Créer un formulaire dans Symfony

[sommaire]

1. ## Objectifs

- Apprendre à créer un formulaire réactif dans symfony avec Bootstrap.
- Etre capable d'insérer des données de formulaire dans la base de données à l'aide de laravel 8.

- ## Présentation

  - Un formulaire Web est un espace dédié sur une page web sur lequel un internaute peut saisir les informations demandées. Il peut être composé d’un champ de texte, de menus déroulants, de cases à cocher ou tout autre moyen qui permet de donner une information. Il y a différents types de formulaires, mais le plus souvent on rencontre des formulaires d’inscription, de contact et de commande. Bien sûr, il peut revêtir toute forme ou format. Les informations récoltées sont souvent stockées en base de données.
  - Les formulaires offrent une véritable interaction avec l’utilisateur. Non seulement elles apportent des informations précieuses mais permettent aussi d’évaluer et de mesurer son intérêt pour le contenu du site web. Pour l’internaute, remplir un formulaire nécessite de la confiance. Cela engage le site à mettre en place une sécurisation des données et parfois de les crypter ou les hacher comme les mots de passe par exemple.
  - Les formulaires Symfony sont des objets PHP qui représentent des formulaires HTML. Ils fournissent une abstraction permettant de gérer facilement les données de formulaire, la validation et l'affichage. Voici les étapes de base pour travailler avec les formulaires en Symfony :
    - **Création d'un formulaire** : Utilisez les classes de formulaire Symfony pour créer un formulaire en définissant ses champs et ses options.
    - **Affichage du formulaire** : Affichez le formulaire dans vos vues Twig à l'aide de fonctions d'aide spéciales.
    - **Traitement des données du formulaire** : Récupérez et traitez les données soumises par le formulaire dans votre contrôleur Symfony.
    - **Validation des données** : Validez les données du formulaire pour vous assurer qu'elles sont correctes avant de les enregistrer dans votre base de données ou de les utiliser dans votre application.

apcpedagogie.com

- **Personnalisation du formulaire** : Personnalisez l'apparence et le comportement de votre formulaire en utilisant des options et des thèmes.

L'installation dans un projet Symfony

- Dans les applications utilisant Symfony Flex, qui est facultatif même s'il est installé par défaut depuis Symfony 4, vous pouvez installer Symfony Form avec la ligne de commande suivante : `composer require Symfony/form`

1. **Type de formulaires et Type de champs**

- Un "Type" de formulaire est une classe PHP. Il permet de construire un formulaire et de définir les différents types de champs. Cette classe, si vous utilisez les outils Symfony comme MakerBundle, sera créée dans le dossier Form et portera par convention le nom de l'entité suivi du mot « Type ». Ainsi un formulaire pour une entité User se nommera « UserType » et aura le chemin d'accès : App\Form\UserType.
- Cette classe devra étendre la classe **AbstractType**, ce qui lui donnera 2 méthodes `buildForm()` et `configureOptions()`.

- **BuildForm()**

- **BuildForm()** dispose d'un constructeur de formulaires $builder (nom de variable donné par défaut) qui est un objet **FormBuilderInterface**.
- **$builder** possède donc quelques méthodes dont une indispensable qui est `add()`. La méthode `add()` va permettre d'ajouter tous les champs que l'on souhaite avoir dans le formulaire. Pour cela, il faudra mettre en premier paramètre le nom du champ et son type.

1. **Exemple de création avec `BuildForm()`**

```
// src/Form/ExampleFormType.php

namespace App\Form;

use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\TextType;
use Symfony\Component\Form\FormBuilderInterface;
```

apcpedagogie.com

```
class ExampleFormType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options)
    {
        $builder
            ->add('fieldName', TextType::class, [
                'label' => 'Field Label',
                // Additional options can be added here
            ]);
    }
}
```

- Dans cet exemple, Example `FormType` est la classe **form**. Il ajoute un seul champ de texte nommé **fieldName** au formulaire.

- **Créer un contrôleur pour gérer le formulaire**

  - Ensuite, vous pouvez créer un contrôleur pour gérer le formulaire :

```
// src/Controller/ExampleController.php

namespace App\Controller;

use App\Form\ExampleFormType;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class ExampleController extends AbstractController
{
    /**
     * @Route("/example", name="example")
     */
    public function example(Request $request): Response
    {
        $form = $this->createForm(ExampleFormType::class);

        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {
```

apcpedagogie.com

```
            // Process form submission
            // For example, save data to the database
            // Redirect or render response
        }

        return $this->render('example/index.html.twig', [
            'form' => $form->createView(),
        ]);
    }
}
```

- Dans ce contrôleur, la méthode **example()** crée une instance du formulaire à l'aide de $this->createForm(), gère la soumission du formulaire et restitue le modèle de formulaire.

- **Créer un modèle pour afficher le formulaire**

  - Enfin, vous devez créer un modèle pour afficher le formulaire :

```
{# templates/example/index.html.twig #}

{% extends 'base.html.twig' %}

{% block title %}Example Form{% endblock %}

{% block body %}

<h1>Example Form</h1>

    {{ form_start(form) }}
    {{ form_row(form.fieldName) }}
    {{ form_end(form) }}
{% endblock %}
```

  - Dans ce modèle, **form_start**, **form_row** et **form_end** sont des fonctions Twig utilisées pour restituer différentes parties du formulaire.

- **configureOptions()**

apcpedagogie.com

- Dans Symfony, la méthode **`configureOptions()`** est utilisée pour configurer les options d'un formulaire.
- Cela inclut la définition des options requises, les valeurs par défaut, la validation, la normalisation, etc.
- Déclarer l'entité sur laquelle vous travaillez dans le **formType** dans **configureOptions()** est une bonne pratique, bien que cela soit optionnel. Cela peut rendre votre code plus clair et éviter des problèmes potentiels à mesure que votre application se développe.

1. **Exemple**

- Voici un exemple de comment vous pouvez utiliser **configureOptions()** dans un **formType** :

```
// src/Form/ExampleFormType.php

namespace App\Form;

use App\Entity\ExampleEntity; // Importez votre entité ici
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class ExampleFormType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options)
    {
        $builder
            ->add('fieldName')
            // Ajoutez d'autres champs ici
            ;
    }

    public function configureOptions(OptionsResolver $resolver)
    {
        $resolver->setDefaults([
            'data_class' => ExampleEntity::class, // Définissez l'entité associée au
formulaire
            // Définissez d'autres options par défaut ici
        ]);
    }
}
```

apcpedagogie.com

- Dans cet exemple, **ExampleFormType** configure l'option **data_class** dans **configureOptions()** pour indiquer à Symfony quelle entité est associée à ce formulaire. Cela permet à Symfony de gérer automatiquement la liaison des données du formulaire à l'entité lors de la soumission du formulaire.
- Il est recommandé de spécifier data_class pour éviter des erreurs potentielles de liaison des données, notamment lorsque vous travaillez avec des entités et des formulaires dans Symfony.

- ## Réalisation

Sommaire du cours Symfony

Source : https://apcpedagogie.com/creer-un-formulaire-dans-symfony · téléchargé le 03/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com