PHP : Fonctions

Les **fonctions** sont un élément fondamental en PHP. Elles permettent d'organiser le code, d'éviter les répétitions et de faciliter la maintenance des applications. Dans cet article, nous allons explorer en détail la **définition des fonctions**, la **portée des variables**, ainsi que le **passage des paramètres par valeur et par référence**. Nous illustrerons chaque concept avec des **exemples concrets et utiles**.

Définition et utilisation des fonctions en PHP

Qu'est-ce qu'une fonction ?

Une fonction est un bloc de code qui exécute une tâche spécifique et qui peut être **réutilisé** à plusieurs endroits dans un programme.

Syntaxe d'une fonction en PHP :

```
1  function nomDeLaFonction() {
2      // Code à exécuter
3  }
```

Exemple simple :

```
<?php
1  function bonjour() {
2      echo "Bonjour, bienvenue sur mon site !";
3  }
4
5  // Appel de la fonction
6  bonjour();
?>
```

Résultat :

```
Bonjour, bienvenue sur mon site !
```

Les fonctions avec paramètres

Une fonction peut accepter **des paramètres** pour effectuer un traitement personnalisé.

Exemple avec un paramètre :

```
<?php
1  function saluer($prenom) {
2      echo "Bonjour, $prenom !<br>";
```

```php
}

saluer("Alice");
saluer("Bob");
?>
```

Résultat :

```
Bonjour, Alice !
Bonjour, Bob !
```

Exemple avec plusieurs paramètres :

```php
<?php
function additionner($a, $b) {
    echo "La somme de $a et $b est " . ($a + $b) . "<br>";
}

additionner(3, 5);
additionner(10, 20);
?>
```

Résultat :

```
La somme de 3 et 5 est 8
La somme de 10 et 20 est 30
```

Valeurs de retour dans une fonction

Une fonction peut **retourner une valeur** avec `return` au lieu d'afficher directement un résultat.

Exemple avec `return` :

```php
<?php
function multiplier($x, $y) {
    return $x * $y;
}

$resultat = multiplier(4, 3);
echo "Le résultat est : $resultat"; // Affiche 12
?>
```

Pourquoi utiliser `return` ?
Cela permet de stocker le résultat dans une variable pour une utilisation ultérieure.

La portée des variables (Scope)

Les variables en PHP ont une **portée** qui définit où elles peuvent être utilisées.

Portée locale

Une variable déclarée **dans une fonction** n'est accessible que dans cette fonction.

Exemple :

```php
<?php
function afficherMessage() {
    $message = "Ceci est un message local.";
    echo $message;
}

afficherMessage();
?>
```

Portée globale

Les variables définies **en dehors d'une fonction** ne sont pas accessibles à l'intérieur d'une fonction, sauf si on utilise `global`.

Exemple :

```php
<?php
$nom = "Alice";

function direBonjour() {
    global $nom; // Permet d'accéder à la variable globale
    echo "Bonjour, $nom !";
}

direBonjour();
?>
```

Définir des valeurs par défaut aux paramètres

Il est possible d'attribuer une **valeur par défaut** à un ou plusieurs paramètres.

- Règle importante : Seuls les **derniers paramètres** peuvent avoir une valeur par défaut.

Exemple avec plusieurs paramètres :

```php
<?php
function afficherMessage($prenom, $langue = "fr") {
    if ($langue == "fr") {
        echo "Bonjour, $prenom !<br>";
    } elseif ($langue == "en") {
        echo "Hello, $prenom !<br>";
    } else {
```

```php
        echo "Salut, $prenom !<br>";
    }
}

afficherMessage("Alice"); // Utilise le français par défaut
afficherMessage("Bob", "en"); // Utilise l'anglais
afficherMessage("Charlie", "es"); // Affiche "Salut, Charlie !"
?>
```

Résultat :

```
Bonjour, Alice !
Hello, Bob !
Salut, Charlie !
```

➡️ **Les valeurs par défaut évitent d'avoir à passer un paramètre systématiquement.**