

# PHP

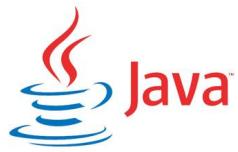
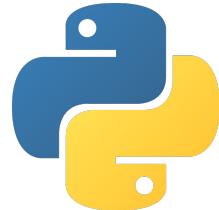
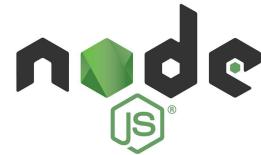
“

PHP (Hypertext Preprocessor), est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

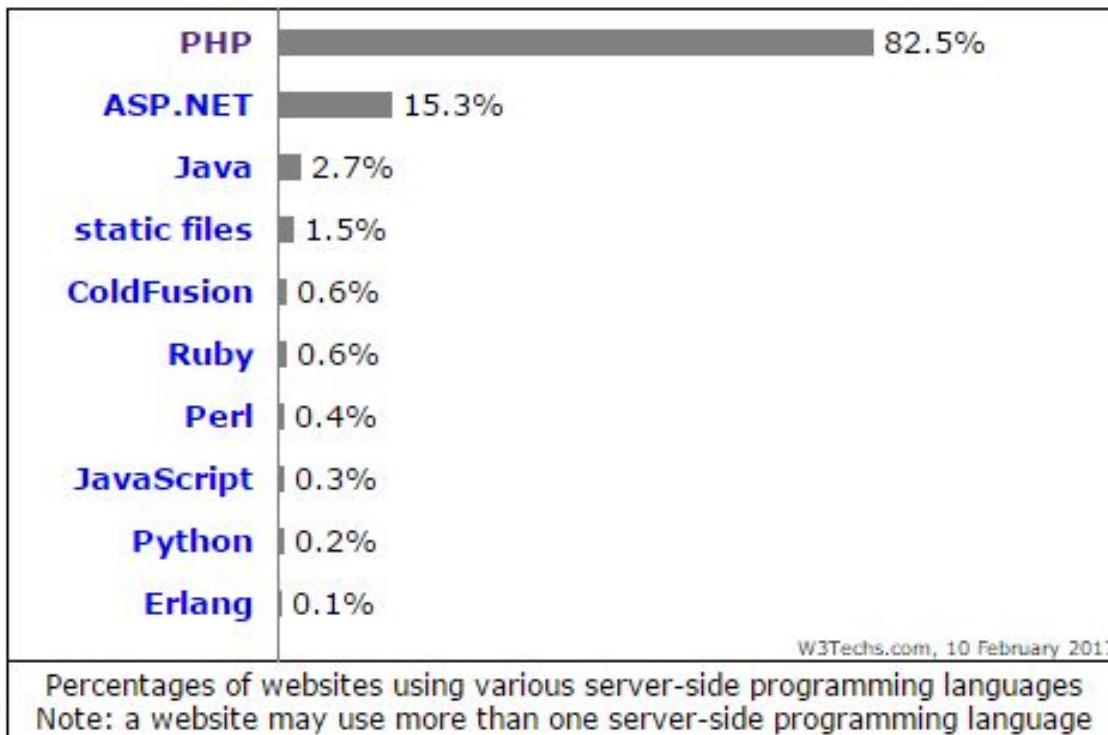
Il a un typage dynamique, faible

Source : [https://fr.wikipedia.org/wiki/Langage\\_serveur](https://fr.wikipedia.org/wiki/Langage_serveur)

# Les codes serveur



# Les codes serveur



# Les bases de données

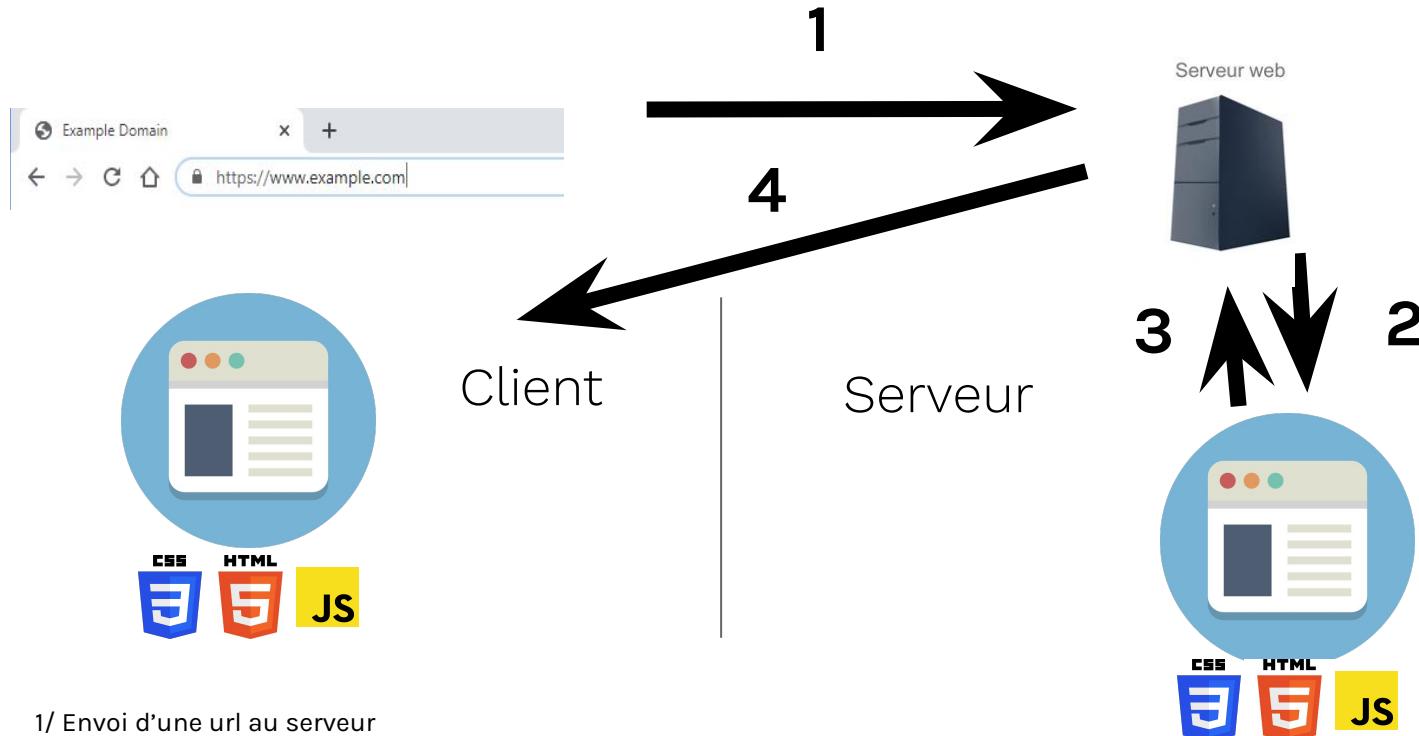
ORACLE®



PostgreSQL



# Comment fonctionne un site internet statique ?



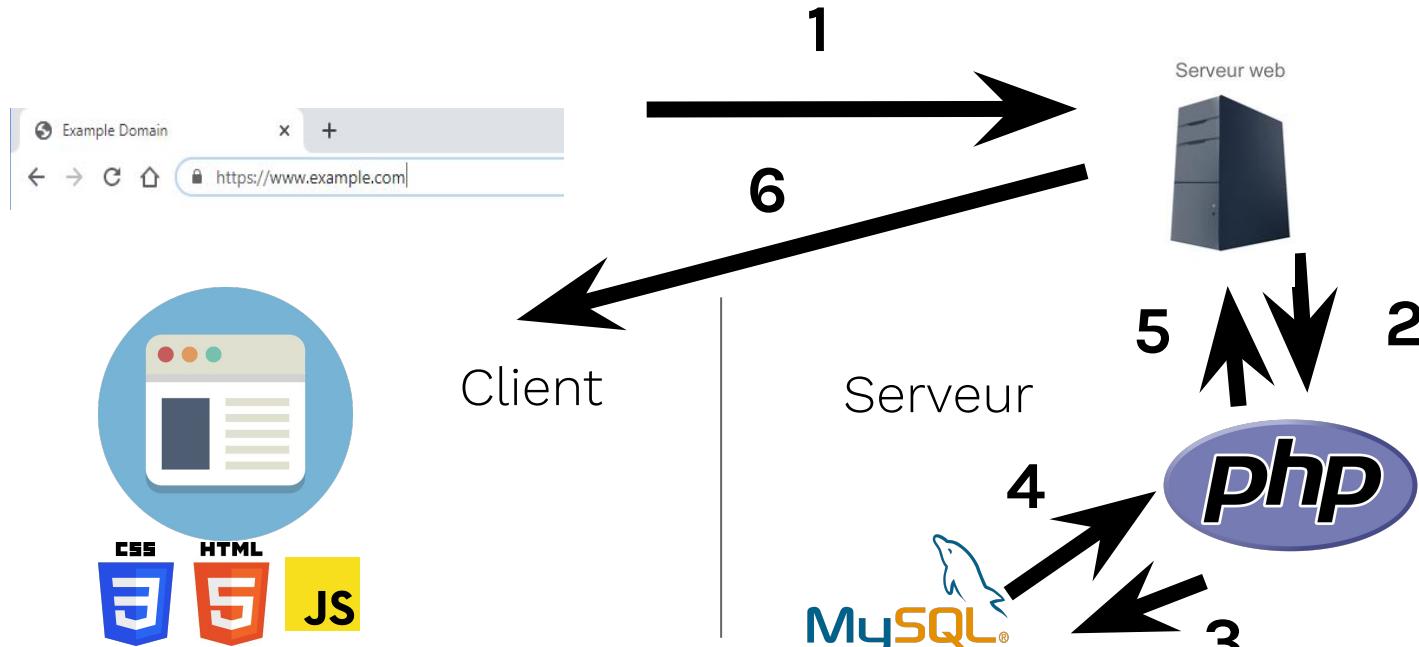
1/ Envoi d'une url au serveur

2/Récupération de la page web

3/ Une fois la page web récupérée, par le serveur

4/ Elle est envoyée vers le client HTTP (navigateur) pour être visible

# Comment fonctionne un site internet dynamique ?



1/ Envoi d'une url au serveur

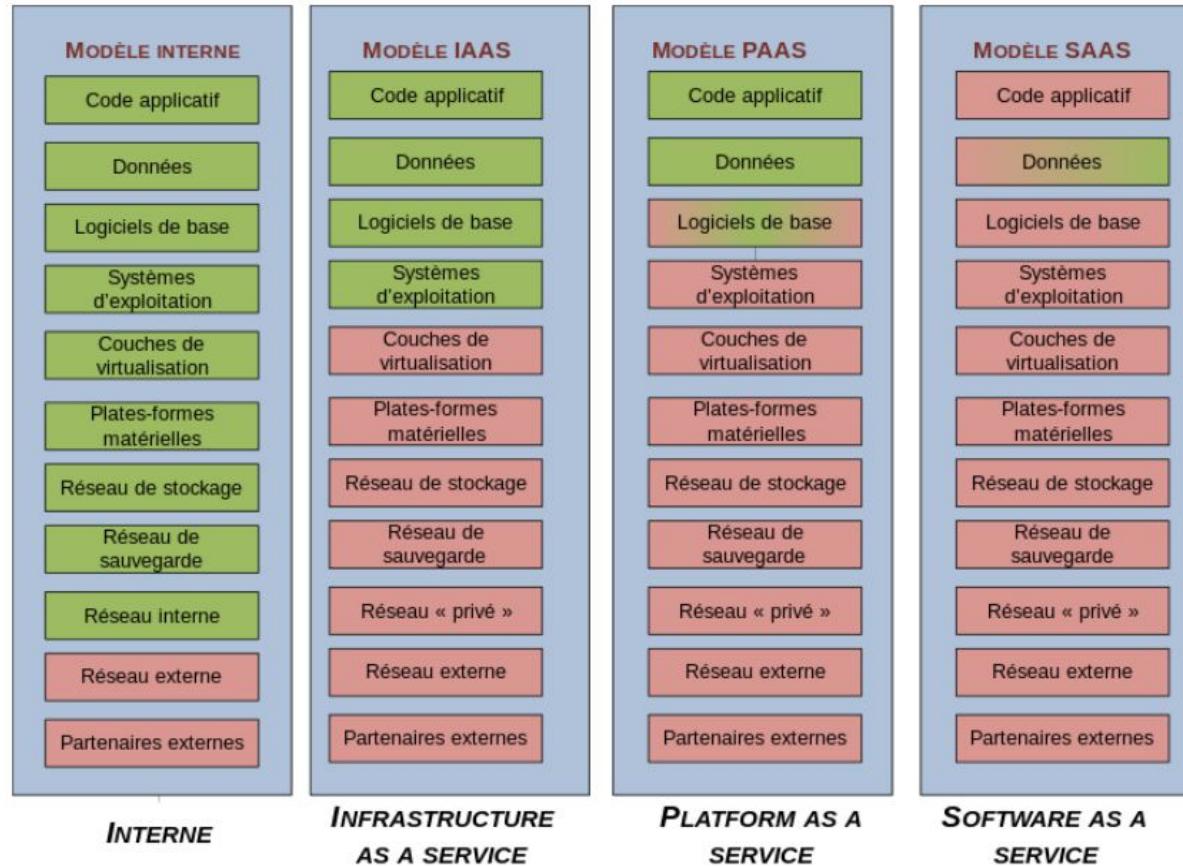
2/ Le code analyse l'url pour savoir à quoi elle correspond (controllers)

3/ On récupère les données dans la base (models)

4/ Une fois les données récupérées, elles sont renvoyées au "controllers" pour être traitées insérées dans une page (views)

5/ Une fois la page web construite, par le serveur

6/ Elle est envoyée vers le client HTTP (navigateur) pour être visible



**INFRASTRUCTURE  
AS A SERVICE**

**PLATFORM AS A  
SERVICE**

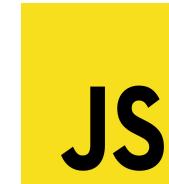
**SOFTWARE AS A  
SERVICE**



Sous la responsabilité  
de l'entreprise



Sous la responsabilité  
du fournisseur



## Quelques définitions :

String : chaîne de caractère (une phrase en gros, entre guillemet)

Int : un chiffre/nombre entier

Float : un chiffre/nombre décimal

Boolean : true /false - 0 / 1

Array : c'est un tableau (indiqué, associatif, multi-dimensionnel)

Foreach : le fait parcourir un array / objet

Concaténation : le fait de mettre une variable dans un string

Function : une fonction

Class : stocke des méthodes (fonctions) et des propriétés  
(variables)

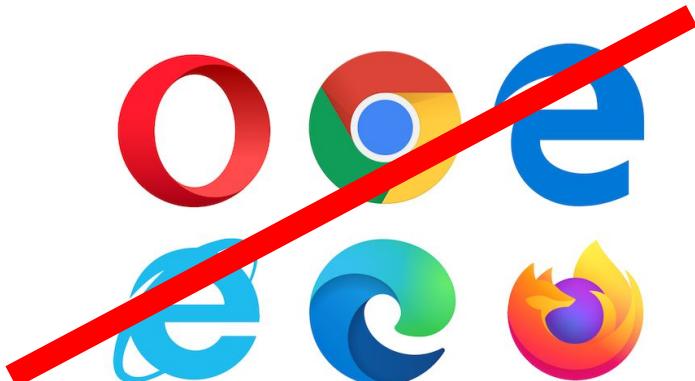


A screenshot of a web browser window. The address bar shows "Fichier | C:/Users/Nicolas/Desktop/index.php". The page content displays the following PHP code:

