

### **Vous avez obtenu la note de 8 sur 36**

Le 05/03/2018 à 21:57 vous aviez obtenu la note de 27 sur 36

Le 18/04/2018 à 16:07 vous aviez obtenu la note de 0 sur 36

Le 18/04/2018 à 16:11 vous aviez obtenu la note de 11 sur 36

Le 19/04/2018 à 11:21 vous aviez obtenu la note de 12 sur 36

Quel module de XAMPP est nécessaire et doit être démarré pour interpréter un fichier PHP ?

#### **Explication:**

Le module [Apache](#) est le serveur web qui permet de servir les requêtes clients, notamment par l'appel à des fichiers PHP via le navigateur.

Le module [MySQL](#) est le serveur de base de données.

Le module [Filezilla](#) est un serveur FTP pour Windows.

Le module [Mercury](#) est un serveur de mail pour Windows.

Une fois le module Apache démarré, le serveur est accessible à l'adresse `http://localhost` à laquelle on ajoute les répertoires présents dans le répertoire `htdocs` de XAMPP.

Exemple : Le répertoire `C:\xampp\htdocs\mon_projet\` ou  
`/Applications/XAMPP/htdocs/mon_projet/` sera accessible à l'adresse  
`http://localhost/mon_projet/`

VApache

---

MySQL

---

Filezilla

---

Mercury

Dans un environnement XAMPP, quel répertoire contient les projets accessibles via le navigateur ?

**Explication:**

Le répertoire `htdocs` est le répertoire publique de XAMPP accessible via le navigateur. Le répertoire choisi à l'installation de XAMPP déterminera l'emplacement du répertoire `htdocs`.

Exemple : Le répertoire `C:\xampp\htdocs\mon_projet\` ou  
`/Applications/XAMPP/htdocs/mon_projet/` sera accessible à l'adresse  
`http://localhost/mon_projet/`

www

---

---

Vhhtdocs

---

public

---

root

Quel est le fichier de configuration principal d'Apache ?

**Explication:**

C'est dans le fichier de configuration `httpd.conf` qu'on trouve généralement toutes les directives et variables d'environnement du serveur web Apache.

`php.ini` est le fichier de configuration principal de PHP.

`my.ini` est le fichier de configuration principal de MySQL.

apache.conf

---

php.ini

---

Vhttpd.conf

---

my.ini

La balise de fermeture de bloc PHP `?>` doit-elle toujours être présente ?

**Explication:**

La balise de fermeture `?>` n'est requise que si le code qui suit la dernière instruction PHP n'est pas du PHP (du HTML par exemple). Dans le cas d'un fichier PHP inclus qui ne contiendrait que du PHP par exemple, seule la balise d'ouverture `<?php` est nécessaire.

Oui, tout le temps.

---

VNon, pas si le fichier ne contient que du PHP.

---

Non, jamais.

Il est possible d'ouvrir et de fermer plusieurs fois les balises PHP dans un même fichier. Vrai ou faux ?

**Explication:**

Il n'y a aucune limite à ouvrir et fermer les balises aussi souvent que nécessaire. Il est même encouragé de le faire, notamment lorsque le code PHP est imbriqué dans du code HTML.

Faux, on ne peut ouvrir les balises PHP qu'une seule fois

---

VVrai, on peut les ouvrir et les fermer à volonté.

---

---

Partiellement faux, tout dépend de la configuration PHP.

Quelle est la bonne balise pour indiquer que le code qui suit celle-ci est du PHP ?

**Explication:**

La bonne façon d'indiquer l'ouverture d'un bloc d'instructions PHP est avec la balise `<?php`, contrairement à la fermeture qui se fait via `?>`. Il existe également une version courte (`<?`) qu'il est déconseillé d'utiliser puisqu'elle dépend de la configuration PHP et est par défaut désactivée.

`<php>`

---

`∇<?php`

---

`<script type="text/php">`

---

`<<<PHP`

Quel est le site de la documentation officielle de PHP ?

**Explication:**

php.net est le seul domaine par lequel on peut accéder à la documentation officielle de PHP.

php.com

---

---

php.fr

---

Vphp.net

---

php.org

Sur quel site peut-on trouver ou poser une question sur du code et bénéficier très rapidement de l'avis d'une grande communauté internationale ?

