esprit
Se former autrement

Validation des données

UP Web

AU: 2019/2020

United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
CONFÉRENCE DES GRANDES ÉCOLES
EXCELLENCE FOR A COMPLEX WORLD
European Accreditation of Engineering Programmes EUR-ACE®
délivrée par CTI Commission des Titres d'Ingénieur
cdio
CONCEIVE DESIGN IMPLEMENT OPERATE

Plan

- Validation des Données : c’est quoi ?
- Validation : Symfony
- Validation Basique
- Validation des chaines de caractère
- Validation des comparaisons
- Validation des dates
- Validation divers

Objectifs

- Apprendre à définir qu'une entité est valide ou pas
- Comprendre la validation des données d’un formulaire côté serveur.
- Se familiariser avec les différents types de validation .

Valider les données d'un formulaire ?

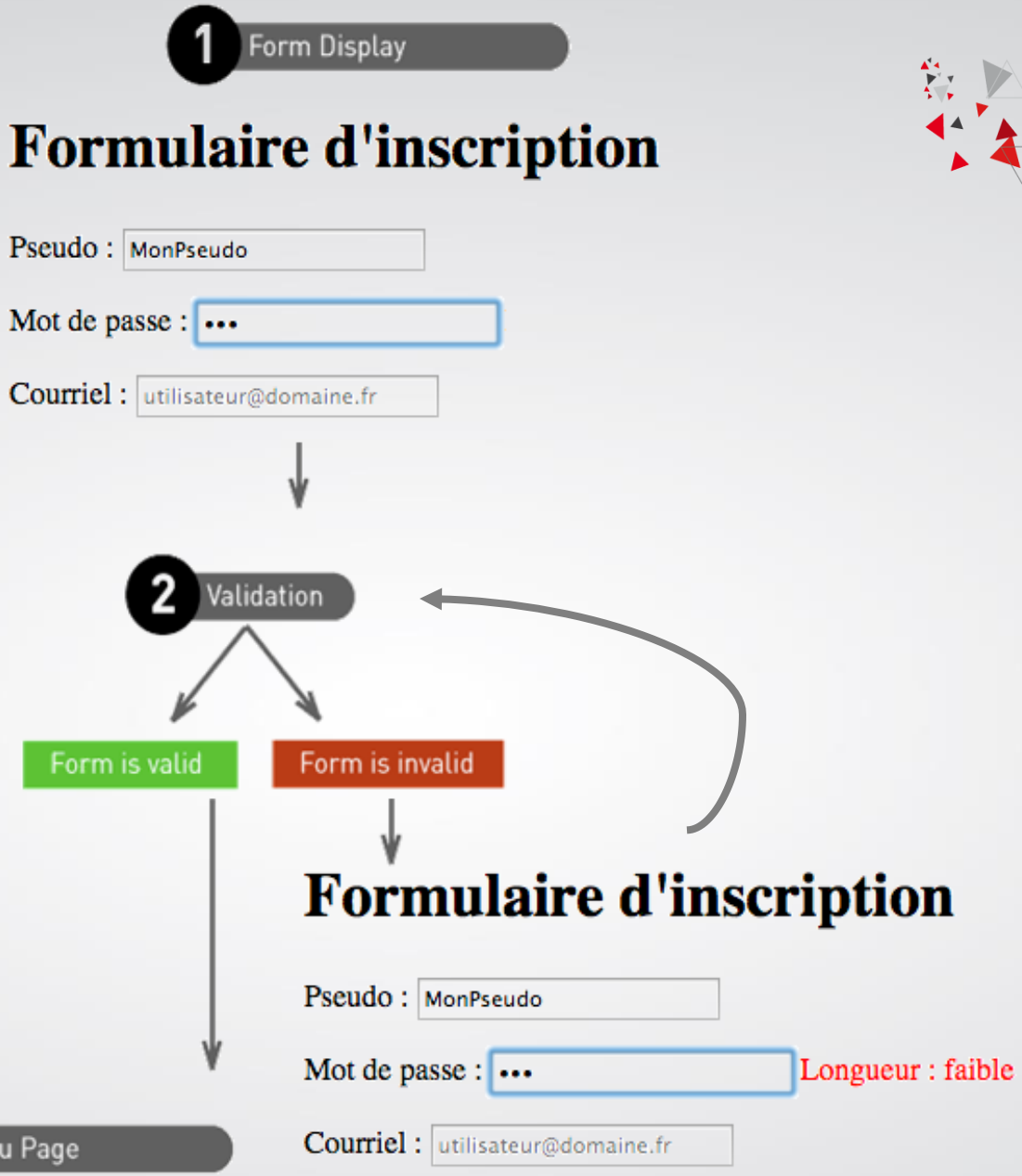
Never trust user input

- La validation est normalement un des premiers réflexes à avoir lorsque l'on demande à l'utilisateur de remplir des informations
- vérifier ce qu'il a rempli ! Il faut toujours considérer que l'utilisateur ne sait pas remplir un formulaire

Valider les données d'un formulaire ?