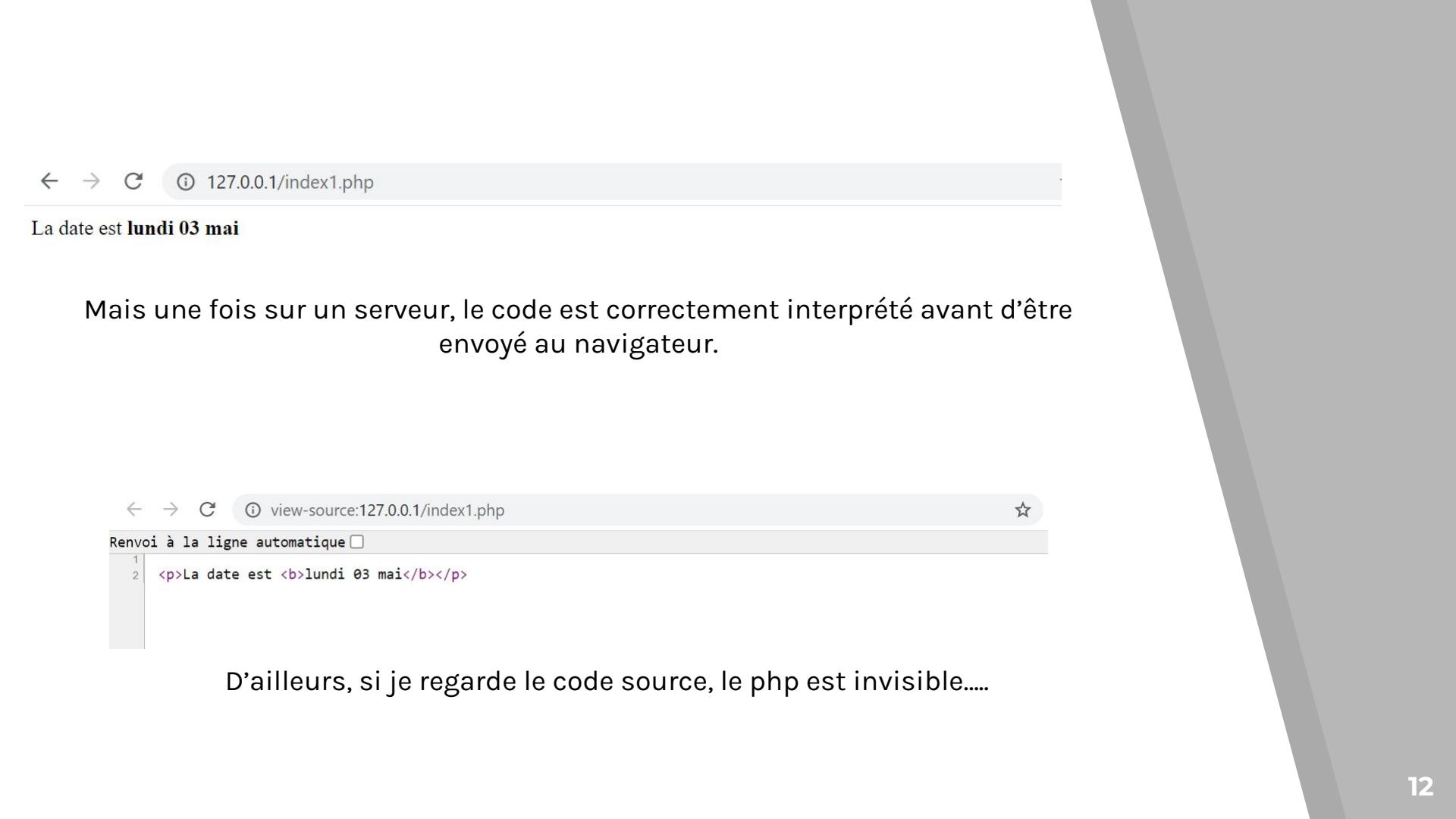
```
<?php

function getFrenchy($date){
    return str_replace(["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday", "January", "February", "March",
"April", "May", "June", "July", "August", "September", "October", "November", "December"],
["lundi", "mardi", "mercredi", "jeudi", "vendredi", "samedi", "dimanche", "janvier", "février", "mars", "avril", "mai", "juin", "juillet", "aout", "septembre", "octobre", "novembre", "décembre"]
,$date);
}

?>
<p>La date est <b><?php echo getFrenchy(date('l d F'));></b></p>
```



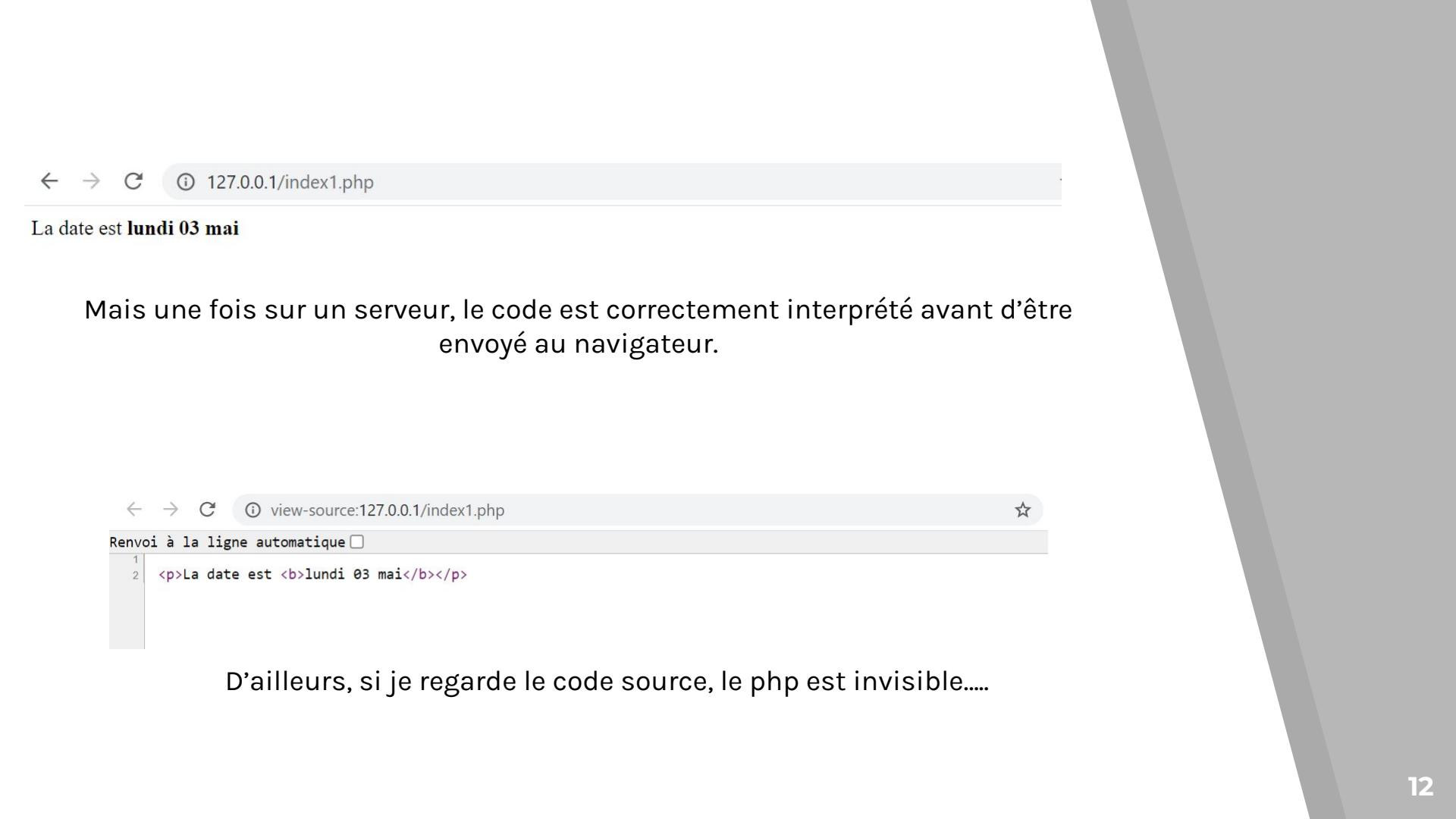
Les navigateurs web sont incapables d'interpréter du code serveur.



← → ⌂ ⓘ 127.0.0.1/index1.php

La date est **lundi 03 mai**

Mais une fois sur un serveur, le code est correctement interprété avant d'être envoyé au navigateur.



← → ⌂ ⓘ view-source:127.0.0.1/index1.php ☆

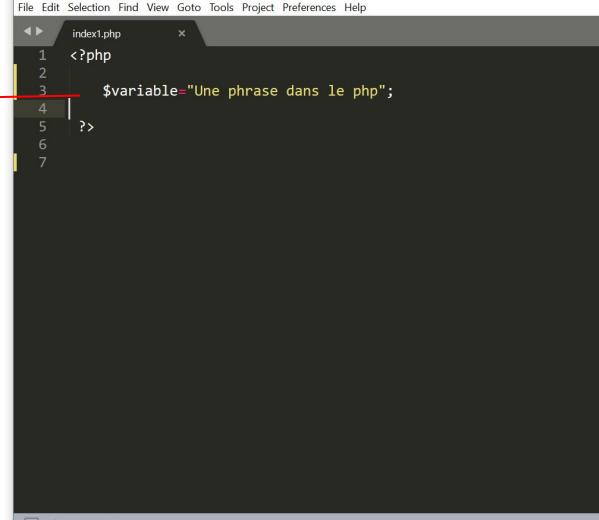
Renvoi à la ligne automatique □

```
1 <p>La date est <b>lundi 03 mai</b></p>
2
```

D'ailleurs, si je regarde le code source, le php est invisible.....

# **LES VARIABLES**

??



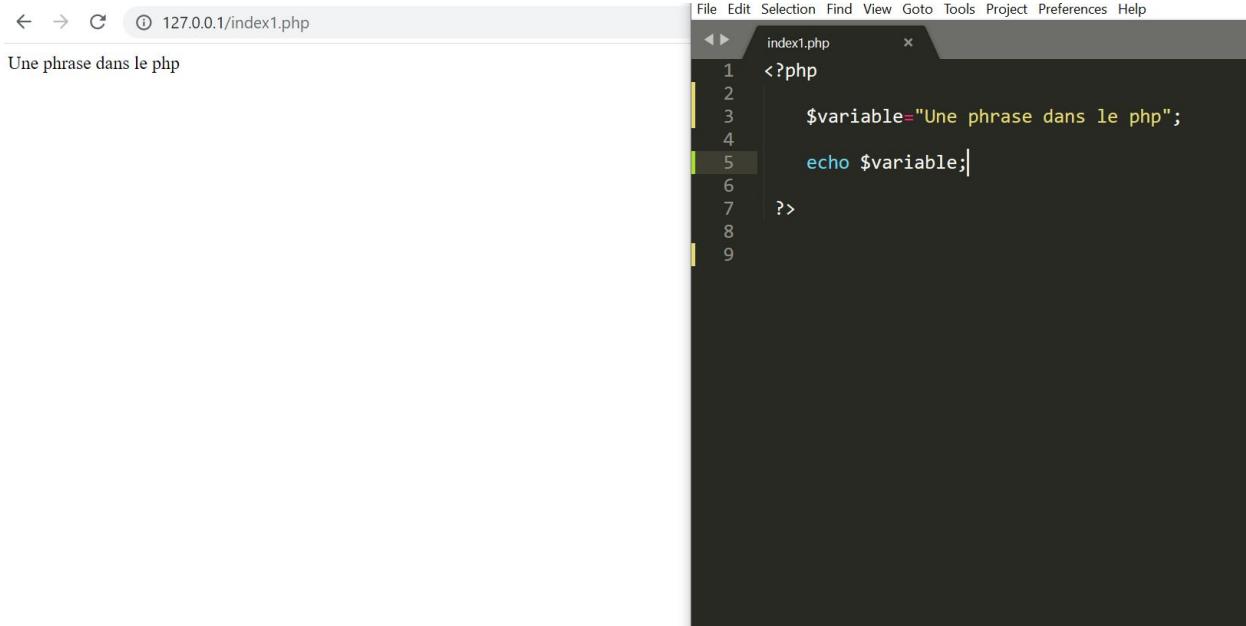
A screenshot of a code editor window titled "index1.php". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The code area contains the following PHP script:

```
<?php  
$variable="Une phrase dans le php";  
?>
```

En php, on peut créer des variables. Les variables sont égales à ce que l'on souhaite.

Une variable doit commencer par \$, les espaces sont interdits, les tirets sont interdits, il ne peut pas y avoir de chiffre directement après le \$. Une commande php se termine par ";"

Dans l'exemple ci-dessus, la variable "\$variable" (j'ai peu d'imagination) vaut "Une phrase dans le php". Mais je ne la vois pas côté client.

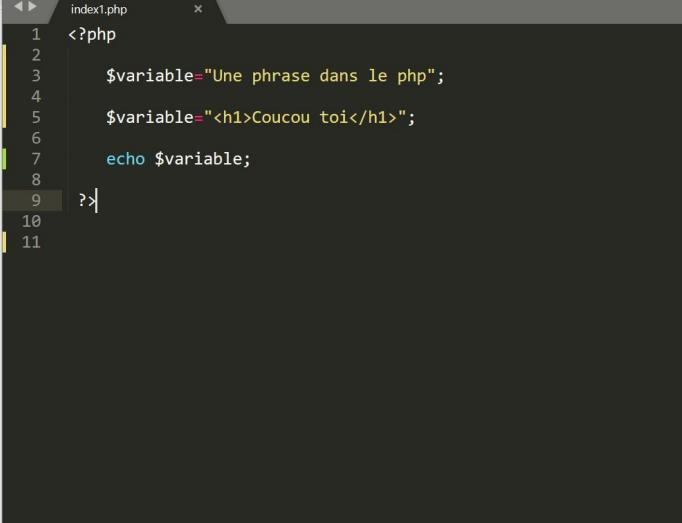


A screenshot of a web browser window showing the URL 127.0.0.1/index1.php. Below the browser, a code editor window titled 'index1.php' displays the following PHP code:

```
<?php  
$variable="Une phrase dans le php";  
echo $variable;  
?>
```

Pour là faire apparaître, je dois mettre “echo” avant....

Coucou toi



The screenshot shows a web browser window with the URL 127.0.0.1/index1.php. The page content is "Coucou toi". Above the browser, a code editor window titled "index1.php" is visible, displaying the following PHP code:

```
<?php  
$variable="Une phrase dans le php";  
$variable=<h1>Coucou toi</h1>;  
echo $variable;  
?>
```

The code editor has line numbers from 1 to 11 on the left. The line containing the variable assignment is highlighted.