**Explication:**

Le site Stack Overflow propose des questions et réponses sur un large choix de thèmes concernant la programmation informatique.

Quand vous rencontrez une problématique courante en intégration/développement, il y a de fortes chances que la question ait été abordée sur Stack Overflow.

À vous ensuite de garder un esprit critique face aux réponses, à bien veiller aux dates de publications et aux commentaires de la communauté.

Quand vous prévoyez de poser une question sur Stack Overflow, vérifiez toujours au préalable que la question n'a pas déjà été posée et résolue, sinon la communauté vous le fera savoir avec une pointe de sarcasme !

W3Schools

---

CodeCademy

---

OpenClassrooms

---

VStackOverflow

À quoi sert la concaténation ?

**Explication:**

La concaténation permet d'assembler au moins deux chaînes de caractères entre elle, qu'elles soient dans des variables ou non.

Le retour de type *string* d'une fonction peut également être concaténé à une chaîne spécifiée.

VÀ assembler au moins deux chaînes de caractères entre elle

---

À enlever des caractères dans une chaîne

---

À compter le nombre de mots dans une chaîne

---

À séparer les mots d'une chaîne en plusieurs variables

Quelle est la bonne syntaxe pour concaténer la variable `$a` et la variable `$b` dans la variable `$c` ?

```
$a = 'Hello ';  
$b = 'World !';  
// $c = ?
```

### Explication:

Le point `.` est l'opérateur de concaténation à utiliser lorsque l'on souhaite concaténer des chaînes de caractères entre elles.

```
$c = $a + $b;
```

---

```
$c = $a . $b;
```

---

```
$c = $a | $b;
```

---

```
$c = $a ++ $b;
```

Que vaut `$c` à la fin du code suivant ?

```
$a = 'John';  
$b = ' !';  
$c = 'Hello ';  
if ($a != NULL) {  
    $c .= $a . $b;  
} else {  
    $c .= 'World' . $b;  
}
```



### Explication:

Ici, `$a` n'est pas égal à `NULL`, la condition du `if` est donc remplie et son bloc de code exécuté. `$c` vaut déjà `Hello`, auquel on concatène `John` puis la variable `$b`, ce qui nous donne l'équivalent de `'Hello ' . 'John' . ' !' .`

`$c` vaut au final `Hello John !`

```
Hello
```

---

```
Hello World !
```

---

```
World !
```

---

```
VHello John !
```

Quel est le type de chacune des trois variables suivantes:

```
$a = 'Hello World !';
```

```
$b = 42;
```

```
$c = true;
```

### Explication:

a. Un ensemble de caractères délimités par des *simple quotes* `'` ou *double quotes* `"` est toujours une chaîne de caractère (*string*).

b. Un nombre entier est toujours de type **integer** (*entier* en anglais), mais un nombre à virgule flottante est de type **float**(*flottant* en anglais)

c. Une variable à `true` ou `false` sera toujours de type **boolean** (ou *booléen* en français).

A. text / B. number / C. truefalse

---

VA. string / B. integer / C. boolean

---

A. chars / B. float / C. scale

---

A. text / B. float / C. boolean

Comment définit-on une variable dont le contenu est une chaîne de caractères en PHP ?

### Explication:

La définition d'une variable se fait toujours par le nom de la variable `$myText`, suivi du signe `=` (ou d'un autre opérateur type concaténation) puis de la valeur à lui attribuer (ici `'Hello World !'`), puis un point-virgule `;` pour marquer la fin de l'instruction.

```
var myText = 'Hello World !';
```

---

```
var('myText', 'Hello World !');
```

---

---

```
V$myText = 'Hello World !';
```

---

```
myText = Hello World !;
```

Lequel de ces noms de variables est incorrect ?

**Explication:**

La variable `$42answer` est incorrecte car le nom d'une variable ne peut pas commencer par un chiffre. Elle peut en revanche commencer par une lettre ou un *underscore* `_`, et contenir des chiffres après le premier caractère qui suit le `$`.

```
$answer42
```

---

```
V$42answer
```

---

```
$answer_42
```

---

```
$_answer42
```

Quelle fonction permet d'afficher le contenu d'une variable ?

