Développement web S2 (R213) – TP #3

Dans cette séance de TP vous allez faire un exercice sur la validation de formulaires.

Liens utiles : CM #1 · TP #1 · TP #2

Dans le répertoire `r213` que vous avez déjà créé, ajoutez un répertoire nommé `r213-tp3` pour sauvegarder les exercices de cette séance (TP #3).
Dans ce répertoire `r213-tp3`, créez un répertoire appelé `exo1` pour le premier exercice, `exo2` pour le deuxième, etc. **Vos exercices ne seront pas corrigés si vous ne respectez pas ces instructions.**

À la fin de la séance, vous devez compresser (en format ZIP ou RAR) le répertoire `r213-tp3` avec tous vos exercices et le soumettre sur AMeTICE.

Seuls les exercices marqués avec une étoile (*) seront évalués. La date limite pour rendre vos exercices est visible sur AMeTICE, et aucun travail ne sera accepté après. Le corrigé sera affiché après l'évaluation des exercices.

Exercices

Exercice 1

Dans cet exercice on va traiter un formulaire avec une adresse e-mail.

Dans un fichier appelé `index.html`, faites un formulaire avec un champ d'adresse électronique.

Dans un fichier PHP appelé `validation.php` vous devez vérifier que c'est une adresse électronique d'Aix Marseille Université, c'est-à-dire qu'elle finit par « `@univ-amu.fr` » ou par « `@etu.univ-amu.fr` ». Pour cela, vous aurez besoin de la fonction `substr()` [doc].

Si c'est bon, affichez un message de réussite. Sinon, affichez un message d'erreur et un lien vers le formulaire

---

La fonction `substr()` (substring) permet d'extraire la fin d'une chaîne de caractères. Par exemple :

```php
<?php
$chaine = 'mon numéro étudiant est g22085675';
$souschaine = substr($chaine, -9); // extraire les 9 derniers caractères
echo $souschaine; // affiche « g22085675 »
?>
```

Après ce premier exercice, vous allez faire une validation complète d'un formulaire avec les deux méthodes vues dans cette partie du cours.

Exercice 2*

Dans cet exercice vous allez faire un formulaire simple, le valider, et afficher les données entrées de façon sécurisée.

Dans un fichier appelé `index.html`, faites un formulaire avec trois champs :

1. un champ de texte (`<input type="text">`) pour un nom
2. un champ pour une adresse e-mail (`<input type="email">`)
3. un champ de texte multi-ligne (`<textarea>`) pour un message

Ajoutez des libellés à chaque champ avec la balise `<label>`.

Dans un fichier PHP appelé `validation.php`, vous devez vérifier que les données entrées sont correctes :

- Le nom doit avoir 3—20 caractères. Pour récupérer la taille d'une chaîne de caractères, utilisez la fonction `strlen()` [doc]
- L'adresse e-mail doit être valide
- Le message doit avoir moins de 400 caractères

Si toutes les trois conditions sont validées, affichez-les sous la forme suivante :

```html
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>R213</title>
</head>
<body>
  <h1>TP #3 / Exo 2</h1>
  <p>
    <a href="mailto:bidule@gmail.com">Carlito</a> a dit : <q>Ils sont de retour. Simple, efficace et très beau je trouve.</q>
  </p>
</body>
</html>
```

Si au moins une des conditions n'est pas validée, affichez un seul message d'erreur approprié (nom beaucoup trop court ou trop long, message trop long) et un lien vers la page avec le formulaire.

Exercice 3

Dans cet exercice vous allez modifier l'exercice 2 pour faire un traitement légèrement différent.

S'il y a des erreurs dans les données du formulaire, il faut les afficher toutes, avec le lien vers le formulaire.

De plus, pour sécuriser votre page web, vous allez utilise la fonction `htmlspecialchars()` [doc] avant d'afficher les données du formulaire. Vérifiez que ça marche en rentrant le message suivant dans le formulaire : `<script>for(let i=0;;i++) alert(`Error (${i})`)</script>`.

Le code HTML des deux pages web doit être valide pour W3C Markup Validation Service.

Notez que si vous avez bien sécurisé votre formulaire, le script précédent ne devrait avoir aucun effet sur votre page web. Essayez de voir ce qui se passe sinon.