La variable peut être réécrite à l'infini. Le code étant lu de haut en bas, la dernière valeur de la variable "\$variable" est "<h1>Coucou toi</h1>" (du html).

```
//String  
$str = 'A string';  
$str2 = "Another string";  
  
//Float  
$flt = 1.0;  
$flt2 = -2.1;  
  
$str3 = '1';  
$str4 = '2.0'; //Integer  
$str5 = "-2";  
$str6 = 'true';  
$str7 = "false";
```

```
//Null  
$varNull = null;  
$VARnull;  
  
//Boolean  
$bol = true;  
$BOL = TRUE;  
$bol2 = false;  
$bol3 = FALSE;
```

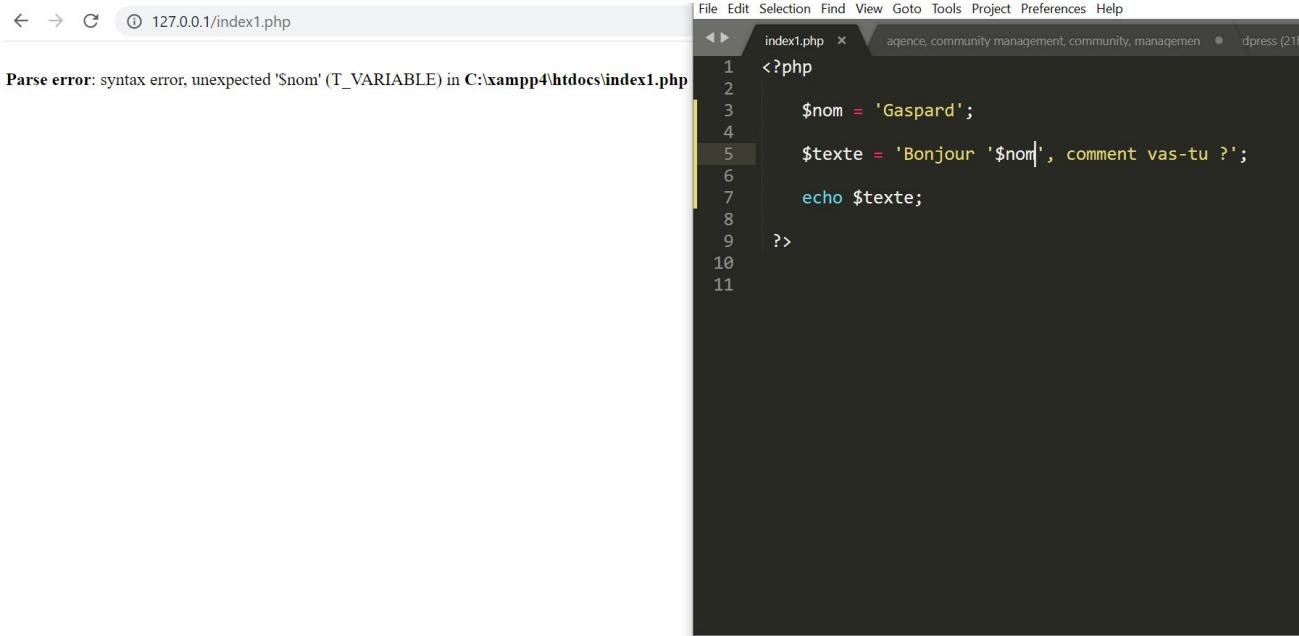
The screenshot shows a web browser window at the top with the URL 127.0.0.1/index1.php. The page content is "Bonjour Gaspard, comment vas-tu ?". Below it is a code editor window titled "index1.php" showing the following PHP code:

```
1 <?php
2
3     $nom = 'Gaspard';
4
5     $texte = 'Bonjour ' . $nom . ', comment vas-tu ?';
6
7     echo $texte;
8
9
10    ?>
11
```

Voici un exemple de concaténation. On peut voir 2 strings :

‘Bonjour’ et ‘, comment vas-tu ?’

Pour que cela fonctionne, il faut mettre un “.” entre la variable (\$nom).



The screenshot shows a web browser window with the URL 127.0.0.1/index1.php. The page displays the following error message:

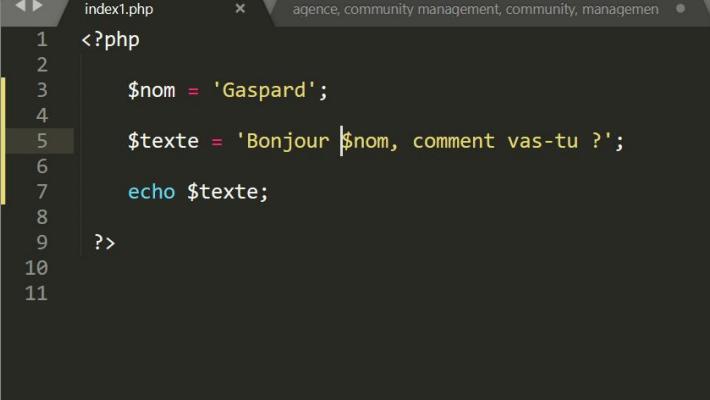
Parse error: syntax error, unexpected '\$nom' (T\_VARIABLE) in C:\xampp\htdocs\index1.php

Below the error message is a code editor window showing the PHP script index1.php:

```
index.php x agence, community management, community, managemen ● dpress (21h)
1 <?php
2
3     $nom = 'Gaspard';
4
5     $texte = 'Bonjour '$nom', comment vas-tu ?';
6
7     echo $texte;
8
9 ?>
10
11
```

Si j'enlève les “.” de chaque côté de ma variable, j'ai une superbe erreur !

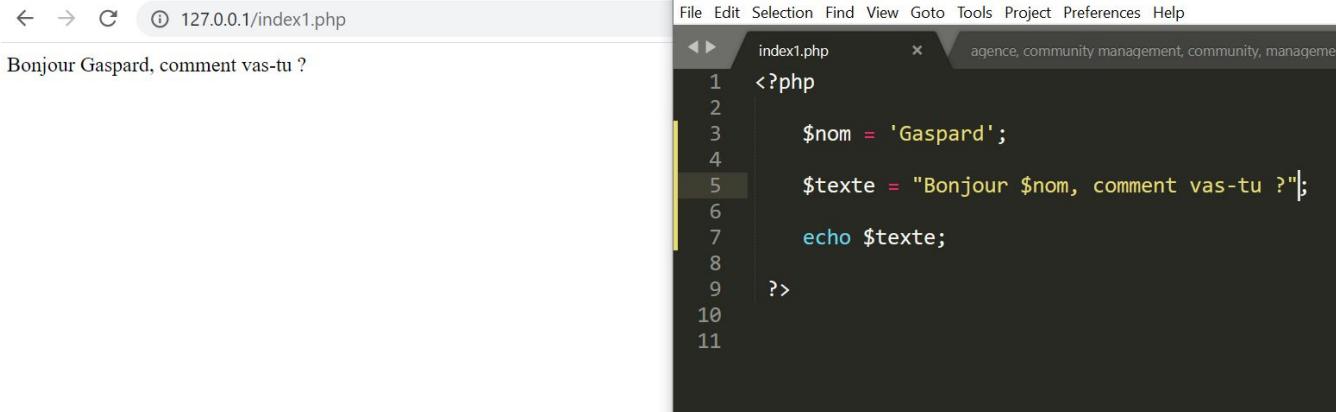
Bonjour \$nom, comment vas-tu ?



```
index1.php      agence, community management, community, managemen
1  <?php
2
3      $nom = 'Gaspard';
4
5      $texte = 'Bonjour $nom, comment vas-tu ?';
6
7      echo $texte;
8
9  ?>
10
11
```

Dans l'exemple ci-dessus, j'utilise des guillemets simples. Nous allons voir la différence entre les guillemets simples et les doubles.

Si je mets une variable (sans concaténation) entre des guillemets simples, elle ne sera pas exécutée en tant que variable mais en tant que string.



The screenshot shows a web browser window with the URL 127.0.0.1/index1.php. The page displays the text "Bonjour Gaspard, comment vas-tu ?". Above the browser, a code editor window is open, showing the PHP script index1.php. The script contains the following code:

```
index1.php
1 <?php
2
3     $nom = 'Gaspard';
4
5     $texte = "Bonjour $nom, comment vas-tu ?";
6
7     echo $texte;
8
9     ?>
10
11
```

Cependant, la particularité des guillemets doubles est d'exécuter les variables.  
C'est une autre façon de concaténer.

```
Addition : 3  
Soustraction : 1  
Division : 2  
Mutiplication : 2  
Modulo : 0  
Mix : 9  
Mix 2 : 0.66666666666667
```

```
Warning: A non-numeric value encountered in C:\xampp4\htdocs\index1.php on line 14  
2  
Erreur : 1 + 2
```

The screenshot shows a code editor with a dark theme. On the left, there is a vertical list of numbers from 1 to 19. To the right of each number is a line of PHP code. Lines 1 through 18 are standard echo statements outputting mathematical operations. Line 19 contains a closing PHP tag. Lines 14 and 15 are highlighted in yellow, indicating they are the source of the warning message shown in the terminal window below.

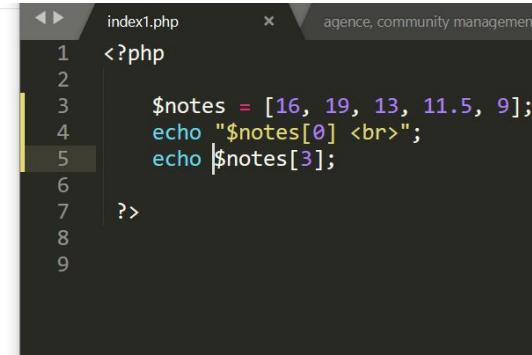
```
<?php  
1 $num = 1;  
2 $num2 = 2;  
3  
4 echo 'Addition : ' .($num + $num2). '<br>';  
5 echo 'Soustraction : ' .($num2 - $num). '<br>';  
6 echo 'Division : ' .($num2 / $num). '<br>';  
7 echo 'Mutiplication : ' .($num2 * $num). '<br>';  
8 echo 'Modulo : ' .($num2 % $num). '<br>';  
9 echo 'Mix : ' .((($num2 + $num) * 3)). '<br>';  
10 echo 'Mix 2 : ' .($num2 * $num) .| 3 . '<br>';  
11 echo "<hr>";  
12 echo 'Erreur : ' .($num + $num2). '<br>';  
13 echo "Erreur : $num + $num2 <br>";  
14  
15  
16 ?>  
17  
18  
19
```

PHP peut faire toute sorte de calcul...

La petite notion à connaître est le “modulo”. Le modulo récupère le reste d'une division.

# **Les Tableaux**

16  
11.5



```
index1.php agence, community management
1 <?php
2
3     $notes = [16, 19, 13, 11.5, 9];
4     echo "$notes[0] <br>";
5     echo $notes[3];
6
7     ?>
8
9
```

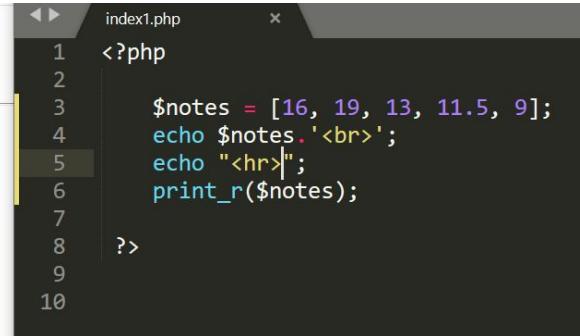
La variable “\$notes” est un tableau indexé.  
“16” est en première position, ‘19’ en deuxième, etc...

Détail important, php commence à compter à partir de 0. Donc le premier élément à l’index 0 et pas 1.

\$notes[0] // On demande la valeur 1 du tableau “\$notes” -> 16  
\$notes[3] // On demande la valeur 4 du tableau “\$notes” -> 11.5

Notice: Array to string conversion in C:\xampp4\htdocs\index1.php on line 4  
Array

Array ( [0] => 16 [1] => 19 [2] => 13 [3] => 11.5 [4] => 9 )



```
index1.php
1 <?php
2
3     $notes = [16, 19, 13, 11.5, 9];
4     echo $notes.'<br>';
5     echo "<hr>";
6     print_r($notes);
7
8 ?>
9
10
```

Petite parenthèse, vous ne pouvez pas afficher un tableau dans son ensemble avec “echo”.

Il faut utiliser “print\_r()”.



Array ( [0] => 16 [1] => 19 [2] => 13 [3] => 11.5 [4] => 9 [5] => coucou [6] => Array ( [0] => 16 [1] => hey ) )

C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
index1.php
1 <?php
2
3     $notes = [16, 19, 13, 11.5, 9, 'coucou',[16, 'hey']];
4
5     print_r($notes);
6
7 ?>
8
9
```

Un tableau peut contenir des nombres, mais aussi du texte et d'autres tableaux..  
Petite question, je fais comment pour afficher "hey" ?.

The screenshot shows a Sublime Text window with the following details:

- File Path: C:\xampp4\htdocs\index1.php
- Editor Title: index1.php
- Content:

```
1 <?php
2
3     $notes = [16, 19, 13, 11.5, 9, 'coucou',[16, 'hey']];
4
5     print_r($notes[6][1]);
6
7 ?>
8
9
```

```
print_r($notes[6][1]);
```

← → C ① 127.0.0.1/index1.php

dupont Sophie



The screenshot shows a Sublime Text window with the file 'index1.php' open. The code is as follows:

```
<?php  
$eleves = ['nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,30]];  
echo $eleves['nom']. ' ' . $eleves['prenom'];  
?>
```

The line 'echo \$eleves['nom']. ' ' . \$eleves['prenom'];' is highlighted.

“\$eleves” est un tableau associatif (avec des colonnes en fait). Au lieu de mettre des indexs entre crochets, je mets directement le nom de la colonne.

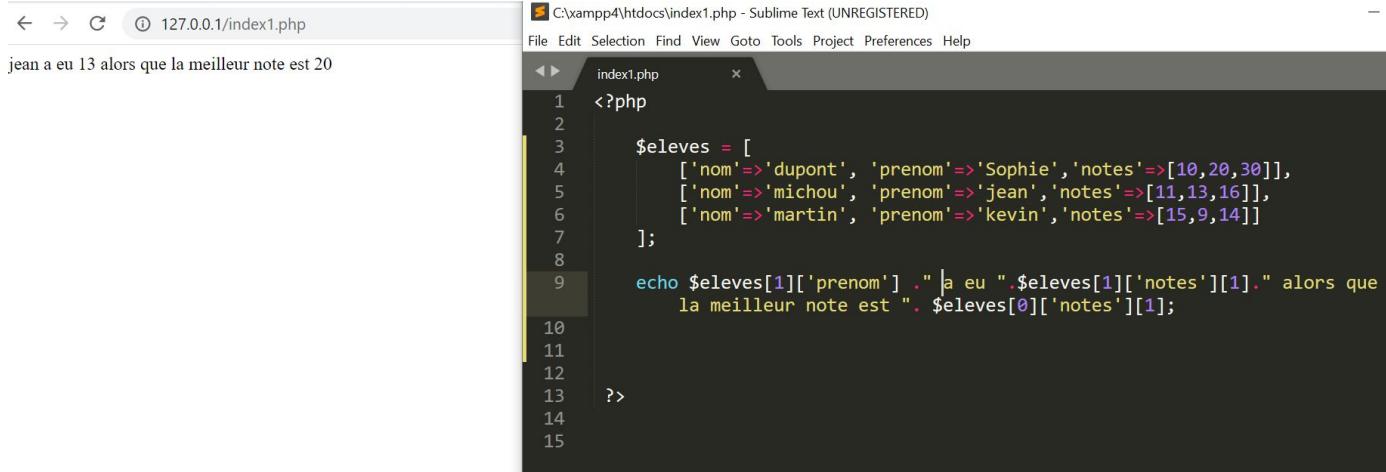
The screenshot shows a browser window at 127.0.0.1/index1.php displaying the output "martin Sophie" followed by an array dump: "Array ( [0] => 10 [1] => 20 [2] => 30 [3] => 12 )". Below it is a Sublime Text editor window titled "index1.php" containing the following PHP code:

