PHP : Dynamiser un site HTML

Vous avez obtenu la note de 0 sur 12

Le 06/03/2018 à 09:39 vous aviez obtenu la note de 7 sur 12

Que se passe-t-il lorsque le fichier inclus n'existe pas, avec la fonction `include()` ?

Explication:

Avec `include()`, PHP va générer un simple **warning** lorsque le fichier à inclure n'existe pas, mais continuer l'exécution du script

- Une erreur fatal arrêtant le script.
- L'erreur est ignorée.
- VUne erreur de type warning, le script continue.
- Cela génère une exception

Soit le fichier "a.php" qui requiert le fichier "b.php", que le fichier "b.php" requiert le fichier "c.php" et que ce dernier requiert le fichier "a.php", il y aura des inclusions en boucle si on inclut un de ces fichiers. Comment pallier à ce problème ?

Plusieurs réponses sont possibles.

Explication:

Il faut utiliser les fonctions `include_once` et `require_once` qui fonctionnent exactement la même manière que `include` et `require` au détail près que si le fichier est déjà inclus, il ne le sera pas une seconde fois. Donc il n'y a plus de boucle infini d'inclusion.

Nous pouvons aussi rassembler le contenu des 3 fichiers en 1, mais cela risque de complexifier le code or nous voulons l'inverse.

- On ne peut rien y faire

- VOn regroupe les 3 fichiers en 1.

- Il n'y aura pas de problème, PHP sait gérer ce cas de figure

---

- VUtiliser les fonctions `include_once()` ou `require_once()` au lieu de `include()` ou `require()`

Que se passe-t-il lorsque le fichier inclus n'existe pas avec la fonction `require()` ?

Explication:

Avec `require()`, PHP va générer une **erreur fatale** et arrêter le script lorsque le fichier à inclure n'existe pas.

- L'erreur est ignorée.

Une erreur de type warning, le script continue.

Cela génère une exception

VUne erreur fatale arrêtant le script.

Quelle instruction permet de sortir d'une boucle avant que la condition de sortie ne soit valide (ou la boucle terminée) ?

Explication:

L'instruction `break` permet de sortir d'une structure `for`, `foreach`, `while`, `do-while`.

`Vbreak;`

`stop;`

`out;`

`end;`

Soit la boucle suivante

```
for($i=0; $i<10; $i++) {
    echo $i.'<br/>';
    $i--;
}
```

Quelle est l'erreur de cette boucle ?

Explication:

Il faut absolument déterminer une condition de sortie sinon nous risquons de faire planter le serveur avec une boucle infinie.

Dans notre cas, la boucle est infinie car la valeur de `i` ne dépassera jamais 9 *(la condition de sortie)* car nous décrémentons `i` à chaque instruction alors qu'il est incrémenté par l'instruction `for`.

- Il n'y a pas de condition de sortie

- VLa boucle est infinie

- Il n'y a pas d'erreur

- La syntaxe de boucle est incorrecte

Quelle boucle est recommandée pour l'affichage des balises HTML ?

Explication:

La boucle `foreach` est fortement recommandé pour l'affichage des balises HTML car c'est elle permet de parcourir un tableau qui contient des informations essentielles pour les balises et

contenu. On peut mettre toute les détails d'un menu dans un tableau et la parcourir pour l'afficher.

- `for`
- `while`
- `Vforeach`
- `loop`

Soit le code suivant :

```
   function count($monTableau) {
        $i = 0;
        foreach($monTableau as $k => $v) {
            $i++;
        }

        return $i;
    }
```

Que se passe-t-il lors de l'exécution du code ci-dessus?

Explication:

`` `count `` est une fonction native de PHP, elle ne peut pas être ré-déclarer de cette manière.

En effet les noms de de fonction doivent être unique.

Aucune erreur

Un warning pour dire que la fonction `count` existe déjà et ne vas pas être pris en compte

---

VUne erreur fatale car la fonction `count` existe déjà nativement

---

La fonction native `count` sera remplacée par celle là

Comment déclarer une fonction qui prend en argument un nombre et retourne comme résultat si il est pair ou non ?

Explication:

Une fonction doit être déclarée avec le mot clé `function`. Pour retourner une valeur, le mot clé `return` est utilisé.

```
est_pair($nombre)
{
    if ($nombre % 2 == 0) {
        return true;
    }

    return false;
}
```

```
est_pair($nombre)
{
    if ($nombre % 2 == 0) {
        echo true;
    }

    echo false;
}
```

```
function est_pair($nombre)
{
    if ($nombre % 2 == 0) {
        return true;
    }

    return false;
}
```

```
Vfunction est_pair($nombre)
{
    if ($nombre % 2 == 0) {
        echo true;
    }

    echo false;
}
```

```
function dire_bonjour($nom){
    return 'Bonjour '.$nom;
}
```

Comment afficher 'Bonjour Toto' avec ce code?

Explication:

Il faut passer le variable 'Toto' en argument en parenthèse à l'appel de la fonction.

Il ne faut pas oublier de faire un `echo` de la fonction pour afficher le résultat car celle-ci ne fait que retourner la valeur et non l'afficher.

```
dire_bonjour 'Toto';
```

```
dire_bonjour('Toto');
```

```
echo dire_bonjour 'Toto';
```

```
Vecho dire_bonjour('Toto');
```

Dans une fonction, comment fait-on pour accéder à une variable globale sans passer par la variable super globale $GLOBALS ou l'instruction `global` ?

Explication:

Une fonction peut être appelée avec des arguments. Il faut donc passer la variable globale en argument à la fonction.

Au sein de la fonction, une copie locale de la variable globale sera alors faite, mais n'est pas nécessairement nommée de la même façon.

```
$nom_globale = 'Toto';

function bonjour($nom)
{
    echo $nom;
}

// Affiche Toto
bonjour($nom_globale);
```

- On ne peut pas

- Cela est sans intérêt

- Utiliser les variables statiques

- VUtiliser les arguments des fonctions

Comment accéder aux variables globales dans une fonction ?

Plusieurs réponses sont possibles.

Explication:

Avec le mot clé `global`, on peut accéder aux variables globales déclarées en dehors de la fonction. On peut aussi accéder à ces variables avec la variable super globale `$GLOBALS`.

```
$ma_variable_globale = 'coucou';

function test()
```

```
{
    echo $GLOBALS[ma_variable_globale'']; //Affiche coucou
    global $ma_variable_globale;

    echo $ma_variable_globale; // /Affiche coucou aussi
}
```

VAvec le mot clé `global`

VAvec la variable super globale `$GLOBALS`

On doit obligatoirement les passer en arguments.

On utilise les variables statiques

Comment accéder à la valeur d'une variable locale d'une fonction dans un script ?

Explication:

Avec le mot clé `return`, on peut retourner n'importe quelle variable à l'intérieur d'une fonction pour l'utiliser après.

```
function test()
{
    $toto = 'étudiant';

    return $toto;
}
```

```
$toto = test();
echo $toto; //Affiche étudiant
```

Grâce à la super variable $GLOBALS

Grâce à la super variable $LOCALS

On ne peut pas

VII faut retourner la variable locale à la fin de la fonction avec le mot clé `return`