**Explication:**

`echo`, suivi d'une variable, affichera son contenu.

show

---

read

---

Vecho

---

display

---

À la fin du code, de quel type est la variable \$b ?

```
$a = 'Hello World!';
```

```
$b = true;
```

```
$a = 40;
```

```
$b = $a + 2;
```

### Explication:

Dans la 2ème partie du code, `$a` et `$b` sont redéfinis, `$a` vaut donc `40` et est un *integer*. On ajoute alors `2` à `$a` ce qui nous donne `42` et est également un *integer*.

*string*

---

*boolean*

---

*NULL*

---

---

*Vinteger*

Que signifient `if`, `elseif` et `else` ?

**Explication:**

Ces mots-clés permettent de définir un bloc de conditions, et désignent successivement **Si**, **Sinon Si** et **Sinon**. Un bloc de condition doit systématiquement démarrer par un `if`, puis d'un ou plusieurs `elseif` si nécessaire, et enfin d'un seul `else` lui aussi facultatif.

Oui / Peut-être / Non

---

Si Vrai / Si Faux / Sinon

---

VSi / Sinon Si / Sinon

---

Si Vrai / Si Faux / Si Null

Qu'est-ce qui entoure un bloc d'instructions de plusieurs lignes à exécuter si la condition est remplie ?

**Explication:**

Ce sont des accolades `{ }` qui doivent entourer le bloc d'instruction à exécuter si la condition précédant le bloc est remplie.

Les parenthèses servent à délimiter la condition, entre le mot-clé et le bloc d'instructions.

Des parenthèses ( )

---

Rien, il faut juste sauter une ligne après la condition

---

Des crochets [ ]

---

Des accolades { }

---

Est-il possible de vérifier plusieurs conditions en un seul `if` ?

### Explication:

Les opérateurs logiques (*et les parenthèses*) permettent de vérifier une infinité de conditions, et ce en un seul `if` ou `elseif` si c'est nécessaire. Il faut cependant faire attention à faire la condition la plus courte possible pour ne pas ralentir inutilement le code.

### Par exemple:

```
// À ÉVITER
if ($a == 1 || $a == 2 || $a == 3 || $a == 4) {
    // instructions
}
```

```
// RECOMMANDÉ
if ($a > 0 && $a < 5) {
```

```
// instructions  
}
```

Non, je dois faire plusieurs `if` imbriqués les uns dans les autres

---

Oui, via des opérateurs logiques

---

Oui, mais seulement si la condition porte sur la même variable

Comment indique-t-on à PHP la fin d'une instruction ?

### Explication:

Même si une instruction s'étend sur plusieurs lignes, c'est avec un `;` que l'on marque la fin d'une instruction, comme le point à la fin d'une phrase en français.

Par un saut de ligne

---

Avec un point-virgule `;`

---

Avec un chevron `>`

---

Avec une accolade `}`

Si je veux vérifier que la variable `$a` est bien égale à 42 (*quelque soit son type*), quelle syntaxe dois-je utiliser ?

### Explication:

Si un seul `=` est présent, il s'agit de l'opérateur d'affectation. Il permet donc d'assigner `42` à `$a`.

L'opérateur `<>` est équivalent à `!=` et signifie *différent de*, tandis que `===` permet bien de vérifier que `$a` vaut `42`, mais aussi qu'il soit du même type (ici, *integer*) et ce n'est pas ce que l'on veut ici.

C'est donc `==` qui doit être utilisé ici pour vérifier que `$a` vaut bien `42`, quelque soit le type de la valeur de `$a`.

```
V$a == 42
```

---

```
$a = 42
```

---

```
$a === 42
```

---

```
$a <> 42
```

Quel est l'opérateur parmi les choix suivants ?

### Explication:

- `<?php` est la balise d'ouverture permettant d'indiquer que ce qui suit est un bloc d'instructions à interpréter par PHP
- `if` est le mot-clé permettant de définir une condition à remplir pour exécuter des instructions PHP
- `echo` est une structure de langage (*ce qui la dispense de parenthèses*) permettant d'afficher une chaîne de caractères



- `===` est bien un opérateur, et de la catégorie **comparaison**. Il permet de vérifier que deux éléments sont égaux et de même type (*integer, string, etc.*).

```
<?php
```

```
if
```

```
v===
```

```
echo
```

Que va afficher le code suivant ?

```
$a = 42;  
  
if ($a == 0) {  
    echo 'Pomme';  
} elseif ($a < 16) {  
    echo 'Poire';  
} elseif ($a > 42) {  
    echo 'Abricot';  
} else {  
    echo 'Orange';  
}
```

**Explication:**

Le code affichera **Orange** car `$a` n'est pas égal à 0, n'est pas strictement inférieur à 16 ni strictement supérieur à 42.