```
<?php
$eleves = ['nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,30]];
$eleves['nom'] = 'martin';
$eleves['notes'][3] = 12;
echo $eleves['nom']. ' ' . $eleves['prenom']. '<br>';
print_r($eleves['notes']);
?>
```

Il est possible de modifier à la volé, les valeurs du tableau.

“\$eleves['nom'] = 'martin';” a remplacé la valeur d’origine qui était “dupond”

“\$eleves['notes'][3] = 12;” a ajouté la note de 12 en index 4 de “notes”



The screenshot shows a Sublime Text window with the file 'index1.php' open. The code defines an associative array '\$eleves' containing four students with their names, first names, and notes. It then prints a message indicating that Jean has the best note, which is 20.

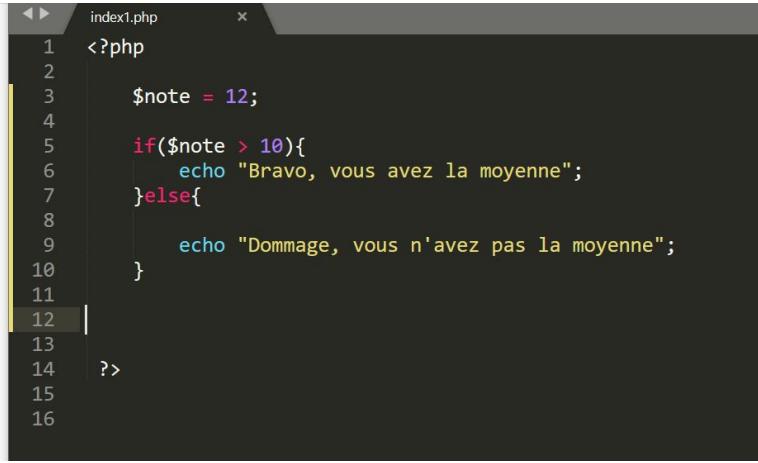
```
<?php
$eleves = [
    ['nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,30]],
    ['nom'=>'michou', 'prenom'=>'jean', 'notes'=>[11,13,16]],
    ['nom'=>'martin', 'prenom'=>'kevin', 'notes'=>[15,9,14]]
];
echo $eleves[1]['prenom'] ." a eu ". $eleves[1]['notes'][1]. " alors que
la meilleure note est ". $eleves[0]['notes'][1];
?>
```

Un tableau peut-être associatif avec index si il a plusieurs lignes....

# A vous de jouer !

# **Les Conditions**

Bravo, vous avez la moyenne



A screenshot of a code editor window titled "index1.php". The code is written in PHP and contains an if-else conditional statement. The output of the code is "Bravo, vous avez la moyenne".

```
<?php  
$note = 12;  
if($note > 10){  
    echo "Bravo, vous avez la moyenne";  
}else{  
    echo "Dommage, vous n'avez pas la moyenne";  
}  
?>
```

Les conditions permettent d'exécuter un code spécifique en fonction de ce qui est demandé.

```
if(condition “Note est strictement supérieur à 10”){  
    //si la condition est correcte  
    J'affiche "Bravo, vous avez la moyenne"  
}else{  
    //Si elle ne l'est pas  
    J'affiche "Dommage, vous n'avez pas la moyenne  
}
```

Bravo, vous avez la moyenne

```
index1.php
1 <?php
2
3     $note = 12;
4
5     if($note > 10){
6         echo "Bravo, vous avez la moyenne";
7     }else{
8
9         echo "Dommage, vous n'avez pas la moyenne";
10    }
11
12 |
13
14 ?>
15
16
```

Les conditions permettent d'exécuter un code spécifique en fonction de ce qui est demandé.

> strictement supérieur

< strictement inférieur

<= inférieur ou égal

>= supérieur ou égal

== est égal

!= n'est pas égal ("!" signifie "pas")

Vous avez juste la moyenne

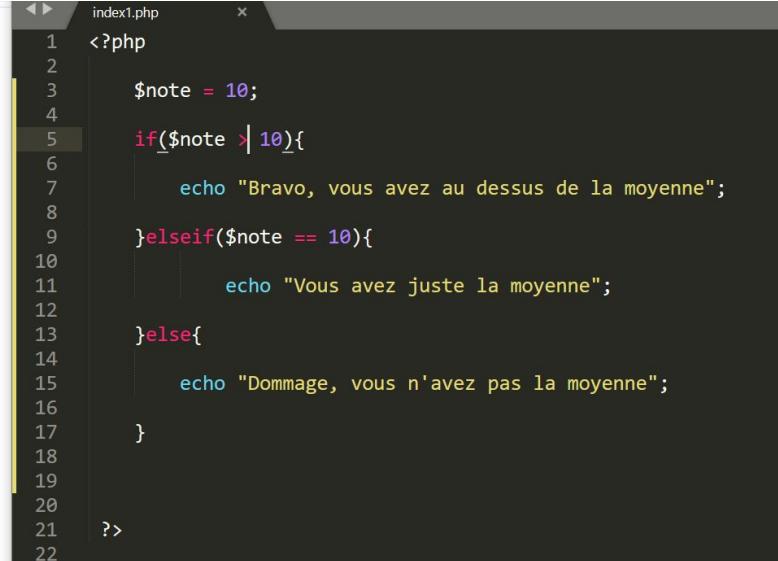
```
index1.php
1 <?php
2
3     $note = 10;
4
5     if($note >= 10){
6
7         if($note == 10){
8             echo "Vous avez juste la moyenne";
9         }else{
10            echo "Bravo, vous avez au dessus de la moyenne";
11        }
12    }else{
13
14        echo "Dommage, vous n'avez pas la moyenne";
15    }
16
17
18
19
20 ?>
21
22
```

Les conditions peuvent s'imbriquer.

Dans l'exemple ci-dessus, on voit qu'il y a une condition spécifique si l'utilisateur a pile 10.

Cependant, la lecture de ce code est un peu difficile.

Vous avez juste la moyenne



The screenshot shows a code editor window titled "index1.php". The code is a PHP script that checks a variable \$note. If \$note is greater than 10, it outputs "Bravo, vous avez au dessus de la moyenne". If \$note is exactly 10, it outputs "Vous avez juste la moyenne". If \$note is less than 10, it outputs "Dommage, vous n'avez pas la moyenne". The code uses standard PHP syntax with if, elseif, and else statements.

```
<?php
$note = 10;
if($note > 10){
    echo "Bravo, vous avez au dessus de la moyenne";
}elseif($note == 10){
    echo "Vous avez juste la moyenne";
}else{
    echo "Dommage, vous n'avez pas la moyenne";
}
?>
```

Il est possible d'enchaîner les conditions avec “elseif” ce qui rend la lecture du code plus simple (et donc le débbugage)

```
<?php  
  
$note = 10;  
  
if($note > 10){  
  
    echo "Bravo, vous avez au dessus de la moyenne";  
  
}elseif($note === 10){  
  
    echo "Vous avez juste la moyenne";  
  
}else{  
  
    echo "Dommage, vous n'avez pas la moyenne";  
  
}  
  
?>
```

En php, il existe l'opérateur “==”  
En fait, php est capable de faire de l'interpolation. Le “==” est plus strict et empêche toute ambiguïté.

Vous avez juste la moyenne



```
index1.php
1 <?php
2
3     $note = "10";
4
5     if($note > 10){
6
7         echo "Bravo, vous avez au dessus de la moyenne";
8
9     }elseif($note == 10){
10
11         echo "Vous avez juste la moyenne";
12
13     }else{
14
15         echo "Dommage, vous n'avez pas la moyenne";
16
17     }
18
19
20
21 ?>
22
23
```

Si je change mon 10 en “10”, je change de type, donc je passe d’un type “int” en “string”. Techniquement, il est impossible de dire “si la valeur de tel mot est supérieur à l’autre...”. Php comprend qu’il s’agit de nombre puisque que j’ai utilisé le symbole “>”, il va donc, de lui même, changer ma phrase en nombre.

Vous avez juste la moyenne

The screenshot shows a code editor window with a dark theme. The file is named 'index1.php'. The code is as follows:

```
index1.php
1 <?php
2
3     $note = "10dfjdkfjdfjd";
4
5     if($note > 10){
6
7         echo "Bravo, vous avez au dessus de la moyenne";
8
9     }elseif($note == 10){
10
11         echo "Vous avez juste la moyenne";
12
13     }else{
14
15         echo "Dommage, vous n'avez pas la moyenne";
16
17     }
18
19
20
21 ?>
22
23
```

The line 'echo "Vous avez juste la moyenne";' is highlighted in yellow.

De même si mon “string” contient des lettres. Puisque qu’il commence par 10, php va se dire “ok, je détecte le nombre 10 dans ce string, donc je pense qu’il a voulu dire 10”

The screenshot shows a web browser window at the URL 127.0.0.1/index1.php. The page displays the text "Dommage, vous n'avez pas la moyenne". Below it is an IDE interface with a dark theme, showing the source code of index.php. The code contains a PHP script that checks a variable \$note against 10. If \$note is greater than 10, it outputs "Bravo, vous avez au dessus de la moyenne". If \$note is equal to 10, it outputs "Vous avez juste la moyenne". If \$note is less than 10, it outputs "Dommage, vous n'avez pas la moyenne". The code also includes a comment with a long string of characters.

```
<?php  
$note = "klklklk10dfjdfkjdfjd";  
if($note > 10){  
    echo "Bravo, vous avez au dessus de la moyenne";  
}elseif($note == 10){  
    echo "Vous avez juste la moyenne";  
}else{  
    echo "Dommage, vous n'avez pas la moyenne";  
}  
?>
```

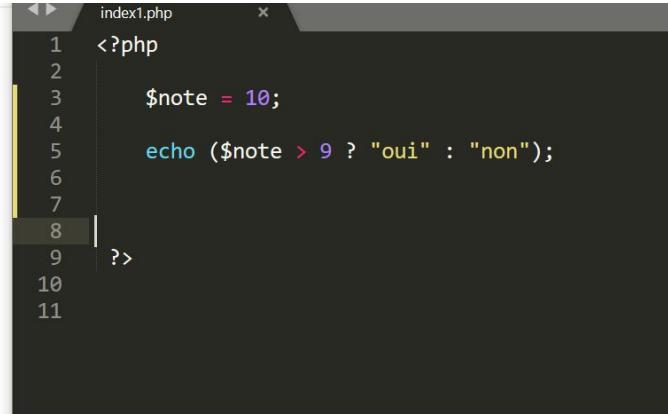
Par contre, si mon texte ne commence pas par un nombre, il ne le convertira pas.

Vous avez juste la moyenne

```
index1.php
1 <?php
2
3     $note = (int)"10";
4
5     if($note >= 10){
6
7         echo "Bravo, vous avez au dessus de la moyenne";
8
9     }elseif($note === 10){
10
11         echo "Vous avez juste la moyenne";
12
13     }else{
14
15         echo "Dommage, vous n'avez pas la moyenne";
16
17     }
18
19
20
21 ?>
22
23
```

Il est possible de demander à php de “forcer” un type de résultat. Il suffit de passer entre parenthèses et avant la variable le type voulu. PHP s’efforcera de convertir la valeur dans le type souhaité.

oui



A screenshot of a code editor window titled "index1.php". The code is as follows:

```
index1.php
1 <?php
2
3     $note = 10;
4
5     echo ($note > 9 ? "oui" : "non");
6
7
8 | ?>
9
10
11
```

The code uses a ternary operator to output "oui" if \$note is greater than 9, and "non" otherwise. Line 8 contains a closing tag that is partially cut off.

On peut également écrire les “if...else” sous une autre forme. A condition de ne pas avoir plus de 2 choix.

Tu as la moyenne

A screenshot of a code editor window titled "index1.php". The code is written in PHP and contains a switch-case statement. The code is as follows:

```
<?php
$note = (int)"10";
switch ($note) {
    case 10:
        echo "Tu as la moyenne";
        break;
    case 9:
        echo "Tu n'as pas la moyenne";
        break;
    default:
        echo 'note inconnue';
        break;
}
?>
```

De même que pour les “if” et “else”, il est possible d’utiliser un “switch case”. A la différence du “if”, le “switch case” ne peut comparer que des valeurs égales à... (case). Le “switch case” ne peut pas comparer si une valeur est plus grande ou plus petite.

c'est ouvert !

The screenshot shows a code editor window with a dark theme. The file is named 'index1.php'. The code contains a PHP script that checks the value of \$heure (hour). If \$heure is between 8 and 12 or between 14 and 19, it outputs 'c'est ouvert !' (it's open!). Otherwise, it outputs 'c'est fermé !' (it's closed!). The code is numbered from 1 to 17. The cursor is at the end of line 15, after the closing tag '?>'.

```
<?php
$heure = 10;
if( $heure >= 8 && $heure < 12 || $heure >= 14 && $heure < 19 ){
    echo "c'est ouvert ! ";
} else{
    echo "c'est fermé ! ";
}
?>
```

Il est également possible que nous ayons besoin qu'une valeur rentre dans un créneau.  
En php, on ne peut pas écrire ET ou encore OU. Leur équivalent est **&&** (et) et **||** (ou).  
Dans le code au dessus, je lui dis que si \$heure est plus grande ou égal à 8 ET inférieur  
à 12 OU que \$heure est supérieur ou égal à 14 ET inférieur à 19, alors c'est ouvert.

c'est ouvert !

The screenshot shows a code editor window titled "index1.php". The code is as follows:

```
<?php  
$heure = 10;  
if( $heure >= 8 && $heure < 12 || $heure >= 14 && $heure < 19 ){  
    echo "c'est ouvert ! ";  
}else{  
    echo "c'est fermé ! ";  
}  
?>
```

The line "c'est ouvert ! " is highlighted in green, indicating it is the output of the script.

VRAI && VRAI = VRAI

VRAI && FAUX = FAUX

FAUX && FAUX = FAUX

VRAI || VRAI = VRAI

VRAI || FAUX = VRAI

FAUX || FAUX = FAUX

# A vous de jouer !

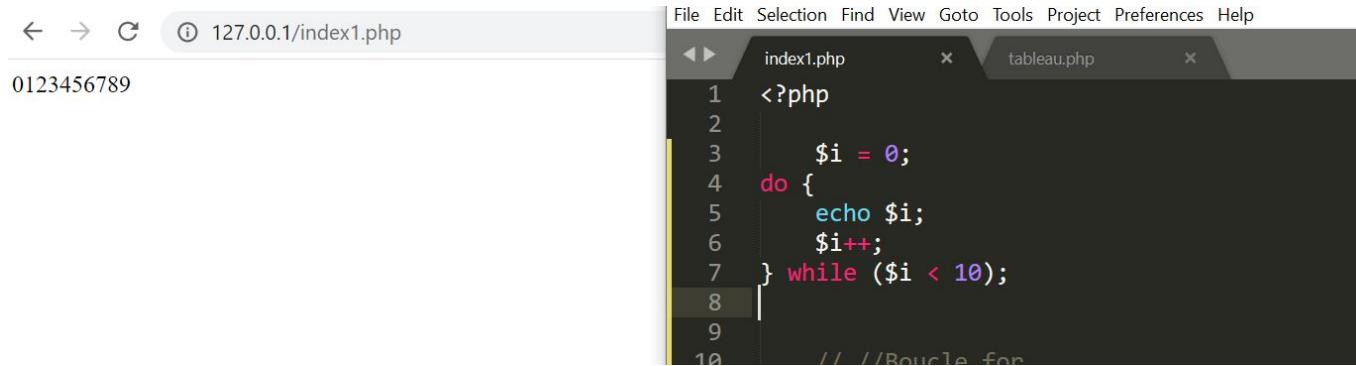
# **Les Boucles**