- S'assurer que les entrées de l'utilisateur sont correctes par rapport à ce qui est déclarée dans l'entité.

La validation des données : Vérification des données côté serveur.


1 Form Display
Formulaire d'inscription
Pseudo : MonPseudo
Mot de passe : •••
Courriel : utilisateur@domaine.fr
2 Validation
Form is valid
Form is invalid
Formulaire d'inscription
Pseudo : MonPseudo
Mot de passe : ••• Longueur : faible
Courriel : utilisateur@domaine.fr
3 Thank You Page

Validation : Symfony

- Vérifier que les données d'un objet sont valides ou non.
- Établir des règles précises pour dire que la valeur d'un attribut (le nom d'utilisateur par exemple doit faire 3 caractères minimum) ou le retour d'une méthode est valide .

→Attacher des règles de validation par rapport aux données dans l'entité : propriété ou méthode .

→ Vérifier les données avant de les enregistrer dans la base de données

Validation : Symfony

- Un formulaire symfony est composé de champs.
- chaque champ est identifié par un nom unique.
- un widget ou un input est associé à chaque champ pour lui permettre d'être afficher à l'utilisateur

->la validation de chaque champ par rapport aux données de l'entité

Validation : Symfony

- **$form->isSubmitted()** : permet de soumettre le formulaire même si les données sont invalides.
- **$form->isValid()** permet de verifier si l'objet contient des données valides ou non.

Contraintes basiques

	Rôle
NotBlank Blank	vérifie que la valeur soumise n'est ni une chaîne de caractères ou un tableau vide, ni NULL. La contrainte Blank fait l'inverse.
NotNull IsNull	vérifie que la valeur est différente de null vérifie que la valeur est égale à null
IsTrue IsFalse	La contrainte IsTrue vérifie que la valeur vaut true, 1 ou "1". La contrainte IsFalse vérifie que la valeur vaut false, 0 ou "0".

```php
class Author
{
    /**
     * @Assert\IsTrue(message="The password cannot match your first name")
     */
    public function isPasswordSafe()
    {
        // ... return true or false
    }
```

Contraintes des chaînes de caractère

	Rôle
Email	vérifie que la valeur est une adresse e-mail valide. La valeur est transformée en chaine de caractères avant la validation
Length	Vérifie que la longeur du chaine de caractère est entre un minimumLength et maximumLength value.
Url	vérifie que la valeur est une adresse URL valide.
Regex	Vérifie que la chaine est confome à l'expression régulière
Ip	vérifie que la valeur est une adresse IP valide.
UserPassword	Vérifie que la valeur saisie dans l'input égale à celle de l'utilisateur connecté

```php
/**
 * @Assert\Email(
 *     message = "The email '{{ value }}' is not a valid email.",
 * )
 */
protected $email;
```

```php
/**
 * @Assert\Regex(
 *     pattern="/\d/",
 *     match=false,
 *     message="Your name cannot contain a number"
 * )
 */
protected $firstName;
```

```php
/**
 * @SecurityAssert\UserPassword(
 *     message = "Wrong value for your current password"
 * )
 */
protected $oldPassword;
```

Contraintes des chaînes de caractère

	Rôle
EqualTo NotEqualTo	Vérifie qu'une valeur est égale à une autre valeur,. Utiliser pour forcer qu'une valeur ne soit pas égale.
IdenticalTo NotIdenticalTo	Vérifie qu'une valeur est identique à une autre valeur définie dans les options. Utiliser pour forcer qu'une valeur ne soit pas identique.

Contraintes de Comparaison

	Rôle
LessThan	Vérifie qu'une valeur est inférieure à une autre valeur définie.
LessThanOrEqual	Vérifie qu'une valeur est inférieure ou égale à une autre valeur définie.
GreaterThan	Vérifie qu'une valeur est supérieure à une autre valeur définie.
GreaterThanOrEqual	Valide qu'une valeur est supérieure ou égale à une autre valeur définie.
Range	Vérifie qu'un numéro donnée ou un objet DateTime object et entre un minimum et un maximum

Contraintes des Dates

	Rôle
Date	vérifie que la valeur est un objet de type Datetime, ou une chaîne de caractères du type YYYY-MM-DD.
DateTime	vérifie que la valeur est un objet de type Datetime, ou une chaîne de caractères du type YYYY-MM-DD HH:MM:SS.
Time	vérifie que la valeur est un objet qui implémente l'interface DateTimeInterface ou une chaîne de caractères du type HH:MM:SS.

