

REUNION D'INFORMATION PRF RÉGION OCCITANIE

Suivez-nous... in www.linkedin.com/school/adrarnumerique





- > INFRASTRUCTURES SYSTÈMES & RÉSEAUX
- > CYBERSÉCURITÉ INFRASTRUCTURES & APPLICATIONS
- > **DEVOPS / SCRIPTING** & AUTOMATISATION
- > **DEVELOPPEMENT** WEB & MOBILE
- > TRANSFORMATION NUMERIQUE DES ENTREPRISES

www.adrar-numerique.com



















Formulaires avancés Symfony



Nous avons déjà vu les formulaires se construisant à l'aide du form builder.

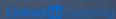
Nous allons voir comment personnaliser les champs avec des attributs et des types différents:

```
UtilisateurType.php
->add('prenom', TextType::class, [
            'attr' => ['class' => 'form-control'], // on appliquera la classe "form-control" à notre input
->add('permis', FileType::class, [
            'mapped' => false, // on ignore l'input lors de la lecture/écriture de l'objet
             'required' => false, // on rend le champs non obligatoire
             'attr' => [
                 'class' => 'form-file',
                 'accept' => 'image/*' // on décide de n'accepter que les fichiers de type image
             'multiple' => false // on ne rend pas possible la multi-sélection
->add('age', IntegerType::class, [
            'attr' => ['min' => '18'], // on appliquera un minimum de 18 en valeur à notre input
```















Formulaires avancés Symfony



Pour personnaliser les labels, nous procéderons comme ci-après:

Vous avez la possibilité de charger le formulaire de cette autre manière plutôt que ligne par ligne

Sinon, vous pouvez directement appeler la fonction "form" qui générera les éléments automatiquement par rapport au **BuilderForm**

```
{{ form(registrationForm) }} inscription.html.twig
```



















Formulaires avancés Symfony



Nous avons déjà vu les formulaires se construisant à l'aide du form builder.

lci nous allons voir comment aller encore plus loin dans la confection de ceux-ci. Par exemple, imaginons que nous voulions avoir accès à une entité B dans un formulaire d'ajout d'une entité A:

Avec ce code, vous verrez que votre formulaire possède désormais une liste déroulante de votre entité.

NB: n'oubliez pas de changer ce qui est entre crochets

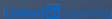
















Insertion d'image (1/4)
Symfony



Si côté Twig vous utilisez la fonction **form_widget()** vous n'avez rien d'autre à faire de ce côté sinon vous devez ajouter un nouveau **form_row()** pour ce nouvel input.

















Insertion d'image (2/4) Symfony



On définit un répertoire avec un nom dans lequel nos fichiers seront envoyés:

config/services.yaml parameters:

avatar_directory: '%kernel.project_dir%/public/uploads/avatars'

Ensuite, nous allons traiter notre donnée dans notre contrôleur. Lors de la vérification de la validité du formulaire, si cette condition est vérifiée nous allons ajouter ce code:

```
UtilisateurController.php
use Symfony\Component\HttpFoundation\File\Exception\FileException;
use Symfony\Component\String\Slugger\SluggerInterface; // Ajoutez ce use
#[Route('/utilisateur/ajout', name: 'app utilisateur ajout')]
  public function ajout(Request $request, SluggerInterface $slugger) { // Ajoutez le slugger en paramètre de votre méthode
    // Ajoutez également EntityManagerInterface $entityManager (plus <u>use</u>) si vous n'utilisez pas les repositories, sinon passez le
repository concerné en paramètre
    if ($form->isSubmitted() && $form->isValid()) {
      $avatarFichier = $form->get('brochure')->getData(); // On récupère les données qui composent l'image
      if ($avatarFichier) { // Si une image a bien été insérée (vu qu'elle n'est pas required, elle peut être vide)
         $originalFilename = pathinfo($avatarFichier->getClientOriginalName(), PATHINFO FILENAME); // On prend le nom de base du fichier
         // this is needed to safely include the file name as part of the URL
         $safeFilename = $slugger->slug($originalFilename);
         $newFilename = $safeFilename.'-'.uniqid().'.'.$avatarFichier->guessExtension();
```

















Insertion d'image (3/4)
Symfony



Ensuite, nous allons traiter notre donnée dans notre contrôleur. Lors de la vérification de la validité du formulaire, si cette condition est vérifiée nous allons ajouter ce code:

```
// Tente de déplacer le fichier vers le répertoire définit plus tôt
                                                                                                               UtilisateurController.php
       try {
           $avatarFichier->move(
               $this->getParameter('avatar_directory'),
               $newFilename
        } catch (FileException $e) {
           // Gérer le cas en cas d'exception levée (droits insuffisants, stockage insuffisant, ...)
        $utilisateur->setImage($newFilename); // On redéfinit l'image de notre objet pour permettre l'enregistrement du bon nom d'image
en BDD
      $utilisateurRepository->save($utilisateur); // Version avec le UtilisateurRepository
      $entityManager->persist($utilisateur); // Version avec l'EntityManager
      $entityManager->flush(); // Version avec l'EntityManager
      return $this->redirectToRoute('app_utilisateur_profil');
   return $this->render('utilisateur/ajout.html.twig', [
        'form' => $form,
    ]);
```

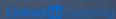
















Insertion d'image (4/4) Symfony



Enfin, pour afficher votre image, vous devrez à nouveau vous servir de la fonction **asset()** avec le lien du dossier et la concaténation du message

```
tilisateur/moncompte.html.twig

img src="{{ asset('uploads/avatars/' ~ utilisateur.avatar) }}" alt="Avatar de l'utilisateur">

# ... #}
```