```
index1.php
```

```
1 <?php
2
3     //Boucle tant que
4     while ( <= 10) {
5         # code...
6     }
7
8
9     //Boucle for
10    for ($i=0; $i < ; $i++) {
11        # code...
12    }
13
14
15    //Boucle Pour chaque
16    foreach ($variable as $key => $value) {
17        # code...
18    }
19
20    |
21    ?>
22
23
```

```
File Edit Selection Find View Goto Tools Project Preferences Help
index1.php    tableau.php
1 <?php
2
3     $i = 1;
4     while ($i <= 10) {
5         echo $i++.<br>; /* La valeur affichée est $i avant
6                         l'incrémentation
7                         (post-incrémantation) */
8     }
9
10    echo "<hr>";|/* exemple 2 */
11
12    $i = 1;
13    while ($i <= 10):
14        echo $i;
15        $i++;
16    endwhile;
17
18
```

La signification d'une boucle while est très simple. PHP exécute l'instruction tant que l'expression de la boucle while est évaluée comme true. La valeur de l'expression est vérifiée à chaque début de boucle, et, si la valeur change durant l'exécution de l'instruction, l'exécution ne s'arrêtera qu'à la fin de l'itération (chaque fois que PHP exécute l'instruction, on appelle cela une itération). Si l'expression du while est false avant la première itération, l'instruction ne sera jamais exécutée.



The screenshot shows a web browser window with the URL 127.0.0.1/index1.php. The browser's toolbar includes back, forward, and refresh buttons. The main content area displays a code editor with two tabs: index1.php and tableau.php. The index1.php tab is active, showing the following PHP code:

```
1 <?php
2
3     $i = 0;
4 do {
5     echo $i;
6     $i++;
7 } while ($i < 10);
8
9
10 // //Boucle for
```

Il existe aussi la boucle “do-while” qui est une boucle “while” qui s’ignore ^^

1  
2  
3  
4  
5  
6  
7  
8  
9  
10

```
index1.php    tableau.php
1 <?php
2
3
4     /* exemple 1 */
5     for ($i = 1; $i <= 10; $i++) {
6         echo "$i <br>";
7     }
8
9
10
11
```

La première expression (expr1) est évaluée (exécutée), quoi qu'il arrive au début de la boucle.

Au début de chaque itération, l'expression expr2 est évaluée. Si l'évaluation vaut true, la boucle continue et les commandes sont exécutées. Si l'évaluation vaut false, l'exécution de la boucle s'arrête.

À la fin de chaque itération, l'expression expr3 est évaluée (exécutée).

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

```
index1.php    tableau.php  
1 <?php  
2  
3  
4     /* exemple 2 */  
5  
6     for ($i = 1; ; $i++) {  
7         if ($i > 10) {  
8             break;  
9         }  
10        echo $i.'<br>';  
11    }  
12  
13
```

Les expressions peuvent éventuellement être laissées vides ou peuvent contenir plusieurs expressions séparées par des virgules. Dans expr2, toutes les expressions séparées par une virgule sont évaluées mais le résultat est obtenu depuis la dernière partie. Si l'expression expr2 est laissée vide, cela signifie que c'est une boucle infinie (PHP considère implicitement qu'elle vaut true, comme en C). Cela n'est pas vraiment très utile, à moins que vous souhaitiez terminer votre boucle par l'instruction conditionnelle break.

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

```
index1.php  
1 <?php  
2  
3  
4     /* exemple 3 */  
5  
6     $i = 1;  
7     for (; ; ) {  
8         if ($i > 10) {  
9             break;  
10        }  
11        echo $i."<br>";  
12        $i++;  
13    }  
14  
15
```

Bon, je pose ça là,  
vous en faites ce que vous voulez mais sachez qu'on peut faire ça

```
1 - 1  
2 - 3  
3 - 6  
4 - 10  
5 - 15  
6 - 21  
7 - 28  
8 - 36  
9 - 45  
10 - 55
```

```
index.php  
1 <?php  
2  
3  
4  
5     for ($i = 1, $j = 0; $i <= 10; $j += $i, print "$i - $j<br>", $i++);  
6  
7  
8  
9  
10    // //Boucle Pour chaque  
11    // foreach ($variable as $key => $value) {  
12    //     # code...  
13    // }  
14  
15 |  
16 ?>  
17  
18
```

Il est possible de tout mettre sur une ligne et d'avoir plusieurs valeurs.

Nouvel opérateur, `+=` (même si c'est plus un raccourci)

Cela signifie que `$j` est égal à la somme de `$i` et `$j`

Equivalent : 
$$\$j = \$i + \$j;$$

Du coup, `=-` c'est la même chose mais en décrémentant.

The screenshot shows a code editor with two tabs: "index.php" and "tableau.php".

**index.php:**

```
1 sandra
2 nicolas
3 kevin
4 julie
5 manon
```

**tableau.php:**

```
1 <?php
2
3     $eleves = ["sandra", "nicolas", "kevin", "julie", "manon"];
4
5     foreach ($eleves as $eleve) {
6         echo "$eleve <br>";
7     }
8
9     echo "<hr>";
10
11
12     foreach(["sandra", "nicolas", "kevin", "julie", "manon"] as $eleve) {
13         echo "$eleve <br>";
14     }
15
16
17
18 ?>
19
20
21
```

À chaque itération, la valeur de l'élément courant est assignée à \$eleve.

En gros, la boucle va parcourir le tableau et à chaque tour, \$eleve prendra la valeur de la ligne en cours.

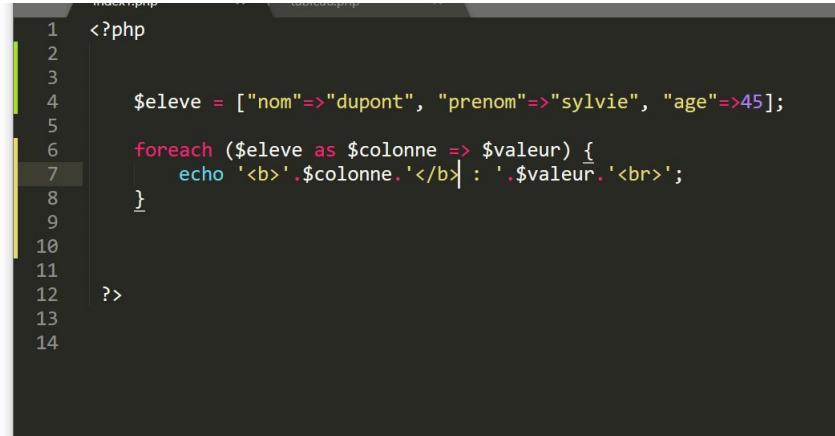
```
sandra  
nicolas  
kevin  
julie  
manon
```

```
sandra  
nicolas  
kevin  
julie  
manon
```

```
index.php          tableau.php  
1  <?php  
2  
3      $eleves = ["sandra", "nicolas", "kevin", "julie", "manon"];  
4  
5  
6      for ($i=0; $i < count($eleves); $i++) {  
7          echo "$eleves[$i] <br>";  
8      }  
9  
10     echo "<hr>";  
11  
12     $ii=0;  
13     while ( $ii < count($eleves)) {  
14         echo "$eleves[$ii] <br>";  
15         $ii++;  
16     }  
17  
18  
19  
20     ?>  
21  
22
```

Notons, qu'il est aussi possible de parcourir ce tableau indicé avec une boucle for et aussi while

```
nom : dupont
prenom : sylvie
age : 45
```



```
index.php
1 <?php
2
3
4     $eleve = ["nom"=>"dupont", "prenom"=>"sylvie", "age"=>45];
5
6     foreach ($eleve as $colonne => $valeur) {
7         echo '<b>' . $colonne . '</b> : ' . $valeur . '<br>';
8     }
9
10
11
12 ?>
13
14
```

La particularité de la boucle `foreach`, c'est sa facilité à récupérer le nom des colonnes.

Attention, notons que le tableau “\$eleve” n'a qu'une seule ligne.

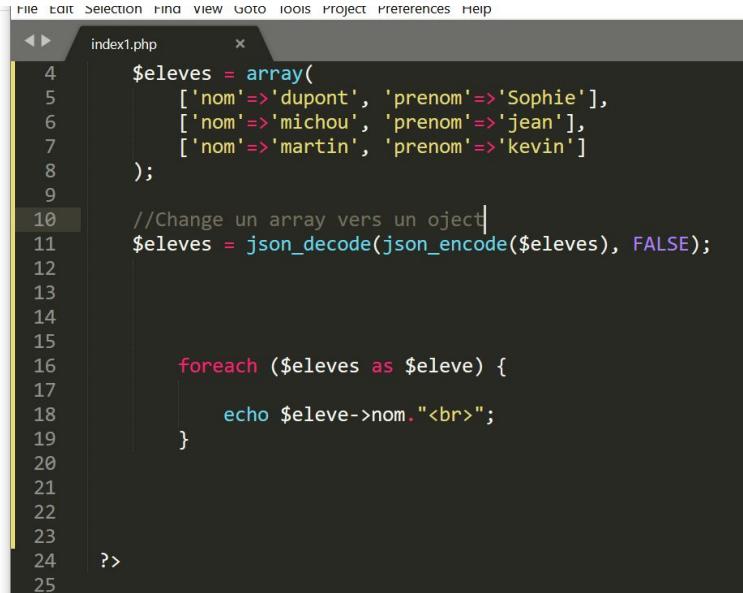
dupont  
michou  
martin

```
index1.php
4     $eleves = [
5         ['nom'=>'dupont', 'prenom'=>'Sophie'],
6         ['nom'=>'michou', 'prenom'=>'jean'],
7         ['nom'=>'martin', 'prenom'=>'kevin']
8     ];
9
10
11
12
13
14     foreach ($eleves as $eleve) {
15
16         echo $eleve['nom'].'
17     }
18
19
20
21
22     ?>
23
24
```

Sur un tableau associatif à plusieurs lignes, si l'on souhaite récupérer la valeur d'une colonne sur chaque ligne, il faut spécifier le nom de la colonne entre crochets (`$eleve['nom_de_la_colonne']`).

C'est plus lisible que de mettre un index.

```
dupont  
michou  
martin
```



The screenshot shows a code editor window with the file 'index1.php' open. The code defines an array '\$eleves' with three elements: 'dupont', 'michou', and 'martin', each associated with a name and a prename. It then uses json\_encode to convert this array into a JSON object, which is then decoded back into an object '\$eleves'. A foreach loop iterates over '\$eleves' to echo the 'nom' property of each element. The code is as follows:

```
File Edit Selection Find View Goto Tools Project Preferences Help
index1.php
4     $eleves = array(
5         ['nom'=>'dupont', 'prenom'=>'Sophie'],
6         ['nom'=>'michou', 'prenom'=>'jean'],
7         ['nom'=>'martin', 'prenom'=>'kevin']
8     );
9
10    //Change un array vers un object
11    $eleves = json_decode(json_encode($eleves), FALSE);
12
13
14
15
16    foreach ($eleves as $eleve) {
17
18        echo $eleve->nom."<br>";
19    }
20
21
22
23
24    ?>
25
```

ATTENTION, si vous “bouclez” sur un objet et non sur un tableau (c'est souvent le cas dans wordpress), la syntaxe change légèrement. Pour récupérer la valeur d'une colonne, on ne met plus de crochets mais une flèche (->).

The screenshot shows a web browser at the URL 127.0.0.1/index1.php displaying the output of a PHP var\_dump() function. Below the browser is a Sublime Text editor window showing the corresponding PHP code.

Browser Output (index1.php):

```
string(6) "coucou"
int(2)
array(3) { [0]=> string(6) "coucou" [1]=> string(2) "ca" [2]=> string(2) "va" }
float(2.8)
object(stdClass)#1 (3) { ["prenom"]=> string(5) "kevin" ["nom"]=> string(6) "durand" ["notes"]=> array(3) { [0]=> int(12) [1]=> int(14) [2]=> string(3) "hey" } }
```

Sublime Text Editor (index1.php):

```
<?php |
1 $string = "coucou";
2 var_dump($string);
3 echo "<hr>";
4 $int = 2;
5 var_dump($int);
6 echo "<hr>";
7 $array = ['coucou', 'ca', 'va'];
8 var_dump($array);
9 echo "<hr>";
10 $float = 2.8;
11 var_dump($float);
12 echo "<hr>";
13 $obj = json_decode(json_encode(["prenom"=>'kevin', "nom"=>'durand', "notes"=>[12, 14, "hey"]]), FALSE);
14 var_dump($obj);
```

Petite diapositive en marge des boucles, pour savoir à quel type de variable vous avez à faire, vous pouvez (devez) utiliser la fonction php var\_dump(\$variable\_a\_verifier);

dupont  
michou  
martin

```
File Edit Selection Find View Goto Tools Project Preferences Help
index1.php x
1 <?php
2
3     $eleves = [
4         ['nom'=>'dupont', 'prenom'=>'Sophie'],
5         ['nom'=>'michou', 'prenom'=>'jean'],
6         ['nom'=>'martin', 'prenom'=>'kevin']
7     ];
8
9
10    for ($i=0; $i < count($eleves) ; $i++) {
11        echo $eleves[$i]['nom']."<br>";
12    }
13
14    ?>
15
16
17
```

Encore une fois, vous pouvez utiliser une boucle for (while aurait fait l'affaire aussi) pour lire ce type de tableau. Par contre, pour récupérer les colonnes avec une boucle for, c'est beaucoup plus laborieux !

← → ⌂ ① 127.0.0.1/index1.php

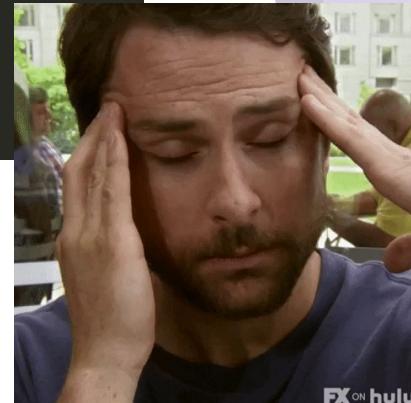
```
nom : dupont  
prenom : Sophie  
nom : michou  
prenom : jean  
nom : martin  
prenom : kevin
```

File Edit Selection Find View Goto Tools Project Preferences Help

index1.php

```
1 <?php  
2  
3     $eleves = [  
4         ['nom'=>'dupont', 'prenom'=>'Sophie'],  
5         ['nom'=>'michou', 'prenom'=>'jean'],  
6         ['nom'=>'martin', 'prenom'=>'kevin']  
7     ];  
8  
9  
10    for ($i=0; $i < count($eleves) ; $i++) {  
11        for ($u=0; $u < 2; $u++) {  
12            echo array_keys($eleves[$i])[$u]. ' .' . $eleves[$i][array_keys($eleves[$i])[$u]]. '<br>;  
13        }  
14    }  
15  
16 ?>  
17  
18
```

La preuve que l'on peut.... Après si vous voulez enfoncez un clou avec une masse....