Il rentrera donc au final dans le `else` qui est le *sinon* de tout le bloc de condition.

Pomme

---

Poire

---

Abricot

---

VOrange

---

Rien du tout

Qu'affiche le code suivant ?

```
$a = '42';  
$b = $a === 42;  
  
if ($a > 0 && !$b) {  
    echo 'Pomme';  
} elseif ($b && $a <= 42) {  
    echo 'Poire';  
} elseif (!$a || $b > 42) {  
    echo 'Abricot';  
}
```

**Explication:**

Au moment de la première condition (`if`),

- `$a` vaut `'42'` et est de type *string*
- `$b` vaut la valeur de retour de la condition "est-ce que `$a` vaut `42` et est de type *integer*"

La condition vérifie que `$a` est strictement supérieur à 0 peu importe son type (*donc 42 est bien supérieur à 0, même si c'est une string*), et que `$b` vaut `false`.

Étant donné que `$a` est bien égal à 42 mais qu'il n'est pas de type *integer*, `$b` est bien égal à `false`. La condition est donc remplie et le code affiche **Pomme**.

---

Si la première condition n'était pas remplie, la seconde vérifierait que `$a` est bien égal à 42 et de type *integer*, et également que `$a` est bien inférieur ou égal à `42`.

Enfin, la dernière condition vérifierait quand à elle que `$a` vaut `false` ou que `$b` est strictement supérieur à `42`.

VPomme

---

Poire

---

Abricot

---

Rien car aucune des conditions n'est remplie et il n'y a pas de `else`

Combien vaut `$d` à la fin du script ?

```
$a = 2;
```

```
$b = 3;
```

```
$c = 4;
```

```
$d = $a + $b * $c;
```

**Explication:**

Exactement comme en arithmétique, les multiplications sont prioritaires, on a donc `$b * $c` ( $3 \times 4 = 12$ ) auquel on ajoute `$a` ( $12 + 2 = 14$ ).

20

---

9

---

V14

---

24

Combien vaut `$c` à la fin de ce script ?

```
$a = 12;
```

```
$b = 8;
```

```
$c = 2;
```

```
$c += $a - $b;
```

**Explication:**

La différence entre `$a` et `$b` est d'abord calculée via l'opérateur d'arithmétique `-` ( $12 - 8 = 4$ ) puis le résultat ajouté à la valeur existante de `$c` via l'opérateur arithmétique d'affectation `+=`, soit  $2 + 4 = 6$ .

Cela est donc équivalent à `$c = $c + ($a - $b)` et donc `$c = 2 + (12 - 8)`.

V6

---

20

---

22

---

42

Quelle réponse permet de faire en sorte que `$d` soit égal à 42 avec les variables suivantes:

`$a = 5;`

`$b = 7;`

`$c = 8;`

`$d = 0;`

**Explication:**

Correction de la question

```
V$d += $a * $b + --$c;
```

---

---

```
$d = ($a + $b) * $c;
```

---

```
$d = $a + $b + $c;
```

---

```
$d = $c-- + ($a * $b);
```

Quelle réponse permet d'avoir le code suivant en une seule ligne, si `$a = 0` ?

```
$a = $a + 1;
```

```
$b = $a;
```

### Explication:

C'est l'opérateur d'incrément `++` qui doit être utilisé ici. Et étant donné que `$b` doit avoir la valeur mise à jour de `$a`, il doit être utilisé avant la variable et non après, auquel cas `$b` aurait été égal à 0, et ensuite `$a` aurait value 1;

```
$b = $a += 1;
```

---

```
V$b = ++$a;
```

---

```
$b = $a ++ 1;
```

---

```
$b = $a++;
```

Que vaut `$b` à la fin du script ?

```
$a = 2;  
$b = $a-- + 1;
```

### Explication:

Le `--` se trouvant après `$a`, la valeur retournée n'est pas encore modifiée.

