PHP : fonctions et cookies

Lydia Rodriguez--de la Nava
CNAM
Juin 2024

Inclure un fichier externe

Il existe deux façons d'inclure un fichier PHP dans un autre :

- `require` qui produit une erreur et arrête le script si le fichier n'est pas trouvé ;
- `include` qui produit simplement un warning si le fichier n'est pas trouvé, mais le code continue.

Disons que nous avons un fichier `fonctions_utiles.php` que nous voulons inclure pour utiliser les fonctions que nous avons écrites dedans :

```php
<?php
        require 'fonctions_utiles.php';

        // ou

        include 'fonctions_utiles.php';
?>
```

Les constantes

Pour définir une constante, c'est-à-dire une valeur qui ne peut pas être modifiée pendant l'exécution de notre code, on utilise la fonction `define` :

```php
<?php
        define('MA_CONSTANTE', 2);

        echo MA_CONSTANTE; // affiche 2
?>
```

Les fonctions

Syntaxe

Pour déclarer une fonction en PHP, on utilise le mot-clé `function` suivit du nom de la fonction, et ses arguments entre parenthèse.

Par exemple, la fonction suivante renvoie la somme des deux valeurs passées en argument :

```php
<?php
        function addition($a, $b) {
                return $a + $b;
        }
?>
```

Syntaxe (suite)

Cette autre fonction affiche simplement "Bonjour !" :

```
<?php
        function dire_bonjour() {
                echo "Bonjour !";
        }
?>
```

Arguments par défaut

On peut donner des valeurs par défaut aux arguments d'une fonction, c'est-à-dire que si la fonction est appelée sans avoir précisé de valeur pour cette argument, elle prendra la valeur par défaut.

Pour ce faire, il suffit de donner une valeur à la variable de notre choix directement dans les arguments de la fonction.

Arguments par défaut (exemple)

Par exemple :

```php
<?php
        function dire_bonjour_a($nom = 'Lydia') {
                echo "Bonjour $nom";
        }

        dire_bonjour_a(); // affiche Bonjour Lydia !

        dire_bonjour_a("Bob"); // affiche Bonjour Bob !
?>
```

Les Cookies

Stocker des informations sur l'utilisateur

On a vu que l'on pouvait stocker des données de l'utilisateur en accédant aux superglobales `$_POST` et `$_GET`.

Néanmoins on a aussi vu qu'il fallait que ces données soient entrées par l'utilisateur dans un formulaire.

Il peut être utile pourtant d'enregistrer des données de navigation de l'utilisateur, pour garder des préférences de l'utilisateur par exemple.

Pourquoi on a besoin des cookies

Ce n'est pas suffisant d'enregistrer ces informations dans des variables normales (celles qui s'écrivent avec un `$`, ie `$var`) puisqu'elles n'existent que dans le fichiers PHP où elles ont été déclarées.

On voudrait un moyen de garder les informations du client quelle que soit la page du site.

Initialiser un cookie avec `setcookie`

La fonction `setcookie` permet de créer simplement un cookie. Il lui faut au moins deux choses :

1. le nom du cookie ;
2. la valeur/le contenu du cookie.

Par exemple :

```
<?php
        setcookie('mon_premier_cookie', 'il est pas mal non ?');
?>
```

Récupérer les cookies

Il existe une supervariable pour récupérer les cookies, qui fonctionne de la même manière que `$_POST` et `$_GET`, c'est-à-dire qu'à la création du cookie, une nouvelle case est ajoutée au tableau `$_COOKIE` avec la clé `'mon_premier_cookie'` et la valeur qui lui correspond :

```php
<?php
        if (isset($_COOKIE['mon_premier_cookie'])) {
                echo $_COOKIE['mon_premier_cookie']; // affiche il est pas mal non ?
        }
?>
```

On fait bien attention de vérifier que le cookie existe bien avec `isset` !!

Modifier un cookie

Pour modifier la valeur d'un cookie, il suffit de rappeler `setcookie` avec la clé du cookie que l'on souhaite modifier, et on lui donne une nouvelle valeur :

```
<?php
                if (isset($_COOKIE['mon_premier_cookie'])) {
                        setcookie('mon_premier_cookie', 'une nouvelle valeur');
        }
?>
```

Date d'expiration d'un cookie

Il n'existe pas de fonction pour supprimer à proprement parler pour supprimer un cookie.

Mais la fonction `setcookie` prend en réalité un troisième argument pour préciser la durée de vie d'un cookie.

Généralement, on utilise la fonction `time()` qui renvoie l'heure actuelle en nombre de secondes depuis le 1er Janvier 1970, à laquelle on ajoute le temps qu'on veut que le cookie existe.

Par exemple, le cookie suivant sera automatiquement supprimé après 1h (ou 3600 secondes) :

```
<?php
        setcookie('id_utilisateur', 'user1', time() + 3600);
?>
```

Suppression d'un cookie

Si on veut supprimer un cookie avant sa fin de vie prévue, on peut le modifier en lui donnant une date dans le passé :

```
<?php
        setcookie('id_utilisateur', 'user1', time() - 3600);
?>
```

Ici on enlève 1h à l'heure actuelle, donc le cookie n'existera plus.

Donner un tableau comme valeur à un cookie

Il est impossible de donner directement un tableau comme valeur à un cookie.

Néanmoins, il existe une fonction `serialize` qui transforme un tableau en une chaîne de caractère, et une fonction `unserialize` qui fait le chemin inverse, c'est-à-dire qui prend une chaîne de caractère qui a été obtenue par `serialize` et qui renvoie le tableau initial.

```php
<?php
        $arr = ['pomme', 'tomate', 'pain de mie'];

        setcookie('liste_fruits', serialize($arr));

        echo print_r(unserialize($_COOKIE['liste_fruits']));
?>
```