```
nom : dupont
prenom : Sophie
nom : michou
prenom : jean
nom : martin
prenom : kevin
```

```
index1.php          tableau.php
4      $eleves = [
5          ['nom'=>'dupont', 'prenom'=>'Sophie'],
6          ['nom'=>'michou', 'prenom'=>'jean'],
7          ['nom'=>'martin', 'prenom'=>'kevin']
8      ];
9
10     $display="";
11
12     foreach ($eleves as $eleve) {
13
14         foreach ($eleve as $colonne => $valeur) {
15             $display .= "$colonne : $valeur <br>";
16         }
17     }
18 }
19
20 echo $display;
21 ?>
22
23
24
25 |
```

Sur un tableau associatif avec plusieurs lignes, si l'on souhaite récupérer le nom de chaque colonne et sa valeur, il faudra 2 boucles.

La première aura pour mission d'aller sur chaque ligne du tableau (ligne 1, 2, 3...)  
La deuxième parcourra la ligne pour récupérer chaque colonne et la valeur associée...

Notice: Array to string conversion in C:\xampp4\htdocs\index1.php on line 15

Notice: Array to string conversion in C:\xampp4\htdocs\index1.php on line 15

Notice: Array to string conversion in C:\xampp4\htdocs\index1.php on line 15

```
nom : dupont
prenom : Sophie
notes : Array
nom : michou
prenom : jean
notes : Array
nom : martin
prenom : kevin
notes : Array
```

```
File Edit Selection Find View Goto Tools Project Preferences Help
index.php x tableau.php x
4     $eleves = [
5         ['nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,10]],
6         ['nom'=>'michou', 'prenom'=>'jean', 'notes'=>[11,13,16]],
7         ['nom'=>'martin', 'prenom'=>'kevin', 'notes'=>[15,9,14]]
8     ];
9
10    $display="";
11
12    foreach ($eleves as $eleve) {
13
14        foreach ($eleve as $colonne => $valeur) {
15            $display .= "$colonne : $valeur <br>";
16        }
17    }
18
19 }
20
21 echo $display;
22 ?>
23
24
```

Bon là, on corse un peu les choses... La difficulté est que la valeur de "notes" est un array  
(tableau)

Ce que je souhaite :

```
nom : dupont
prenom : Sophie
notes : 10, 20, 10,
nom : michou
prenom : jean
notes : 10, 20, 10, 11, 13, 16,
nom : martin
prenom : kevin
notes : 10, 20, 10, 11, 13, 16, 15, 9, 14,
```

```
nom : dupont
prenom : Sophie
notes : 10, 20, 10,
nom : michou
prenom : jean
notes : 10, 20, 10, 11, 13, 16,
nom : martin
prenom : kevin
notes : 10, 20, 10, 11, 13, 16, 15, 9, 14,
```

```
index.php          database.php
4      $eleves = [
5          ['nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,10]],
6          ['nom'=>'michou', 'prenom'=>'jean', 'notes'=>[11,13,16]],
7          ['nom'=>'martin', 'prenom'=>'kevin', 'notes'=>[15,9,14]]
8      ];
9
10     $notes="";
11     $display="";
12
13     foreach ($eleves as $eleve) {
14
15         foreach ($eleve as $colonne => $valeur) {
16
17             if(is_array($valeur)){
18
19                 foreach ($valeur as $v) {
20                     $notes .= "$v, ";
21                 }
22
23                 $valeur = $notes;
24             }
25
26             $display .= "$colonne : $valeur <br>";
27         }
28     }
29
30     echo $display;
31
32 ?>
```

Solution, un troisième foreach !

← → ⌂ 127.0.0.1/index1.php

```
nom : dupont
prenom : Sophie
notes : 10,12,20,
nom : michou
prenom : jean
notes : 11,13,16,
nom : martin
prenom : kevin
notes : 15,9,14,
```

File Edit Selection Find View Goto Tools Project Preferences Help

index1.php    tableaux.php    contact.php    404.php    header.php

```
3     $eleves = [
4         ['nom'=>'dupont', 'prenom'=>'Sophie','notes'=>[10,12,20]],
5         ['nom'=>'michou', 'prenom'=>'jean','notes'=>[11,13,16]],
6         ['nom'=>'martin', 'prenom'=>'kevin','notes'=>[15,9,14]]
7     ];
8
9     $d="";
10    foreach ($eleves as $eleve) {
11
12        foreach ($eleve as $key => $value) {
13
14            if(is_array($value)){
15                $n="";
16
17                foreach ($value as $valeur) {
18                    $n .= $valeur.',';
19                }
20                $value = $n;
21            }
22
23            $d .= "$key : $value <br>";
24
25
26        }
27    }
28
29    echo $d;
30}
```

Erratum....

# A vous de jouer !

# **Les Fonctions**

```
<?php  
  
    print_r(expression);  
    array_keys(input);  
    count(var);  
  
?>
```

Depuis le début, nous avons déjà vu certaines fonctions.

Elles permettent d'effectuer une partie de logique et de renvoyer un résultat.

Elles permettent d'éviter de se répéter et d'effectuer des traitements spécifiques.

Les fonction ont un nom (qui respecte la même convention que les variables), puis de parenthèses. Dans ces parenthèses, il peut y avoir un ou plusieurs paramètres (séparés par des virgules). Ces paramètres peuvent être des variables ou des valeurs.

```
4  
-----  
4  
index1.php  
1 <?php  
2  
3     $arr = [1,2,3,4];  
4  
5     echo count($arr);  
6  
7     ////////////////  
8     echo "<hr>";  
9     ////////////////  
10  
11    $res = count($arr);  
12    echo $res;  
13  
14  
15    ?>  
16  
17
```

Les résultats envoyés par les fonctions peuvent soit être directement affichés ou stockés dans une variable pour être réutilisés plus tard.

Pour savoir comment utiliser une fonction, il suffit de se rendre sur la documentation de php (<https://www.php.net/manual/fr/funcref.php>)

Elles sont rangées par familles.

Submit a Pull Request   Report a Bug

## count

(PHP 4, PHP 5, PHP 7, PHP 8)

count — Compte tous les éléments d'un tableau ou quelque chose d'un objet

### Description

```
count ( Countable|array $value , int $mode = COUNT_NORMAL ) : int
```

Compte tous les éléments d'un tableau ou quelque chose d'un objet.

Pour les objets, **count()** retourne le nombre de propriétés non-statiques, sans tenir compte de la visibilité. Si [SPL](#) est disponible, vous pouvez utiliser la fonction **count()** en implémentant l'interface [Countable](#). Cette interface a exactement une méthode, [Countable::count\(\)](#), qui retourne la valeur renournée par la fonction **count()**.

Reportez-vous à la section sur les [Tableaux](#) du manuel, pour plus de détails sur le fonctionnement des tableaux en PHP.

<https://www.php.net>

[array\\_change\\_key\\_case](#)[array\\_chunk](#)[array\\_column](#)[array\\_combine](#)[array\\_count\\_values](#)[array\\_diff\\_assoc](#)[array\\_diff\\_key](#)[array\\_diff\\_uassoc](#)[array\\_diff\\_ukey](#)[array\\_diff](#)[array\\_fill\\_keys](#)[array\\_fill](#)[array\\_filter](#)[array\\_flip](#)[array\\_intersect\\_assoc](#)[array\\_intersect\\_key](#)[array\\_intersect\\_uassoc](#)[array\\_intersect\\_ukey](#)[array\\_intersect](#)[array\\_key\\_exists](#)[array\\_key\\_first](#)[array\\_key\\_last](#)[array\\_keys](#)

Si je ne savais pas comment utiliser la fonction “count()”, j’irais chercher sur php “count” (<https://www.php.net/manual/fr/function.count.php>)

The screenshot shows the official PHP documentation website. At the top, there's a navigation bar with links for 'php', 'Downloads', 'Documentation', 'Get Involved', and 'Help'. The 'Documentation' link is highlighted. To the right of the navigation is a search bar. Below the navigation, the main content area has a dark header 'Liste de paramètres'. The first parameter listed is 'value', described as 'Un tableau ou un objet [Countable](#)'. The second parameter is 'mode', with a note explaining it counts recursively if set to COUNT\_RECURSIVE. A yellow callout box titled 'Attention' warns about potential infinite loops and E\_WARNING errors. Below these, a section 'Valeurs de retour' states it returns the number of elements in 'value'. To the right of the main content, a sidebar lists many other array-related functions like array\_map, array\_merge\_recursive, etc.

**Liste de paramètres**

**value**  
Un tableau ou un objet [Countable](#).

**mode**  
Si le paramètre optionnel **mode** vaut **COUNT\_RECURSIVE** (ou 1), **count()** va compter récursivement les tableaux. C'est particulièrement pratique pour compter le nombre d'éléments d'un tableau.

**Attention** La fonction **count()** peut détecter les récursions afin d'éviter les boucles infinies, mais émettra une alerte de type **E\_WARNING** à chaque fois qu'une boucle infinie surviendra (dans le cas où un tableau contient lui-même plus d'une boucle infinie) et retournera un compteur plus grand que l'attendu.

**Valeurs de retour**

Retourne le nombre d'éléments dans **value**. Quand le paramètre n'est ni un [tableau](#), ni un [objet](#) qui implémente l'interface [Countable](#), 1 sera retourné. Il y a néanmoins une exception : si le paramètre **value** vaut **null**, 0 sera retourné.

array\_map  
array\_merge\_recursive  
array\_merge  
array\_multisort  
array\_pad  
array\_pop  
array\_product  
array\_push  
array\_rand  
array\_reduce  
array\_replace\_recursive  
array\_replace  
array\_reverse  
array\_search  
array\_shift  
array\_slice  
array\_splice  
array\_sum  
array\_udiff\_assoc  
array\_udiff\_uassoc  
array\_udiff  
array\_uintersect\_assoc  
array\_uintersect\_uassoc  
array\_uintersect

En allant plus bas, je vois la liste des paramètres à utiliser, si ils sont obligatoires ou optionnels, leur type. La documentation me donne aussi des informations sur ce que me renvoie la fonction. Je vois qu'il y a un mode en 2ème paramètre qui est optionnel. Je peux l'activer en mettant “1”.

En clair, **count()** va me compter combien il y a d'élément dans le tableau mais si un élément est un tableau, il ne me dira pas combien cet élément a d'éléments. Sauf si j'utilise le mode récursif.

```
index1.php
1 <?php
2
3     $arr = [1,2,3,4,[1,2]];
4
5     echo count($arr,1);
6
7     ///////////////////////////////////////////////////
8     echo "<hr>";
9     ///////////////////////////////////////////////////
10
11    $res = count($arr);
12    echo $res;
13
14
15    ?>
16
17
```

Si je teste, je vois que mon premier count() me renvoie 7 alors que le deuxième me renvoie 5.

En fait, le premier à le mode récursif d'activer (avec le paramètre 1), il me compte donc tous les éléments de tous les tableaux.

Alors que mon deuxième count() reste basique, il compte combien j'ai d'éléments dans le tableau \$arr (5)

## Description

---

```
count ( Countable|array $value , int $mode = COUNT_NORMAL ) : int
```

Ici, c'est ce qu'on appelle la signature de la fonction. En un coup d'oeil, on voit comment l'utiliser.

## Description

```
count ( Countable|array $value , int $mode = COUNT_NORMAL ) : int
```

Le nom  
de la  
fonction

Le type des paramètres

Quand un paramètre à  
Le symbole “=”, c'est  
qu'il est optionnel

Le type renvoyé par la  
fonction

# array\_chunk

(PHP 4 >= 4.2.0, PHP 5, PHP 7, PHP 8)

array\_chunk – Sépare un tableau en tableaux de taille inférieure

## Description

```
array_chunk ( array $array , int $length , bool $preserve_keys = false ) : array
```

Sépare le tableau **array** en plusieurs tableaux comptant **length** éléments. Il est aussi possible que le dernier tableau contienne moins de valeurs.

Le nom  
de la  
fonction

Le type des paramètres

On voit que le 3ème  
paramètre est  
optionnel

Le type renvoyé par la  
fonction

```
Array ( [0]=> Array ( [0]=> dupont [1]=> Sophie ) [1]=> Array ( [0]=> Array ( [0]=> 10 [1]=> 20 [2]=> 10 )) )
```

```
Array ( [0]=> Array ( [nom]=> dupont [prenom]=> Sophie ) [1]=> Array ( [notes]=> Array ( [0]=> 10 [1]=> 20 [2]=> 10 )) )
```

C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
index1.php      tableau.php
1 <?php
2
3     $input_array = [ 'nom'=>'dupont', 'prenom'=>'Sophie', 'notes'=>[10,20,10]];
4     print_r(array_chunk($input_array, 2));
5     echo "<hr>";
6     print_r(array_chunk($input_array, 2, true));
7
8
9 ?>
10
11
```

Et voici la fonction en action

# var\_dump

(PHP 4, PHP 5, PHP 7, PHP 8)

var\_dump — Affiche les informations d'une variable

## Description

```
var_dump ( mixed $value , mixed ...$values ) : void
```

**var\_dump()** affiche les informations structurées d'une variable, y compris son type et sa valeur. Les tableaux et les objets sont explorés récursivement, avec des indentations, pour mettre en valeur leur structure.

Mixed veut dire “tous types”

... signifie que l'on peut mettre autant de paramètres que l'on souhaite. Cela veut aussi dire que c'est optionnel.

Void signifie que la fonction ne Renvoie rien.

← → C

127.0.0.1/index1.php

```
array(2) { [0]=> string(6) "coucou" [1]=> string(3) "toi" } int(23) string(9) "hey ca va" float(45.89)
```

C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)  
File Edit Selection Find Goto Tools Project Preferences Help

```
index1.php      tableau.php  
1 <?php  
2  
3 |  
4 |     var_dump(['coucou', 'toi'], 23, 'hey ca va', 45.89);  
5  
6  
7 ?>  
8  
9
```

<https://www.php.net/manual/fr/function.is-real.php>

The screenshot shows the PHP documentation page for the `is_real()` function. The URL in the address bar is `https://www.php.net/manual/fr/function.is-real.php`. The page header includes the PHP logo, navigation links for Downloads, Documentation, Get Involved, Help, and a search bar. A red circle highlights the text "(PHP 4, PHP 5, PHP 7, PHP 8) is\_real – Alias de [is\\_float\(\)](#)". The main content area has a dark background with white text. It contains a "Description" section stating "Cette fonction est un alias de : [is\\_float\(\)](#)". A pink warning box says "Avertissement Cet alias est OBSOLÈTE en PHP 7.4.0, et SUPPRIMÉ à partir de PHP 8.0.0.". Below this is a "User Contributed Notes" section with a note button and a "0" count. The footer shows the URL again.

Change language: French

Submit a Pull Request Report a Bug

## is\_real

(PHP 4, PHP 5, PHP 7, PHP 8)  
is\_real – Alias de [is\\_float\(\)](#)

### Description

Cette fonction est un alias de : [is\\_float\(\)](#).