Cela équivaut donc à: `$b = 2 + 1;`.

À la fin du script, `$a` quand à lui vaut `1`;

0

---

1

---

2

---

V3

Quel niveau d'erreur de cette liste n'existe pas en PHP ?

### Explication:

Les 3 niveaux d'erreurs en PHP sont `NOTICE`, `WARNING` et `ERROR`, de l'information à l'erreur critique.

L'affichage du niveau d'erreur est réglable dans la configuration de PHP.

NOTICE

---

WARNING

---

ERROR

---

VMESSAGE

Que signifie l'erreur suivante ?

Parse error: syntax error, unexpected 'echo' (T\_ECHO), expecting ',' or ';' in /www/script.php on line 3

**Explication:**

En effet, en l'absence du point-virgule, l'instruction continue d'être exécutée.

La fonction `echo` n'existe pas

---

VII manque un `;` à la ligne précédente

---

La fonction `echo` a été mal écrite

---

Il y a un `,` ou `;` de trop à la ligne 3



Quelle fonction permet de stopper l'exécution du script et éventuellement d'afficher un message ?

**Explication:**

C'est la fonction `die (...)` (*alias de* `exit (...)`) qui permet de stopper l'exécution du script à l'endroit où elle est placée en affichant un message si celui-ci est passé en argument.

```
stop('Message à afficher');
```

---

```
Vdie('Message à afficher');
```

---

```
debug('Message à afficher');
```

---

```
dump('Message à afficher');
```

Quel est la bonne requete HTTP pour récupérer l'article d'un blog ?

**Explication:**

Il faut bien utiliser le verbe **GET** dans la requête pour dire que nous voulons récupérer la ressource et non créer, mettre à jour ou supprimer la ressource.

```
PUT /blog/1 HTTP/1.1
```

---

```
VGET /blog/1 HTTP/1.1
```

---

---

PUT /blog/1 HTTP/1.1

---

DELETE /blog/1 HTTP/1.1

Quelle réponse le serveur doit-il retourner pour indiquer que la ressource demandée n'existe pas ?

**Explication:**

Le serveur doit retourner le code de statut **404** de la réponse pour indiquer au client que la ressource demandée n'existe pas. Pour cela, il suffit de consulter la [liste des code de status](#) pour retourner le bon code.

HTTP/1.1 200 OK Date: Sun, 13 Sep2015 14:05:05 GMT Server: lighttpd/1.4.19  
Content-Type: text/html

---

HTTP/1.1 201 Created Date: Sun, 13 Sep2015 14:05:05 GMT Server: lighttpd/1.4.19  
Content-Type: text/html

---

VHTTP/1.1 404 Not Found Date: Sun, 13 Sep2015 14:05:05 GMT Server: lighttpd/1.4.19  
Content-Type: text/html

---

HTTP/1.1 500 Internal Server Error Date: Sun, 13 Sep2015 14:05:05 GMT Server:  
lighttpd/1.4.19 Content-Type: text/html

Quelle est la fonctionnalité d'un serveur HTTP Apache ?

**Explication:**

Un serveur Apache est un serveur HTTP pouvant interpréter du PHP afin de répondre à une requête client en respectant le protocole de communication client-serveur Hypertext Transfer Protocol (HTTP), qui a été développé pour le World Wide Web. Pour plus d'informations, consultez cet article sur [Apache HTTP Server](#).

Stocker les informations dynamiques d'un site internet.

---

Compiler et exécuter le code PHP

---

VRépondre à une requête HTTP d'un client.

---

Envoyer une requête HTTP à un autre serveur.

Parmi les propositions ci-dessous, laquelle est utilisée pour stocker des informations en vue de rendre un site dynamique ?

**Explication:**

Les serveurs web PHP fonctionnent le plus souvent conjointement à MySQL (système de gestion de base de données) qui permet de stocker les informations dynamiques de votre site : le contenu des pages, les commentaires, les utilisateurs, la configuration, etc. Les données sont stockées sous la forme de plusieurs tableaux de données appelés des tables.

VMySQL

---

---

La dernière version de mon navigateur web

---

PHP

---

Serveur HTTP