Validation des Dates

```php
class Event
{
    /** @Assert\DateTime() */
    private $startDate;

    /**
     * @Assert\DateTime()
     * @Assert\Expression("value > this.startDate")
     */
    private $endDate;

    // ...
}
```

```php
class Event
{
    /** @Assert\DateTime() */
    private $startDate;

    /**
     * @Assert\DateTime()
     * @Assert\GreaterThan(propertyPath="startDate")
     */
    private $endDate;
}
```

esprit
Se former autrement

Divers Contraintes

	Rôle
Choice	Vérifie que la valeur donnée fait partie d'un ensemble donné de choix valides. Il peut également être utilisé pour valider que chaque élément d'un tableau est l'un de ces choix valides.
File Image	vérifie que la valeur est un fichier valide, c'est-à-dire soit une chaîne de caractères qui point vers un fichier existant,
Image	Image vérifie que la valeur est valide selon la contrainte précédente File . Il est également possible de mettre des contraintes sur la hauteur max ou la largeur max de l'image.

```php
const GENRES = ['fiction', 'non-fiction'];

/**
 * @Assert\Choice({"New York", "Berlin", "Tokyo"})
 */
protected $city;

/**
 * You can also directly provide an array constant to the "choices" option in the annotation
 *
 * @Assert\Choice(choices=Author::GENRES, message="Choose a valid genre.")
 */
protected $genre;
```

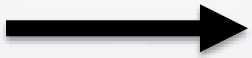
Divers Contraintes

	Role
UniqueEntity	Vérifie qu'un ou des champs particuliers d'une entité Doctrine sont uniques. Ceci est couramment utilisé, par exemple, pour empêcher un nouvel utilisateur de s'enregistrer en utilisant une adresse électronique déjà existante dans le système.
All	Appliquée à un tableau, cette contrainte vous permet d'appliquer une collection de contraintes à chaque élément du tableau.
Valid	activer la validation sur les objets incorporés en tant que propriétés sur un objet en cours de validation. Cela vous permet de valider un objet et tous les sous-objets qui lui sont associés.
Count	que le nombre d'éléments d'une collection donnée est compris entre une valeur minimale et une valeur maximale.

```php
/**
 * @Assert\All({
 *     @Assert\NotBlank,
 *     @Assert\Length(min=5)
 * })
 */
protected $favoriteColors = [];
```

```php
class Address
{
    /**
     * @Assert\NotBlank
     */
    protected $street;

    /**
     * @Assert\NotBlank
     * @Assert\Length(max=5)
     */
    protected $zipCode;
```



```php
/**
 * @Assert\NotBlank
 * @Assert\Length(min=4)
 */
protected $firstName;

/**
 * @Assert\NotBlank
 */
protected $lastName;

/**
 * @Assert\Valid
 */
protected $address;
```

Les contraintes peuvent aussi être définies lors de la création du formulaire.

```php
public function buildForm(FormBuilderInterface $builder, array $options)
{
    $builder
        ->add('myField', TextType::class, [
            'required' => true,
            'constraints' => [new Length(['min' => 3])]
        ])
    ;
}
```

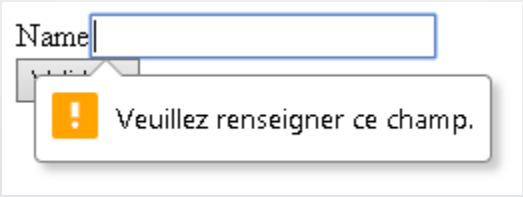
```php
class Author
{
    /**
     * @Assert\NotBlank
     * @Assert\Length(min=3)
     */
    private $firstName;
}
```

→ Créer ses propres validateurs

→ Éviter les failles de sécurité

Validation HTML5

- Les navigateurs qui prennent en charge HTML5 appliquent automatiquement certaines contraintes de validation côté client.
  - La validation la plus courante est activée en rendant un attribut requis sur les champs requis
  - Affichage d'un message de navigateur natif si l'utilisateur essaie de soumettre le formulaire avec des données non valide



- La validation côté client, cependant, doit être désactivée en ajoutant l'attribut novalidate à la balise de formulaire et formnovalidate à la balise de soumission.

→ **l'utilité : tester vos contraintes de validation côté serveur**

```
{{ form_start(form, {'attr': {'novalidate': 'novalidate'}}) }}
{{ form_widget(form) }}
{{ form_end(form) }}
```

```
->add( child: 'validate', type: SubmitType::class,[
    'attr' => ['formnovalidate ' => 'formnovalidate']
]);
```

esprit
Se former autrement