**Avertissement** Cet alias est OBSOLÈTE en PHP 7.4.0, et SUPPRIMÉ à partir de PHP 8.0.0.

### User Contributed Notes

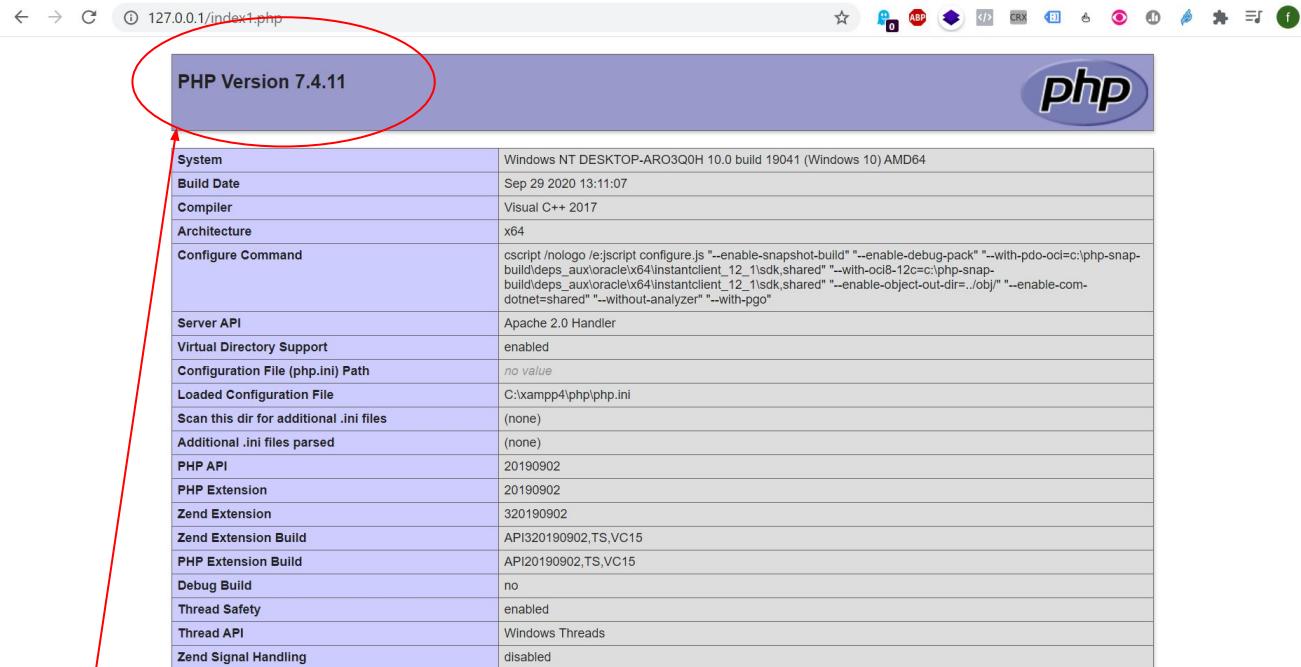
There are no user contributed notes for this page.

https://www.php.net

Sur cette fonction, on peut voir que c'est un alias d'une autre fonction (`is_float()`).

Par défaut, je préfère prendre la fonction de base plutôt que son “équivalence”.

De plus, nous pouvons voir que cette fonction a été dépréciée. Elle est obsolète depuis PHP 7.4 et abandonnée en php 8.0



PHP Version 7.4.11	
	
System	Windows NT DESKTOP-ARO3Q0H 10.0 build 19041 (Windows 10) AMD64
Build Date	Sep 29 2020 13:11:07
Compiler	Visual C++ 2017
Architecture	x64
Configure Command	cscript /nologo /e:json configure.js --enable-snapshot-build --enable-debug-pack --with-pdo-oci=c:\php-snap-build\deps_aux\oracle\x64\instantclient_12_1\ sdk\shared --with-oci8=12;c:\php-snap-build\deps_aux\oracle\x64\instantclient_12_1\ sdk\shared --enable-object-out-dir=..obj --enable-com-dotnet=shared --without-analyzer --with-pgo
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	no value
Loaded Configuration File	C:\xampp4\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,TS,VC15
PHP Extension Build	API20190902,TS,VC15
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads
Zend Signal Handling	disabled

Au fait, pour connaître sa version de php, il y a une fonction pour ça ^^  
phpinfo();

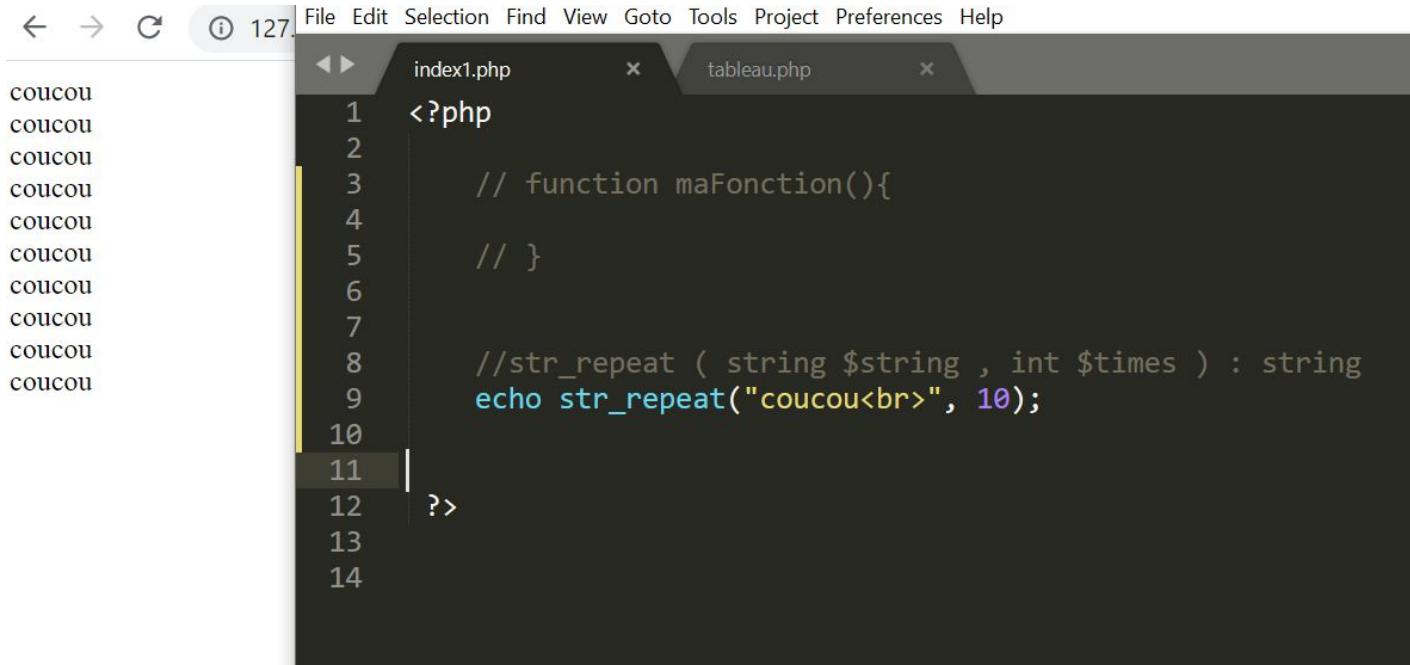
Voici ma version de php.

```
<?php  
  
    function maFonction(){  
        |  
        }  
    |
```

Ok, depuis tout à l'heure on a vu des fonctions natives de php.

Mais savez-vous que l'on peut créer nos propres fonctions (et on le fait très souvent !)

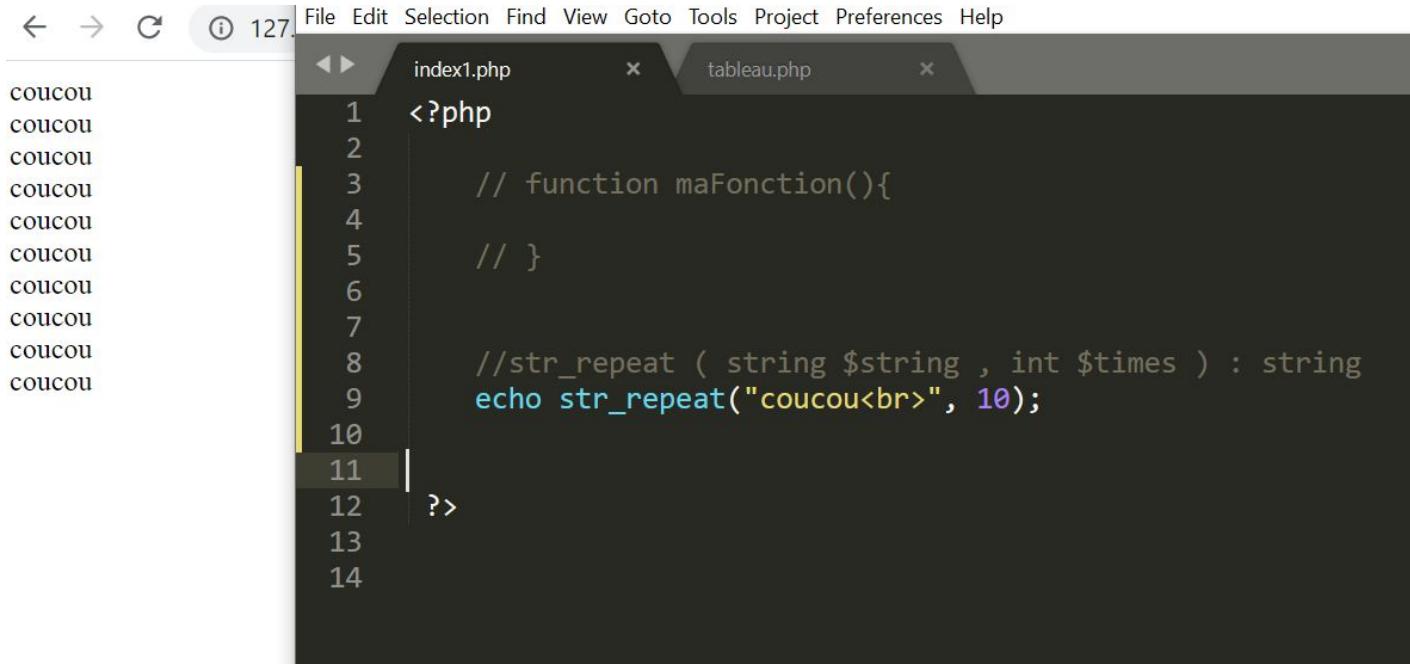
De base, si une fonction fonctionne, c'est soit qu'elle est native au langage, soit qu'on l'a créée.



A screenshot of a code editor window titled "index1.php". The file contains the following PHP code:

```
coucou
1 <?php
2
3     // function maFonction(){
4
5         // }
6
7
8     //str_repeat ( string $string , int $times ) : string
9     echo str_repeat("coucou<br>", 10);
10
11    ?>
12
13
14
```

Nous allons reproduire une fonction native de php comme exercice.  
Pour l'exercice, nous allons reproduire la fonction native de php, str\_repeat()



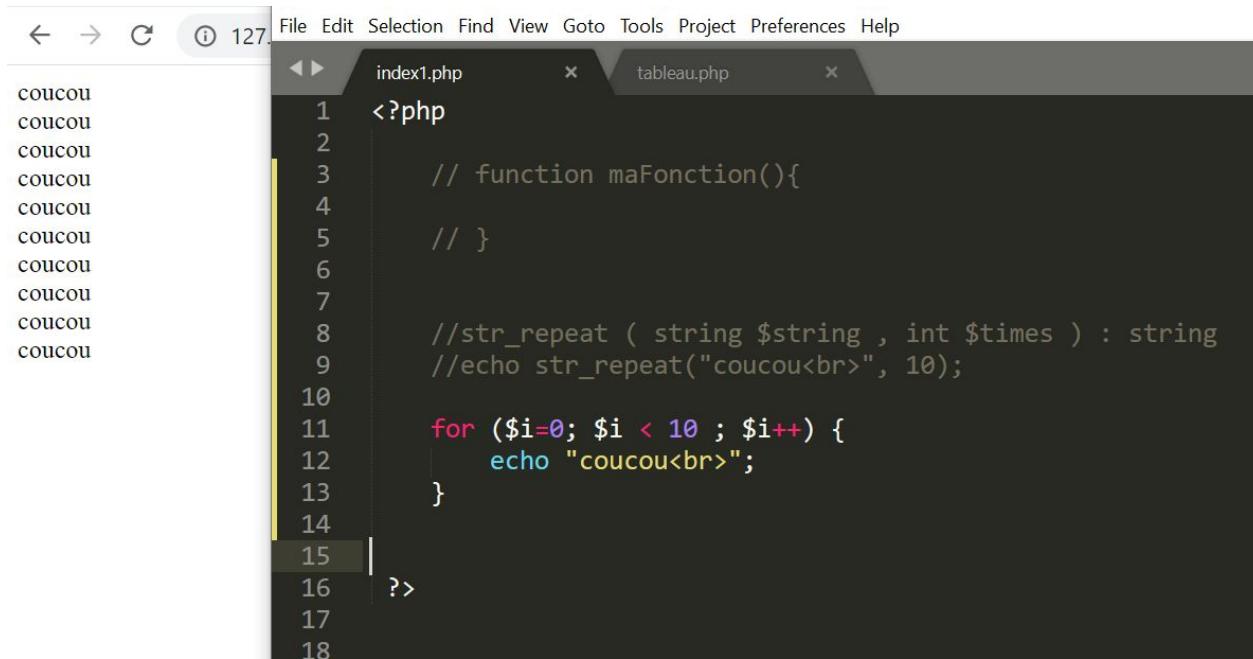
The screenshot shows a code editor window with two tabs: "index1.php" and "tableau.php". The "index1.php" tab is active and contains the following PHP code:

```
1 <?php
2
3     // function maFonction(){
4
5     // }
6
7
8     //str_repeat ( string $string , int $times ) : string
9     echo str_repeat("coucou<br>", 10);
10
11    ?>
12
13
14
```

The code uses the `str_repeat` function to print the string "coucou<br>" ten times, resulting in ten lines of "coucou" separated by line breaks.

Au final, on se rend compte que c'est une fonction qui répète X fois une chaîne de caractère.

Connaît-on une façon de répéter (boucler) un string autant de fois que l'on souhaite ?



The screenshot shows a code editor with two tabs open: "index1.php" and "tableau.php". The "index1.php" tab is active and contains the following PHP code:

```
coucou
1 <?php
2
3     // function maFonction(){
4
5     // }
6
7
8     //str_repeat ( string $string , int $times ) : string
9     //echo str_repeat("coucou<br>", 10);
10
11    for ($i=0; $i < 10 ; $i++) {
12        echo "coucou<br>";
13    }
14
15    ?>
16
17
18
```

La réponse était la boucle “for”.

Maintenant, on doit insérer la boucle dans notre fonction.

The screenshot shows a code editor with two tabs: 'index1.php' and 'tableau.php'. The 'index1.php' tab is active and contains the following PHP code:

```
coucou  
1 <?php  
2  
3     function maFonction(){  
4         for ($i=0; $i < 10 ; $i++) {  
5             echo "coucou<br>";  
6         }  
7     }  
8  
9     maFonction();  
10  
11  
12     //str_repeat ( string $string , int $times ) : string  
13     //echo str_repeat("coucou<br>", 10);  
14  
15  
16  
17  
18     ?>
```

A red arrow points from the text 'Initialisation de la fonction.' to the line 'function maFonction()'. Another red arrow points from the text 'J'ai bien mis ma boucle "for" dans ma fonction "maFonction()" que j'ai ensuite appelée.' to the line 'maFonction();'.

J'ai bien mis ma boucle "for" dans ma fonction "maFonction()" que j'ai ensuite appelée.

Initialisation de la fonction.

