Affichage avec Twig

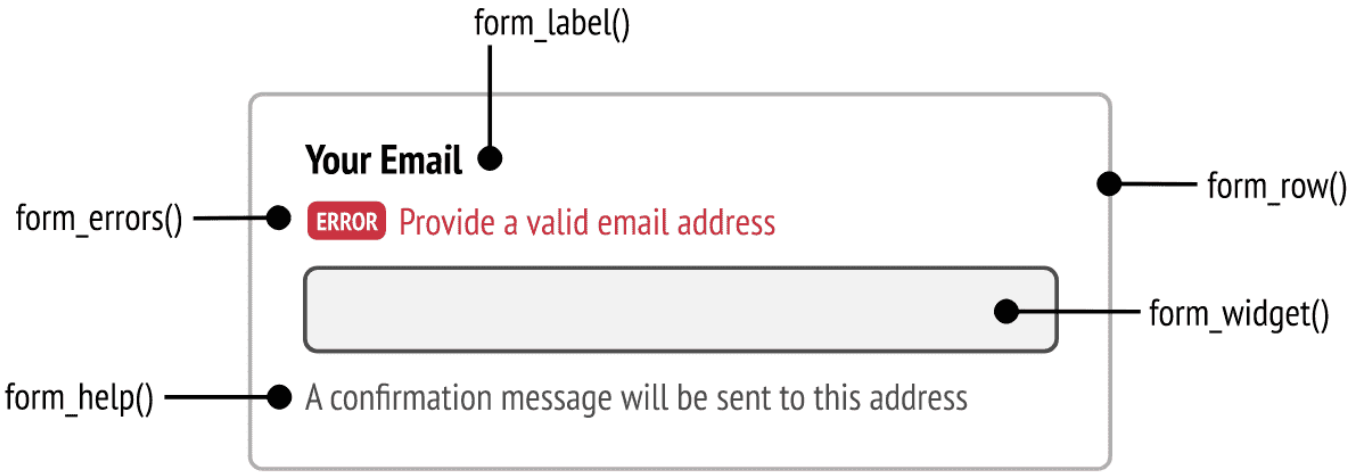
Afficher un formulaire avec `Twig`

Nous allons commencer par quelques rappels de choses que nous avons vu jusqu'à maintenant.

Nous avions vu que pour afficher l'ensemble d'un formulaire `Symfony` dans un `template Twig` il suffisait de faire :

```
{{ form(nomFormulaire) }}
```

Nous avions également vu que chaque champ créé par le composant `Form` de `Symfony` avait cette structure :



Dans cette leçon nous allons voir comment nous pouvons modifier l'affichage du formulaire en utilisant d'autres fonctions `Twig` que `form()`.

Personnaliser l'affichage avec d'autres fonctions `Twig`

Les fonctions `Twig` suivantes permettent de personnaliser l'affichage des formulaires `Symfony` : `form_start()`, `form_end()`, `form_label()`, `form_widget()`, `form_help()`, `form_errors()` et `form_row()`.

Les fonctions `form_start()` et `form_end()`

La fonction `form_start()` permet de créer la balise `HTML form` ouvrante et de spécifier des options pour le formulaire, comme par exemple l'action à utiliser. La fonction

`form_end()` permet de créer la balise `HTML form` fermante et appel automatiquement la fonction `form_rest()` qui affiche tous les champs qui n'ont pas encore été affichés.

Vous pouvez donc afficher l'ensemble d'un formulaire en faisant :

```
{{ form_start(nomFormulaire, {'method': 'GET'}) }}

{{ form_end(nomFormulaire) }}
```

La fonction `form_row()`

La fonction `form_row()` permet d'afficher le champ spécifié :

```
{{ form_start(nomFormulaire) }}
  {{ form_row(nomFormulaire.content) }}
{{ form_end(nomFormulaire) }}
```

Cela affiche automatiquement le label, les erreurs, la partie aide et le champ lui-même (partie `widget`).

Nous verrons que nous pourrons l'utiliser avec des thèmes.

La fonction `form_label()`

La fonction `form_label()` permet d'afficher la partie label du champ spécifié.

Elle permet de remplacer le label par défaut ou celui défini lors de la création du formulaire dans le contrôleur. Elle permet également d'ajouter les mêmes options que celles vues pour la création du formulaire :

```
{{ form_start(nomFormulaire) }}
  {{ form_label(nomFormulaire.content, 'Contenu de la tâche', {'label
_attr': {'class': 'tache'}}) }}
{{ form_end(nomFormulaire) }}
```

Les fonctions `form_help()` et `form_errors()`

La fonction `form_help()` permet d'afficher la partie aide de l'élément de formulaire :

```
{{ form_help(nomFormulaire.content) }}
```

Cela peut être utile pour afficher l'aide à un autre endroit ou lui ajouter une classe.

La fonction `form_errors()` permet d'afficher les erreurs d'un élément de formulaire ou des erreurs globales qui ne sont pas attachées à un champ :

```
{{ form_errors(nomFormulaire.content) }}

{{ form_errors(nomFormulaire) }}
```

La fonction `form_widget()`

La fonction `form_widget()` permet d'afficher la partie `widget` de l'élément, à savoir l'élément `HTML` du champ comme par exemple un `input`.

Vous pouvez ajouter toutes les options du disponible pour le `widget`, par exemple en ajoutant une classe :

```
{{ form_widget(nomFormulaire.content, {'attr': {'class': 'contenu'}}) }}
```

Maintenant que nous avons vu ces fonctions, nous allons pouvoir attaquer les thèmes.