Documentation Symfony

Sommaire

1Form : pour la création de formulaire 2
2Les Routes avec Annotations 3

1 Form : pour la création de formulaire

Symfony permet l'utilisation de form, celui-ci permet de crée des formulaires simplement dans le Controller. Pour cela, nous avons, crée une variable qui prend la valeur

$this->createFormBuilder($EntitéConcernée)

On lui ajoute ensuite les champs que l'on veut.

$this->createFormBuilder($EntitéConcernée)
->add('Attribut1', TextType::class)
->add('Attribut2', IntegerType::class)

Les champs doivent avoir les noms d'attributs de la table de l'entitée concernée. Exeception faite des inputs de type Submit.

Suite à ça, on ajoute la ligne « handleRequest($objetRequest) ; »
Celle-ci permet de gérer l'envoi du formulaire.

Enfin, il nous suffit de décider ce qui est à faire une fois le formulaire envoyé.
La commande :
If($form->isSubmitted() && $form->isValid())

le isSubmitted() identifie quand le formulaire est envoyé et le isValid() vérifie que tous les champs sont corrects.

Dans le if, on fait le traitement puis on renvoie (Ou non) vers une autre méthode/page ou autre.

Exemple de Formulaire utilisant form :

```php
$em = $this->getDoctrine()->getManager();
$test = $em->getRepository(Test::class)->find($id);

if(!$test){
    return $this->redirectToRoute('test', array('message'=>'Ce test n\'existe pas'));
}
$form = $this->createFormBuilder($test,['attr' => ['class' => 'm-form m-form--fit m-form--label-align-right']]
    ->add('nom', TextType::class, array('data' => $test->getNom()))
    ->add('save', SubmitType::class)
    ->getForm();
$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {
    $newName = $form["nom"]->getData();
    $test->setNom($newName);
    $em->flush();

    return $this->redirectToRoute('test');
}

return $this->render('test/updateTest.html.twig', array('form' => $form->createView()));
```

2 Les Routes avec Annotations

Annotations permet de crée des routes très simplement. Il suffit pour cela de mettre un commentaire d'une certaine façon (/* ….... */) et de mettre l'URL de la route après @Route

Exemple :

```
/*
 * @Route('/index')
*/
```

Annotations à cependant certaines options intéressantes.
On peut assigner un nom à la route afin de pouvoir y accéder même si celle-ci change d'URL.
Aussi, On peut définir un paramètre de l'URL en tant que variable. Par exemple, si j'ai une route :

```
/index/{variable}
```

/index/2 et /index/test me renverrons vers cette même route. L'intéret de cela est que l'on peut récupérer ces variables.
Ainsi, on en écrivant :

```
/*
* @Route('/index/{v}')
*/
function exemple($v){
…..}
```

On peut récupérer la variable de l'URL.

Enfin, on peut spécifier quelle type de variable on attend. De cette manière on peut avoir deux URL avec la même route mais pas avec la même variable qui revoie à deux routes différentes.
On spécifie à l'aide de l'option « requirements »

On ecrit donc : requirements={'page'='\d+'}

\d+ permet de récupérer les urls entrées avec des chiffres ou nombres comme variables.