Programmation Orienté Objet

PHP – Web

I. Découpage des fichiers

I.1. Inclusion de fichiers externes

Imaginons que nous voulions utiliser une fonction writeMessage() dans toutes les pages d'un site : il faudrait mettre le code de cette fonction dans chacune des pages. Imaginez alors pour un site de 1000 pages : ce n'est clairement pas possible en termes de maintenabilité du code, car il faudrait reporter 1000 fois la moindre modification effectuée dans le code de la fonction writeMessage().

Pour résoudre ce problème, PHP offre un mécanisme : l'inclusion de fichiers. On parle alors de fichier externe.

Créez le fichier PHP suivant, appelons-le hello.php :

```php
<?php
// Fichier 'hello.php'

function writeMessage($text) {
   $html = "<h1>".$text."</h1>";
   echo $html;
}
?>
<!DOCTYPE html>
<html lang="fr">
<head>
    <title>Inclusion de fichiers PHP</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
<?php
writeMessage($message);
?>
<br>
<?php
writeMessage("Bonjour tout le monde !");
?>
</body>
</html>
```

Tout d'abord, nous allons déplacer la fonction writeMessage() dans un second fichier PHP nommé fonctions.php. Ce fichier fonctions.php est un fichier de bibliothèque (ou encore librairie) de code, avec pour seul contenu le code source de la fonction writeMessage() :

```php
<?php
// Fichier 'fonctions.php'

function writeMessage($text) {
   $html = "<h1>".$text."</h1>";
   echo $html;
}

```

Il s'agit donc de factoriser à un seul emplacement le code de fonctions utilisées dans plusieurs pages, cela rejoint la définition même d'une fonction qui est d'être réutilisable.

On pourra bien entendu par la suite ajouter autant de fonctions que nécessaire dans notre fichier de bibliothèque fonctions.php.

Dans le fichier d'origine hello.php, on peut maintenant supprimer le code de la fonction writeMessage() et le remplacer par l'inclusion (chargement ou appel) du fichier fonctions.php via la fonction PHP native include() qui prend en argument le chemin vers le fichier et son nom :

```php
include("fonctions.php");
```

Cette fonction include permet de recopier dans la page le contenu du fichier dont l'URL est passée en paramètre.

Il suffit donc de recopier cette ligne dans toutes les pages où nous voulons utiliser notre bibliothèque de fonctions personnelle.

PHP permet au développeur de créer et de manipuler ses propres fonctions. Pour illustrer ceci, nous allons encore une fois modifier `hello.php` pour définir une fonction d'écriture d'un titre :

```php
<?php
include("fonctions.php");
$message = "Hello world !";
?>
<html>
<head>
    <title>Inclusion de fichiers PHP</title>
</head>
<body>
<?php
writeMessage($message);
?>
<br>
<?php
writeMessage("Bonjour tout le monde !");
?>
<script src="js/scripts.js"></script>
</body>
</html>
```

I.2. Découpage d'une page HTML

Non seulement vous allez trouver sur le web des bibliothèques de fonctions libres de droits à inclure dans vos programmes, mais vous allez pouvoir les utiliser pour découper du simple code HTML en plusieurs fichiers.

Par exemple, vous pourriez découper une page HTML de la façon suivante : en-tête, contenu principal et pied de page :

fichier `index.php`

```php
<?php
include("entete.php");
<body>
 page de test
</body>
include("pieddepage.php");

```

fichier `entete.php`

```php
<!DOCTYPE html>
<html lang="fr">
<head>
   <meta charset="utf-8">
   <title>Inclusion de fichiers PHP</title>
   <link rel="stylesheet" href="css/style.css">
</head>
```

fichier `pieddepage.php`

```php
<script src="js/scripts.js"></script>
</body>
</html>
```

II. Les différentes fonctions d'inclusion

PHP fournit 4 fonctions d'inclusion de fichiers :

- include() : lève une erreur de type avertissement (warning), c'est-à-dire qui ne bloque pas l'exécution du code suivant l'appel de la fonction include().
- require() : lève une erreur dite fatale, le script s'arrête PHP là.
- include_once() : pareil que pour include() mais le fichier n'est chargé qu'une seule fois, lors du premier appel dans le site
- require_once() : pareil que pour require() mais le fichier n'est chargé qu'une seule fois, lors du premier appel dans le site

Compléments :

Différences entre include() et require()

La gestion des erreurs en PHP

III. Vérification de l'existence d'un fichier

Dans le cadre d'une inclusion de fichier, il faut s'assurer que le fichier à inclure existe bien. Ceci se fait avec la fonction PHP file_exists(), à laquelle on passe le chemin du fichier dont on veut vérifier l'existence :

```php
<?php
if (file_exists("chemin/entete.php") )
{
    include("chemin/entete.php");
}
else
{
     // Erreur (à gérer)
}

```

IV. Exercice

Reprendre une ancienne page HTML et découpez la comme ci-dessus.

