6 minutes

Les filtres Twig

Les filtres `Twig`

Les filtres permettent de modifier le contenu d'une expression avant de l'afficher.

La syntaxe est `{{ expression | filtre }}`. Il est possible d'ajouter des espaces pour plus de clarté : `{{ expression | filtre }}`.

Il existe environ 70 filtres `Twig`, dont une partie ne peut être utilisé qu'avec `Symfony`.

Nous allons voir les principaux maintenant, et en verrons d'autres au fur et à mesure de la formation.

Principaux filtres pour traiter des nombres

Nous allons voir les principaux filtres pour les nombres.

Le filtre `abs`

Le filtre `abs` permet de retourner la valeur absolue d'un nombre :

```
{# Si n = -5, affiche 5 #}
<h1>
  {{ n | abs }}
</h1>
```

Copier

Le filtre `round`

Le filtre `round` permet d'arrondir un nombre :

```
{# Affiche 43.3 #}
{{ 43.22|round(1, 'ceil') }}
```

Copier

Le premier argument est la précision (nombre de chiffres après la virgule).

Le deuxième argument est la méthode d'arrondi : `ceil` (supérieur), `floor` (inférieur) ou `common` (au plus proche, valeur par défaut).

Le filtre `format_number`

Le filtre `format_number` permet de formater un nombre.

Pour ce filtre il faut installer une dépendance :

```
composer require twig/intl-extra
```

Copier

Il est ensuite possible de faire des formatages avancés, comme par exemple :

```
{# Affiche par exemple : 22.5% #}
{{ '0.22586478'|format_number({rounding_mode: 'floor', fraction_digit: 1}, style='percent') }}
```

Copier

Nous ne verrons pas ici une liste exhaustive des options, des styles et des formats locaux possibles car il y en a une trentaine et cela ne serait pas intéressant.

Reportez vous à la liste des options possibles à cette adresse.

Principaux filtres pour traiter des chaînes de caractères

Nous allons voir les principaux filtres pour les chaînes de caractères.

Le filtre `capitalize`

Le filtre `capitalize` permet de mettre le premier caractère en majuscule et les autres en minuscule :

```
{# Affiche Test #}
{{ 'test' | capitalize }}
{{ 'TEST' | capitalize }}

{# Affiche Hello un test #}
{{ 'Hello UN TEST' | capitalize }}
```

Copier

Le filtre `title`

Le filtre `title` permet de mettre le premier caractère de chaque mot en majuscule et les autres en minuscule :

```
{# Affiche Hello Un Test #}
{{ 'Hello UN TEST' | title }}
```

Copier

Le filtre `trim`

Le filtre `trim` permet d'enlever les espaces au début et à la fin d'une chaîne de caractères :

```
{# Affiche test #}
{{ '        test    ' | trim }}
```

Copier

Les filtres `lower` et `upper`

Le filtres `lower` et `upper`, permettent respectivement de mettre un text en minuscules ou en majuscules :

```
{# Affiche TEST #}
{{ 'test' | upper }}

{# Affiche test #}
{{ 'TEST' | lower }}
```

Copier

Le filtre `join`

Le filtre `join` permet de retourner une chaîne de caractères à partir d'une séquence d'éléments, en précisant le séparateur à utiliser en premier argument, et éventuellement un séparateur pour les deux derniers éléments :

```
{# Affiche : 42, 43, 44 #}
{{ [42, 43, 44]|join(', ') }}

{# Affiche : 42, 43 et 44 #}
{{ [42, 43, 44]|join(', ', ' et ') }}
```

Copier

Principaux filtres pour traiter des séquences

Nous allons voir les principaux filtres pour les séquences d'éléments : par exemple des tableaux.

Le filtre `length`

Le filtre `length` permet de retourner la valeur de la fonction `PHP count()` appliqué à l'élément.

Il est particulièrement utile :

```
{% set n = [1,2,3,4] %}

{# Affiche par test #}
{% if n|length > 3 %}
  Test
{% endif %}
```

Copier

Principaux filtres pour traiter des dates

Nous allons voir les principaux filtres pour les dates.

Le filtre `date`

Le filtre `date` permet de formater une date.

Il accepte des `DateTime`, les chaînes de caractères acceptées par `strtotime()` et des `DateInterval`.

Si vous ne vous souvenez plus de ces classes et fonction PHP, nous vous recommandons d'aller revoir le chapitre sur les dates de la formation PHP.

```
{# Affiche par exemple : 23/07/2021 #}
{{ "+1 week 2 days 4 hours"|date("d/m/Y") }}
```

Copier

Par défaut les dates sont affichés dans le fuseau horaire spécifié dans le fichier `php.init` (*cf le chapitre date de PHP*).

Le filtre `date_modify`

Le filtre `date_modify` permet de modifier une date.

Il accepte des `DateTime`et les chaînes de caractères acceptées par `strtotime()`.

Nous pouvons donc faire :

```
{# Affiche par exemple : 17/07/2020 #}
{{ "last year"|date_modify("+3 day")|date("d/m/Y") }}
```

Copier

Le filtre `format_datetime`

Le filtre `format_datetime` permet de formater une date suivant la locale de l'utilisateur.

A noter que le filtre `format_date` a le même fonctionnement mais ne peut formater l'heure.

Pour ce filtre il faut installer une dépendance :

```
composer require twig/intl-extra
```

Copier

Vous pourrez ensuite formater des dates de cette manière :

```
{# Affiche par exemple : samedi 7 août 2021 à 21:39 #}
{{ '2021-08-07 21:39:10'|format_datetime('full', 'short', locale='fr') }}
```

Copier

Le premier argument est le format de la date.

Le second argument est le format du temps.

Vous pouvez passer en argument nommé la locale, sinon celle par défaut sera utilisée.

Les valeurs possibles sont `none`, `short`, `medium`, `long` et `full`.

Le formatage respecte les normes `Unicode` que vous pouvez retrouver à cette adresse.

Principaux filtres de programmation fonctionnelle

Nous allons voir les principaux filtre de programmation fonctionnelle comme `map()` ou `filter()`.

Le filtre `filter`

Le filtre `filter()` permet de filtrer les éléments en utilisant une fonction fléchée.

```
{% set tableau = [1, 2, 3, 4, 5, 6] %}

{# Affiche par exemple : 2 4 6 #}
{{ tableau|filter(el => el % 2 == 0)|join(' ')}}
```

Copier

C'est souvent utile en combinaison avec une boucle `for` :

```
{% set tableau = [1, 2, 3, 4, 5, 6] %}
```

```
{# Affiche par exemple : 3 4 5 6 #}
{% for el in tableau|filter(el => el > 2) %}
  {{ el }}
{% endfor %}
```

Copier

Le filtre `map`

Le filtre `map()` permet d'itérer sur une séquence d'éléments comme la fonction `PHP array_map()` :

```
{% set utilisateurs = [
    {prenom: "Pierre", nom: "Philippe"},
    {prenom: "Jean", nom: "Dupont"},
] %}

{# Affiche Pierre Philippe, Jean Dupont #}
{{ utilisateurs|map(p => "#{p.prenom} #{p.nom}")|join(', ') }}
```

Copier

Le filtre `reduce`

Le filtre `reduce()` permet de réduire une séquence d'éléments à une seule valeur.

Par exemple, il permet de faire la somme des éléments d'un tableau :

```
{% set n = [42, 21, 12] %}

{# Affiche 75 #}
{{ n|reduce((curr, val) => curr + val) }}
```

Copier

Nous verrons les autres filtres utilisables uniquement avec `Symfony`, dans les chapitres suivants lorsque nous aurons abordés les notions correspondantes.