récupéré à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/3ee3287e60ee5f2856611be898f343da7a2211a9/templates/Magasin/valeurStock.html.twig

et voici le listing :

Magasin/valeurStock.html.twig

```
{% extends 'Layouts/layout_vente.html.twig' %}

{% block title %}{{ parent() }} : magasins{% endblock %}

{% block titre_principal %}{{ parent() }} : valeur du stock du magasin {{ id }}{% endblock %}

{% block vue %}
    <p>
        La valeur du stock est {{ total }} euro(s).
    </p>
{% endblock %}
```

Explications :

- on voit que le code est grandement simplifié par rapport à la version sans héritage, et par conséquent beaucoup plus facile à lire et à maintenir.

Discussion sur les trois autres vues

Les trois fichiers peuvent être récupérés à :

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/3ee3287e60ee5f2856611be898f343da7a2211a9/templates/Magasin/stock.html.twig

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/3ee3287e60ee5f2856611be898f343da7a2211a9/templates/Produit/list.html.twig

https://gitlab.com/subrenat/l3-web-2022-23/-/blob/3ee3287e60ee5f2856611be898f343da7a2211a9/templates/Produit/view.html.twig

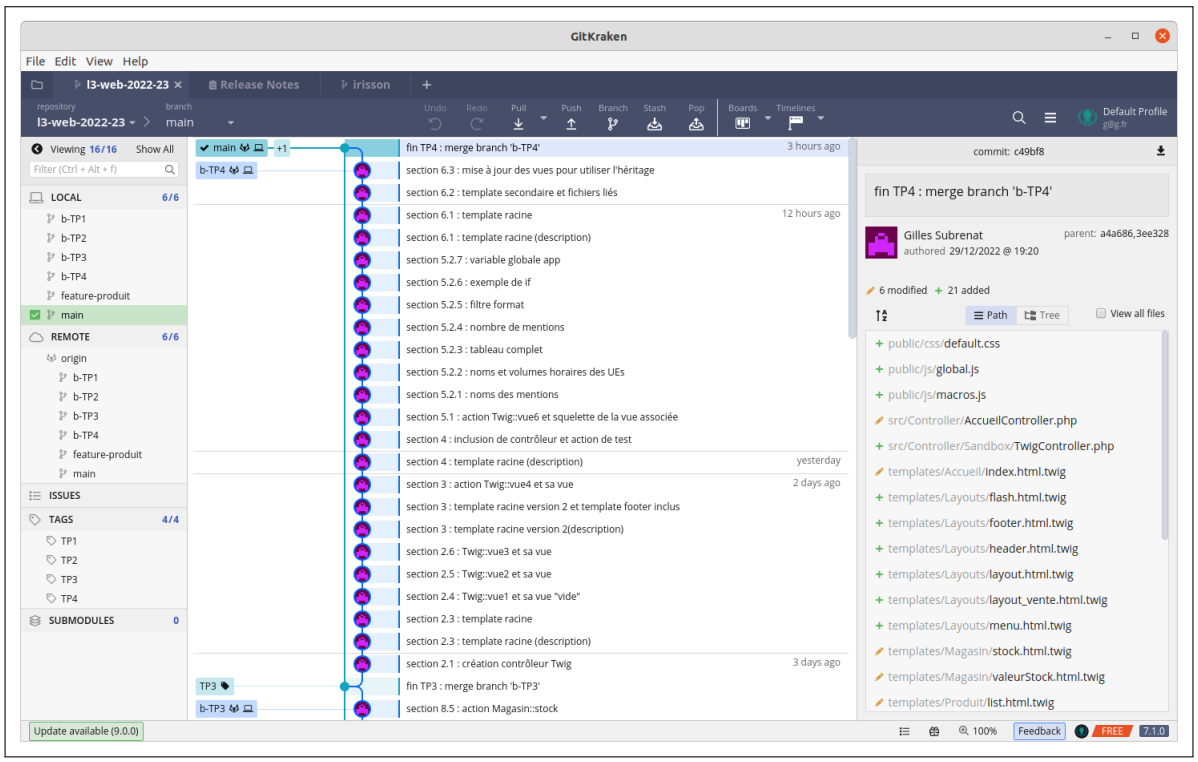
Pas de discussion particulière si ce n'est de rappeler que les codes des actions n'ont pas eu besoin d'être changés.

7 Git : code source

Le code source à l'issue de ce TP peut-être consulté à :
https://gitlab.com/subrenat/l3-web-2022-23/-/blob/TP4

Les différentes étapes du TP sont dans la branche *b-TP4*.

Voici un screenshot fait avec git-kraken (https://www.gitkraken.com) :


GitKraken
File Edit View Help
l3-web-2022-23
Release Notes
irisson
repository
l3-web-2022-23
branch
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Boards
Timelines
Default Profile
Viewing 16/16
Show All
Filter (Ctrl + Alt + f)
LOCAL 6/6
b-TP1
b-TP2
b-TP3
b-TP4
feature-produit
main
REMOTE 6/6
origin
b-TP1
b-TP2
b-TP3
b-TP4
feature-produit
main
ISSUES
TAGS 4/4
TP1
TP2
TP3
TP4
SUBMODULES 0
main +1
b-TP4
fin TP4 : merge branch 'b-TP4' 3 hours ago
section 6.3 : mise à jour des vues pour utiliser l'héritage
section 6.2 : template secondaire et fichiers liés
section 6.1 : template racine 12 hours ago
section 6.1 : template racine (description)
section 5.2.7 : variable globale app
section 5.2.6 : exemple de if
section 5.2.5 : filtre format
section 5.2.4 : nombre de mentions
section 5.2.3 : tableau complet
section 5.2.2 : noms et volumes horaires des UEs
section 5.2.1 : noms des mentions
section 5.1 : action Twig::vue6 et squelette de la vue associée
section 4 : inclusion de contrôleur et action de test
section 4 : template racine (description) yesterday
section 3 : action Twig::vue4 et sa vue 2 days ago
section 3 : template racine version 2 et template footer inclus
section 3 : template racine version 2(description)
section 2.6 : Twig::vue3 et sa vue
section 2.5 : Twig::vue2 et sa vue
section 2.4 : Twig::vue1 et sa vue "vide"
section 2.3 : template racine
section 2.3 : template racine (description)
section 2.1 : création contrôleur Twig 3 days ago
TP3
fin TP3 : merge branch 'b-TP3'
b-TP3
section 8.5 : action Magasin::stock
commit: c49bf8
fin TP4 : merge branch 'b-TP4'
Gilles Subrenat
authored 29/12/2022 @ 19:20
parent: a4a686,3ee328
6 modified + 21 added
Path
Tree
View all files
public/css/default.css
public/js/global.js
public/js/macros.js
src/Controller/AccueilController.php
src/Controller/Sandbox/TwigController.php
templates/Accueil/index.html.twig
templates/Layouts/flash.html.twig
templates/Layouts/footer.html.twig
templates/Layouts/header.html.twig
templates/Layouts/layout.html.twig
templates/Layouts/layout_vente.html.twig
templates/Layouts/menu.html.twig
templates/Magasin/stock.html.twig
templates/Magasin/valeurStock.html.twig
templates/Produit/list.html.twig
Update available (9.0.0)
100%
Feedback
FREE
7.1.0