V. Inclusion de fonction PHP dans les pages HTML

V.1. Affichage simple

Soit le code HTML suivant permettant l'affichage d'employés

index.html

```html
<!DOCTYPE html>
<html>

<head>
    <title>Employés</title>
    <link href="../css/style.css" rel="stylesheet" />
</head>

<body>
    <div id="contenu">
        <div class="employe">
            Employe
        </div>
        <div class="employe">
            Employe 2

        </div>
        <div class="employe">
            Employe 3
        </div>
        <div class="employe">
            Employe 4

        </div>
    </div>
</body>

</html>
```

Remplaçons maintenant les valeurs « en dur » Employe, Employe2, … par des variables PHP créer avec les exercices Orienté Objet vu précédemment.

Dans la première partie du fichier, on définit les variables de type Employe

index.php

```php
<?php
Require("Employe.class.php");

$dateEmbauche1 = new DateTime("11-03-2001");
$dateEmbauche2 = new DateTime("03-06-1997");
$dateEmbauche3 = new DateTime("27-04-1999");
$dateEmbauche4 = new DateTime("09-09-1978");
$dateEmbauche5 = new DateTime("17-11-2011");

$employe1 = new Employe(["nom"=>"Ducoin", "prenom"=>"Ludovic", "dateEmba
$employe2 = new Employe(["nom"=>"Marc", "prenom"=>"Isabelle", "dateEmbau
$employe3 = new Employe(["nom"=>"Toto", "prenom"=>"Michael", "dateEmbauc
$employe4 = new Employe(["nom"=>"Duchemin", "prenom"=>"Lucas", "dateEmba
$employe5 = new Employe(["nom"=>"Dupont", "prenom"=>"Elliot", "dateEmbau

?>
```

Dans la 2eme partie, on recopie le code HTML en incluant des parties de PHP

```php
?>
<!DOCTYPE html>
<html>

<head>
    <title>Employés</title>
    <link href="../css/style.css" rel="stylesheet" />
</head>

<body>
    <div id="contenu">
        <div class="employe">
            <?php $employe1->afficheEmployeHTML(); ?>
        </div>
        <div class="employe">
        <?php $employe2->afficheEmployeHTML(); ?>

        </div>
        <div class="employe">
        <?php $employe3->afficheEmployeHTML(); ?>
        </div>
        <div class="employe">
        <?php $employe4->afficheEmployeHTML(); ?>

        </div>
    </div>
</body>

</html>

```

Le php et l'HTML s'entremêle. Cette technique peut être utilisée si les parties de PHP sont courtes.

V.2. Transformation de la partie HTML en PHP

Programme équivalent au précédent.

Le HTML est compressé à l'intérieur de balise echo.

index.php

```php
<?php
Require("Employe.class.php");

$dateEmbauche1 = new DateTime("11-03-2001");
$dateEmbauche2 = new DateTime("03-06-1997");
$dateEmbauche3 = new DateTime("27-04-1999");
$dateEmbauche4 = new DateTime("09-09-1978");
$dateEmbauche5 = new DateTime("17-11-2011");

$employe1 = new Employe(["nom"=>"Ducoin", "prenom"=>"Ludovic", "date
$employe2 = new Employe(["nom"=>"Marc", "prenom"=>"Isabelle", "dateE
$employe3 = new Employe(["nom"=>"Toto", "prenom"=>"Michael", "dateEm
$employe4 = new Employe(["nom"=>"Duchemin", "prenom"=>"Lucas", "date
$employe5 = new Employe(["nom"=>"Dupont", "prenom"=>"Elliot", "dateE

echo '<!DOCTYPE html><html><head><title>Employés</title>
<link href="../css/style.css" rel="stylesheet" />
</head><body><div id="contenu">
<div class="employe">';
$employe1->afficheEmployeHTML();
echo '</div><div class="employe">';
$employe2->afficheEmployeHTML();
echo '</div><div class="employe">';
$employe3->afficheEmployeHTML();
echo '</div><div class="employe">';
$employe4->afficheEmployeHTML();
echo '</div></div></body></html>';

```

Utilisation de foreach

index.php

```php
<?php
require "Employe.class.php";

$dateEmbauche1 = new DateTime("11-03-2001");
$dateEmbauche2 = new DateTime("03-06-1997");
$dateEmbauche3 = new DateTime("27-04-1999");
$dateEmbauche4 = new DateTime("09-09-1978");
$dateEmbauche5 = new DateTime("17-11-2011");

$ListeEmployes[] = new Employe(["nom" => "Ducoin", "prenom" =>
$ListeEmployes[] = new Employe(["nom" => "Marc", "prenom" => "
$ListeEmployes[] = new Employe(["nom" => "Toto", "prenom" => "
$ListeEmployes[] = new Employe(["nom" => "Duchemin", "prenom"
$ListeEmployes[] = new Employe(["nom" => "Dupont", "prenom" =>

echo '<!DOCTYPE html><html><head><title>Employés</title>
<link href="../css/style.css" rel="stylesheet" />
</head><body><div id="contenu">';
foreach ($ListeEmployes as $unEmploye) {
    echo '<div class="employe">';
    $unEmploye->afficheEmployeHTML();
    echo '</div>';
}
echo '</div></body></html>';

```

VI. Exercice

Reprendre une page HTML et y injecter des données issues de classes PHP