Gestion des formulaires PHP

Dans ce chapitre, vous allez apprendre à gérer les formulaires avec PHP.

PHP - Un formulaire HTML simple

Les superglobales PHP **$_GET** et **$_POST** sont utilisées pour collecter les données de formulaire.

L'exemple ci-dessous affiche un formulaire HTML simple avec deux champs de saisie et un bouton d'envoi :

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

Lorsque l'utilisateur remplit le formulaire ci-dessus et clique sur le bouton Soumettre, les données du formulaire sont envoyées pour être traitées dans un fichier PHP nommé "welcome.php". Les données du formulaire sont envoyées avec la méthode HTTP **POST**.

Pour afficher les données soumises, vous pouvez simplement faire écho à toutes les variables. Le fichier "welcome.php" ressemble à ceci :

```
<html>
<body>

Bienvenue <?php echo $_POST["name"]; ?><br>
Votre adresse email est: <?php echo $_POST["email"]; ?>

</body>
</html>
```

La sortie pourrait ressembler à ceci

```
Bienvenue John
Votre adresse email est: john.doe@example.com
```

Le même résultat peut également être obtenu en utilisant la méthode HTTP GET :

```
<html>
<body>

<form action="welcome_get.php" method="get">
Nom: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

et "welcome_get.php" ressemble à ceci :

```
<html>
<body>

Bienvenue <?php echo $_GET["name"]; ?><br>
Votre adresse email est: <?php echo $_GET["email"]; ?>

</body>
</html>
```

Le code ci-dessus est assez simple. Cependant, il manque la chose la plus importante. Vous devez valider les données du formulaire pour protéger votre script contre les codes malveillants.

Pensez SÉCURITÉ lors du traitement des formulaires PHP !

Cette page ne contient aucune validation de formulaire, elle montre simplement comment vous pouvez envoyer et récupérer des données de formulaire.

Cependant, les pages suivantes montreront comment traiter les formulaires PHP en toute sécurité ! Une bonne validation des données du formulaire est importante pour protéger votre formulaire contre les pirates et les spammeurs !

GET ou POST

GET et **POST** créent tous deux un tableau (par exemple array(key1 => value1, key2 => value2, key3 => value3, ...)). Ce tableau contient des paires clé/valeur, où les clés sont les noms des contrôles de formulaire et les valeurs sont les données d'entrée de l'utilisateur.

GET et POST sont traités comme **$_GET** et **$_POST**. Ce sont des variables superglobaux, ce qui signifie qu'ils sont toujours accessibles, quelle que soit leur portée - et vous pouvez y accéder à partir de n'importe quelle fonction, classe ou fichier sans rien faire de spécial.

$_GET est un tableau de variables transmis au script courant via les paramètres d'URL.

$_POST est un tableau de variables transmis au script courant via la méthode HTTP POST.

Quand utiliser GET ?

Les informations envoyées depuis un formulaire avec la méthode GET sont visibles par tous (tous les noms et valeurs des variables sont affichés dans l'URL). GET a également des limites sur la quantité d'informations à envoyer. La limite est d'environ 2000 caractères. Cependant, comme les variables sont affichées dans l'URL, il est possible de marquer la page. Cela peut être utile dans certains cas.

GET peut être utilisé pour envoyer des données non sensibles.

Remarque : GET ne doit JAMAIS être utilisé pour envoyer des mots de passe ou d'autres informations sensibles !

Quand utiliser POST ?

Les informations envoyées à partir d'un formulaire avec la méthode POST sont invisibles pour les autres (tous les noms/valeurs sont intégrés dans le corps de la requête HTTP) et n'ont aucune limite sur la quantité d'informations à envoyer.

De plus, POST prend en charge des fonctionnalités avancées telles que la prise en charge de l'entrée binaire en plusieurs parties lors du téléchargement de fichiers sur le serveur.

Cependant, comme les variables ne sont pas affichées dans l'URL, il n'est pas possible de marquer la page

Les développeurs préfèrent POST pour envoyer des données de formulaire.

Ensuite, voyons comment nous pouvons traiter les formulaires PHP de manière sécurisée dans les chapitres suivants !