PHP : Poster vers la même page

Voici un **article simple, détaillé, précis et concis** pour apprendre à **poster un formulaire PHP vers la même page** et à **vérifier si le formulaire a été soumis**.

Il est courant en PHP d'envoyer les données d'un formulaire **vers la même page**. Cela permet d'afficher le formulaire et son traitement au même endroit, ce qui simplifie la structure du projet.

🔁 Comment ça fonctionne ?

1. La page contient **le formulaire HTML**.
2. Quand l'utilisateur clique sur "Envoyer", les données sont **envoyées à la même URL**.
3. Le code PHP **au début de la page** détecte si le formulaire a été soumis et affiche un résultat.

📌 Exemple de base

```php
<?php
// Traitement du formulaire
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $nom = $_POST["nom"];
    echo "Bonjour, $nom !";
}
?>

<form method="post" action="">
    <label>Votre nom :</label>
    <input type="text" name="nom">
    <input type="submit" value="Envoyer">
</form>
```

🔍 Explication

- `method="post"` : indique qu'on envoie les données avec la méthode POST.
- `action=""` : signifie "envoie le formulaire vers **cette même page**".
- `$_SERVER["REQUEST_METHOD"]` : contient le type de requête HTTP. Si c'est `"POST"`, alors le formulaire a été envoyé.
- `$_POST["nom"]` : contient la valeur entrée par l'utilisateur.

✅ Comment savoir si le formulaire a été soumis ?

```php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    // Le formulaire a été soumis
}
```

Cela permet de séparer clairement :

- le **traitement** (à faire seulement quand on clique sur "Envoyer")
- l'**affichage du formulaire** (toujours visible)

💡 Bonnes pratiques

- Vérifie toujours que les champs existent avec `isset()` avant d'y accéder.
- Utilise `htmlspecialchars()` pour éviter l'injection HTML lors de l'affichage.
- Tu peux aussi utiliser une variable `$erreur` ou `$message` pour afficher des retours à l'utilisateur.

🧪 Exemple amélioré

```php
<?php
$message = "";
if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST["nom"])) {
    $nom = htmlspecialchars($_POST["nom"]);
    $message = "Bonjour, $nom !";
}
?>

<form method="post" action="">
    <label>Votre nom :</label>
    <input type="text" name="nom">
    <input type="submit" value="Envoyer">
</form>

<?php
if ($message !== "") {
    echo "<p>$message</p>";
}
?>
```

🧠 À retenir

- `action=""` ou `action="<?= $_SERVER['PHP_SELF'] ?>"` poste vers **la même page**.
- On vérifie la soumission avec `$_SERVER["REQUEST_METHOD"]`.
- Toujours valider et sécuriser les données avec `htmlspecialchars()`.

Posté ou pas ?

Tu verras parfois dans certains exemples que on utilise plutôt ceci:

```php
if (!empty($_POST)) {
   // ...
}
```

Oui, on peut tout à fait utiliser `if (!empty($_POST))` — **mais il y a une nuance importante à connaître**. Voici un comparatif pour bien comprendre :

🔍 `if ($_SERVER["REQUEST_METHOD"] === "POST")`

- ✔️ **Méthode précise** : tu testes **le type de requête HTTP**.
- ✔️ Tu sais **à coup sûr** que le formulaire a été soumis avec `POST`, même si aucun champ n'a été rempli.
- 🛠️ Idéal si tu veux distinguer une requête `GET` d'une requête `POST`.

🔍 `if (!empty($_POST))`

- ✔️ Tu vérifies si **des données ont effectivement été envoyées**.
- ⚠️ Mais si l'utilisateur soumet le formulaire **vide**, `$_POST` sera un tableau vide → donc la condition sera **fausse**, même si on a bien cliqué sur "Envoyer".

📌 Exemple pour comparer

```php
<?php
// Cas 1 : test avec REQUEST_METHOD
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    echo "Formulaire soumis (même vide)";
}

// Cas 2 : test avec !empty
if (!empty($_POST)) {
    echo "Formulaire soumis et des champs ont été remplis";
}
?>
```

✅ Conclusion

Cas	`$_SERVER["REQUEST_METHOD"] === "POST"`	`!empty($_POST)`
Détecter **la soumission du formulaire**	✅	⚠️ Faux si le formulaire est vide
Détecter **que des champs ont été remplis**	❌	✅
Recommandé pour un premier test	✅ Oui	🔸 À combiner avec un autre test

🧠 Recommandation

Tu peux **utiliser les deux ensemble**, comme ceci :

```php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    if (!empty($_POST)) {
        // Des données ont été soumises
    } else {
        // Formulaire soumis vide
    }
}
```