Twigによるビュー

デフォルトで、SymfonyはTwigというレンダリングエンジンを使います。
これはHTMLテンプレートエンジンで、PHPコードを書かずにHTMLの中に変数や制御構造を簡単に埋め込むことができます。

補足：TwigはSymfonyなしでも、どんなPHPプロジェクトでも使えます。

TwigはHTMLファイルに機能を追加します。
ファイル名が **name.html.twig** となっていることに気づくでしょう。

これらのtwigファイルにはPHPは含まれていません。データを挿入したい場合は、twig独自の機能を使います。

base.html.twig

コントローラーから呼ばれるファイルを見る前に、**base.html.twig** に注目しましょう。

これは普通のHTMLファイルのように始まりますが、`{}` の中が見えたらそれがtwigの開始・終了タグです。
3種類あります：

- `{{ }}` ダブル中括弧はPHPの「**echo**」に対応します。
- `{# #}` はtwigのコメントです。
- `{% %}` は出力を伴わないコードです。

ブロック

baseファイルには以下のtwigがあります：

- titleタグの中にある `block` というキーワード。ブロックとは、他のテンプレートで上書き可能な名前付きのコンテンツ領域です。
  デフォルトではその中の内容が表示されますが、子テンプレートがこのブロックを再定義すると元の内容は置き換わります。
- 次にcss用のブロック、javascript用のブロックがあります。
  - javascriptブロックの中に「importmap」というブロックもあります（後で説明します）。
  - これらのブロックは例えばページごとに異なるスクリプトやスタイルを追加・置換するのに便利です。
- 最後にbodyに空の`body`ブロックがあります。

ここで、デフォルトのタイトルを「Symfony Course」に変えましょう。

home/index.html.twig

コントローラーを作成すると、Symfonyは自動で「index」という対応するビューをコントローラー名のフォルダーに生成します。

デフォルトの内容は次の通りです：

- `extends` キーワードがあります。これはこのファイルが「base.html.twig」を継承していることを示します。ほとんどのページテンプレートで使います。
  重要：`{% extends "base.html.twig" %}` を使うテンプレートは、すべてのコンテンツがブロックの中に書かれていなければ無視されます。
- 次にtitleブロックがあります。ページタイトルに対応しています。コメントアウトするとbaseのタイトルが表示されます。
  デフォルトの内容を消さずに、`parent()`関数で追記することも可能です。例えば、タイトルを次のように書き換えます：

```
{{ parent() }} - Home
```

タイトルはデフォルトと新しい内容の合体になります。

- 次はbodyブロックです。baseでは空でしたが、ここにはコントローラーやテンプレートの位置情報が入っています。これは削除可能です。
  ここに書いた内容がbaseの空のbodyを置き換えます。
- ここには他に何もありませんが、もし特定ページ用のcssやjsがあれば対応するブロックで呼べます。
- ブロック名は自由に付けられます。例えばページごとに異なるfooterを用意したければfooterブロックを作って置き換えられます。

VS Codeの設定

注意：**HTMLのemmetが使えません。**

twigファイルなのでVS Codeはtwig言語として認識しており、HTMLではありません。
でも簡単に修正できます：

1. VS Codeの設定で `emmet.includeLanguages` を探す
2. 項目を追加
3. キーに `twig` を設定
4. 値に `html` を設定

これでtwigファイルでもHTMLのemmetが使えます。

表示

renderメソッドは変数をビューに渡せます。例えば：

```
{{ controller_name }}
```

変数の値が表示されます。

変数表示時にフィルターを適用して加工もできます。

```
{{ controller_name | upper }}
```

値が大文字になります。

使えるフィルター・関数はたくさんあり、自作も可能です。詳しくは：

https://twig.symfony.com/doc/3.x/#reference

変数が連想配列やオブジェクトの場合は、

`variable.property`

のようにアクセスします。

例えば `pays` 変数なら：

```
{{ pays.france }}
<br>
{{ pays.england | reverse }}
```

注意：twigは配列全体は直接表示できません。

条件分岐

Twigはif、elseif、elseも使えます：

```
<strong>
{% if chiffre > 5 %}
    {{ chiffre }} は5より大きい
{% elseif chiffre < 5 %}
    {{ chiffre }} は5より小さい
{% else %}
    あなたの数字は5です。
{% endif %}
</strong>
```

配列ループ

もちろん配列のループも可能です：

```
<ul>
    {% for f in fruits %}
        <li>{{ f }}</li>
    {% endfor %}
</ul>
```

配列に `| reverse` フィルターもかけられます。

twigはSQLクエリ結果などの空配列も扱えます。`for`に `else` を入れるだけ：

```
{% for v in vide %}
    ここは表示されません（空配列）
{% else %}
    配列は空です。
{% endfor %}
```

XSS対策

SymfonyのtwigはデフォルトでXSS対策のために全表示データをエスケープします：

```
{{ xss }}
```

変数にHTMLやJSを含めて表示したい場合は、`raw`フィルターを使います。

header と footer

少し離れて、`layout` フォルダに `_header.html.twig` と `_footer.html.twig` を作ります。まずヘッダーから。

h1には2つ変数を置きました。

- `header` はヘッダー全体を変更可能にします。
- `title` はh1のタイトルだけを変えられます。

```
<header>
    {% if header is defined %}
        {{ header|raw }}
    {% else %}
    <h1>
        {{ title ?? "Symfony Course" }}
    </h1>
    {% endif %}
</header>
```

footerにはトップページへのリンクがあります。
サイト構築時のリンク管理は重要です。あとでルートが変わっても全部変えなくて済むようにしましょう。

```
<footer>
    <a href="">トップページへ戻る</a>
</footer>
```

path()

トップページルートを「app_home」としました。twigの`path()`関数を使います。

```
<a href="{{ path("app_home") }}">トップページへ戻る</a>
```

ルート名が変わらない限り、ルートURLが自動で得られます。

メールなどでは `url("routeName")` を使うと、ドメイン名を含むURLが取得できます。

include

base.html.twigのbody周りに以下を追加しましょう：

```
{% include "layout/_header.html.twig" %}
<main class="{{ mainClass ?? '' }}">
    {% block body %}{% endblock %}
</main>
{% include "layout/_footer.html.twig" %}
```

`include` は他のtwigファイルを挿入します。

TwigファイルはPHPを含まず、Twigの機能だけでデータを埋め込みます。

`main`タグにクラスが動的に渡せるようになっています（`mainClass`変数）。

CSSの読み込み

CSSやJSの扱いは後で詳しく説明します。
とりあえず `/assets/styles/app.css` に配布されたCSSを入れてください。

グローバル変数 "app"

Twigでどこでも使えます。

- ログインユーザー情報
- フラッシュメッセージ
- セッション情報
- リクエスト情報
- など（公式ドキュメント参照）

今回は現在のルート名取得に使います。ヘッダーに以下を追加：

```
<nav class="header-navigation">
    <a href="{{ path('app_home') }}" class="{{ app.current_route == 'app_home' ?
'active' : '' }}">ホーム</a>
    <a href="" class="{{ app.current_route == 'app_user_list' ? 'active' : '' }}">
ユーザー一覧</a>
</nav>
```

現在ルート名と比較して該当リンクにクラスを付けています。

コントローラーのinclude

例えば右側に最近の記事リストを出したい場合。
各コントローラーでクエリを繰り返すのは避けたいです。

twigは他のビューではなく、別コントローラー自体をincludeできます。

表示したい場所に書きます：

```
{{ render(controller("App\\Controller\\ControllerName::MethodName")) }}
```