Le problème, c'est que l'on veut pouvoir modifier facilement le string et le nombre de fois que l'on veut le répéter...

```
<?php
```

```
function maFonction(){
    for ($i=0; $i < 10 ; $i++) {
        echo "coucou<br>";
    }
}
```



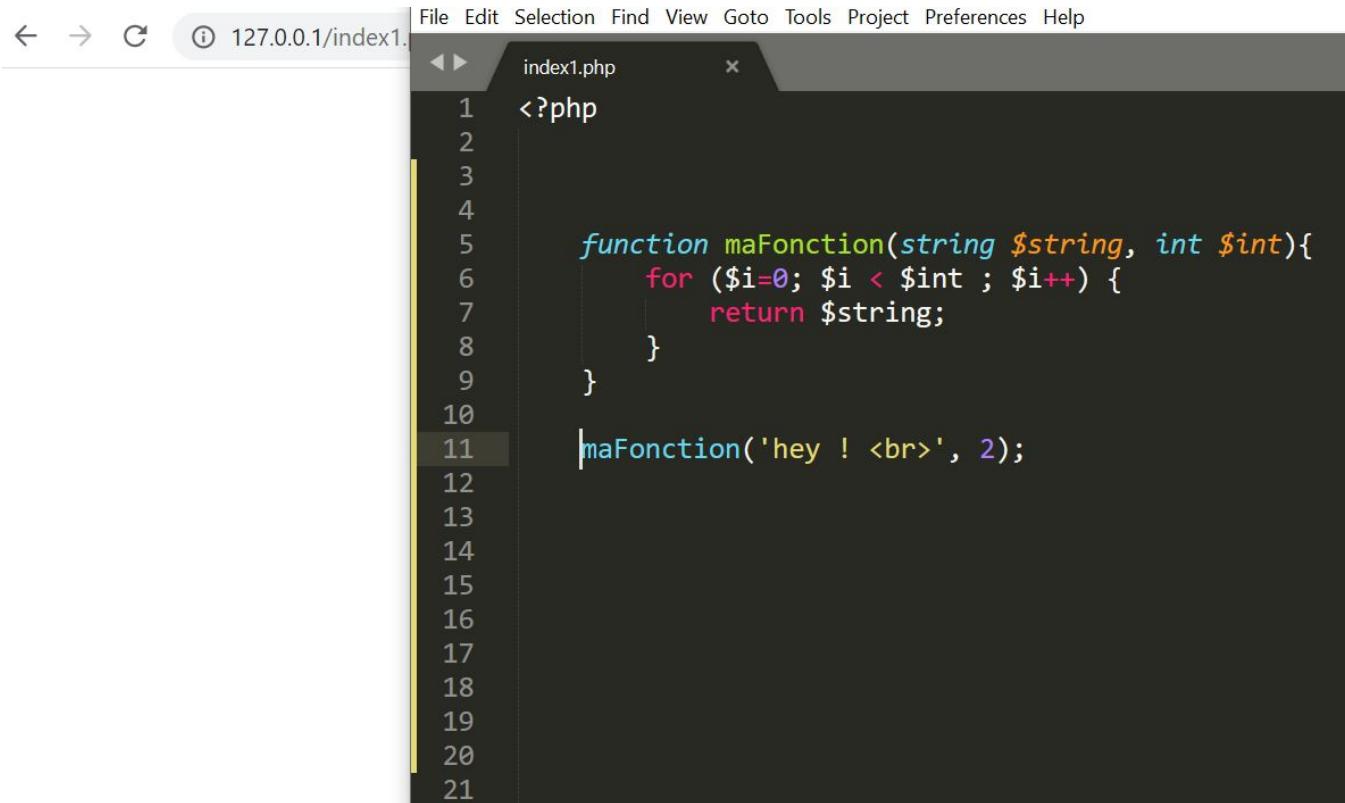
```
function maFonction(string $string, int $int){
    for ($i=0; $i < $int ; $i++) {
        echo $string;
    }
}
```

Il va falloir faire passer des paramètres pour modifier “dynamiquement” ces données. Je souhaite modifier la chaîne de **caractère (string)** et le nombre (int) de fois que l'on veut afficher ce string. Depuis php 7, il est possible de **typer les paramètres** des fonctions. Ce n'est pas obligatoire mais c'est une sécurité !

```
hey !
hey !
les fonctions, c'est facile !
```

```
index1.php  tableau.php
1 <?php
2
3
4
5     function maFonction(string $string, int $int){
6         for ($i=0; $i < $int ; $i++) {
7             echo $string;
8         }
9     }
10
11    maFonction('hey ! <br>', 2);
12    maFonction("les fonctions, c'est facile ! <br>", 20);
13
14
15
16
17
18
19
20
21
22    ?>
23
24
```

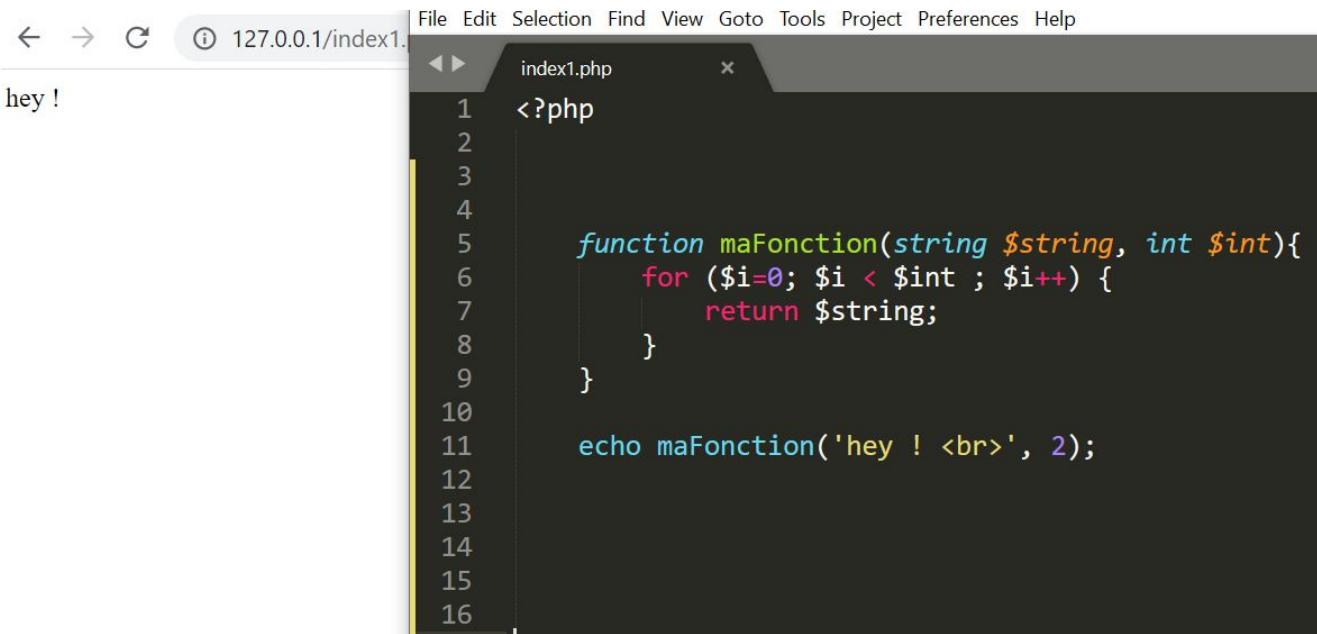
Maintenant, on voit que je peux modifier le texte et le nombre de fois que je souhaite afficher la fonction “à la volée”. Mais je voudrais que le résultat ne s'affiche que si je le souhaite et non systématiquement.



The screenshot shows a code editor window with the file "index1.php" open. The code is as follows:

```
<?php
function maFonction(string $string, int $int){
    for ($i=0; $i < $int ; $i++) {
        return $string;
    }
}
maFonction('hey ! <br>', 2);
```

Pour ce faire, je vais remplacer “echo” par “return”. Du coup, je choisis si je veux afficher ma fonction (je dois mettre “echo” avant) ou la stocker dans une variable. Dans l'exemple au dessus, je n'ai pas mis de “echo”, donc rien ne s'affiche.



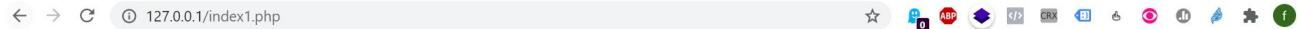
The screenshot shows a web browser window with the URL 127.0.0.1/index1.php. The page content is "hey !". Above the browser is a code editor window titled "index1.php" showing the following PHP code:

```
hey !
1 <?php
2
3
4
5     function maFonction(string $string, int $int){
6         for ($i=0; $i < $int ; $i++) {
7             return $string;
8         }
9     }
10    echo maFonction('hey ! <br>', 2);
11
12
13
14
15
16
```

Oui **MAIS.....** La particularité du “return”, c'est qu'il stoppe l'exécution du code. Rien ne survit après un “return”. C'est pour cela que mon string “hey !” ne s'affiche qu'une fois, car après le script s'arrête.

```
File Edit Selection Find View Goto Tools Project Preferences Help  
index1.php  
1 <?php  
2  
3  
4  
5     function maFonction(string $string, int $int){  
6         $display="";  
7         for ($i=0; $i < $int ; $i++) {  
8             $display.=$string;  
9         }  
10        return $display;  
11    }  
12  
13    echo maFonction('hey ! <br>', 2);  
14    $_temp = maFonction('belette <br>', 10);  
15    echo $_temp;  
16  
17  
18  
19
```

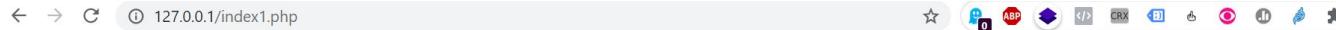
La solution est d'initialiser une variable “\$display” pour ensuite lui donner toutes les chaînes de caractères (en paramètre) et une fois qu'il a fini, on “return” cette valeur et donc on stoppe le script.

A screenshot of the Sublime Text editor. The title bar says "C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)". The code editor shows the following PHP script:

```
<?php
function maFonction(string $string, int $int){
    $display="";
    for ($i=0; $i < $int ; $i++) {
        $display.=$string;
    }
    return $display;
}
maFonction('hey ! <br>');
|
```

The line "maFonction('hey ! <br>');" is highlighted in red, indicating a syntax error.

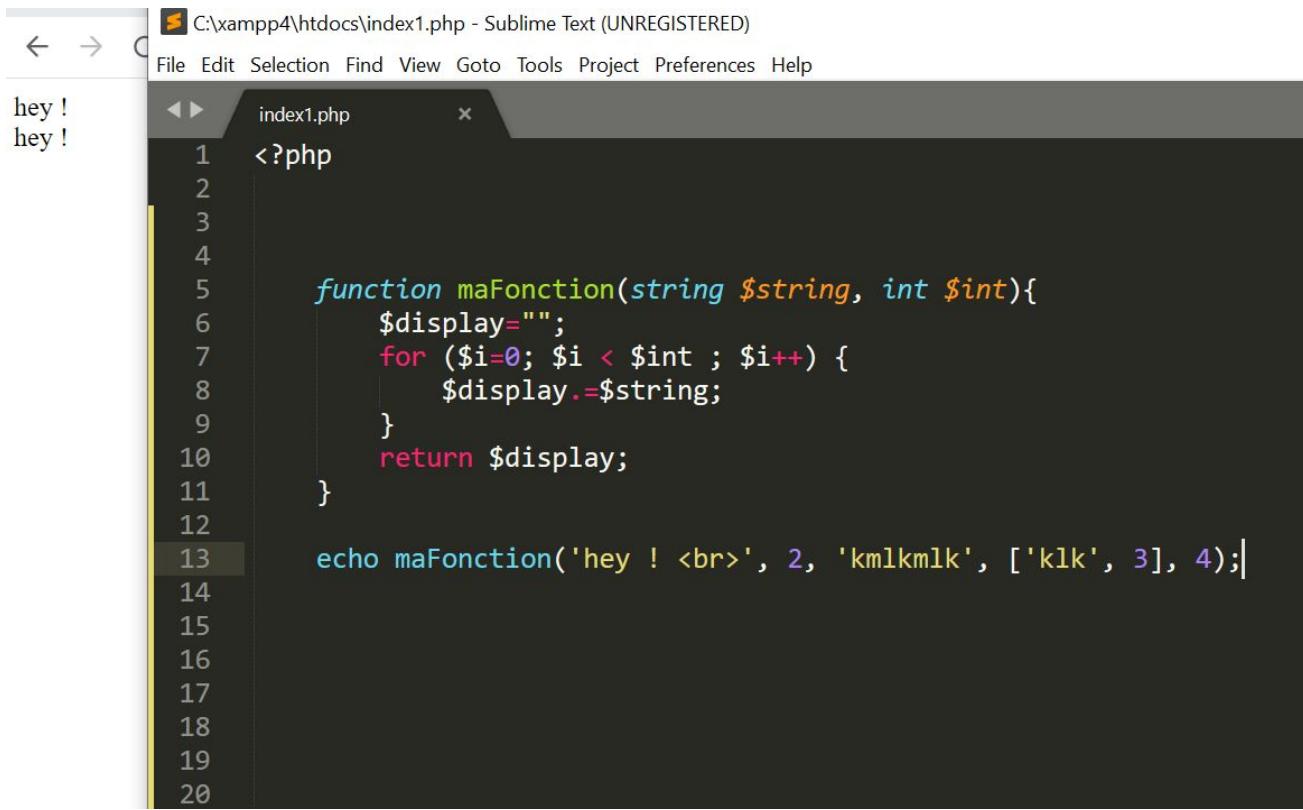
Si j'oublie un paramètre obligatoire, j'ai une erreur....

A screenshot of the Sublime Text editor. The title bar says 'C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)'. The menu bar includes 'File', 'Edit', 'Selection', 'Find', 'View', 'Goto', 'Tools', 'Project', 'Preferences', and 'Help'. The main pane shows the following PHP code:

```
index1.php
1 <?php
2
3
4
5     function maFonction(string $string, int $int){
6         $display="";
7         for ($i=0; $i < $int ; $i++) {
8             $display.= $string;
9         }
10        return $display;
11    }
12
13 echo maFonction('hey ! <br>', "klklklk");
14
15
16
17
18
```

The code contains a syntax error: 'maFonction' is defined with parameters '\$string' and '\$int', but the call at line 13 uses 'hey ! <br>' for the first parameter and 'klklklk' for the second, which is a string instead of an integer.

Si je ne respecte pas les types de mes paramètres, j'ai une erreur....



C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

hey !  
hey !

```
index1.php
```

1 <?php  
2  
3  
4  
5 function maFonction(string \$string, int \$int){  
6 \$display="";  
7 for (\$i=0; \$i < \$int ; \$i++) {  
8 \$display.=\$string;  
9 }  
10 return \$display;  
11 }  
12  
13 echo maFonction('hey ! <br>', 2, 'kmlkmlk', ['klk', 3], 4);  
14  
15  
16  
17  
18  
19  
20

Par contre, si j'ajoute plus de paramètres que prévu, ça n'a aucune incidence.



The screenshot shows a code editor window with the file "index1.php" open. The code defines a function "maFonction" that takes three parameters: a string, an integer, and a boolean "echo". If "echo" is true, it prints the result; if false, it returns it. The code then calls this function twice with different arguments.

```
hey !
hey !
Youpi !
Youpi !
Youpi !
Youpi !
hey ! <br>
Youpi ! <br>
```

```
index1.php
1 <?php
2
3
4
5     function maFonction(string $string, int $int, bool $echo = false){
6         $display="";
7         for ($i=0; $i < $int ; $i++) {
8             $display.=$string;
9         }
10
11         if($echo) echo $display;
12         else return $display;
13     }
14
15     maFonction('hey ! <br>', 2, true);
16
17     echo maFonction_('Youpi ! <br>', 4);
```

Ici, j'ai rajouté un troisième paramètre optionnel.

Si le troisième paramètre est à “true”, alors il s'affichera sans “echo”.

Si rien, se sera un return, donc je devais mettre un echo pour afficher la valeurs.

# A vous de jouer !

# Include & require

File Edit Selection Find View Goto Tools Project Preferences Help

```
<?php  
echo 'Je suis le Header avec le menu !';  
?>
```

```
<?php  
echo 'Je suis le Footer !';  
?>
```

```
<?php  
include('header.php');  
echo "<br><br>Je suis la page index !<br><br>";  
require('footer.php');
```

Les includes et requires insèrent du code dans une page.

Je suis le Header avec le menu !

Je suis la page index !

Je suis le Footer !

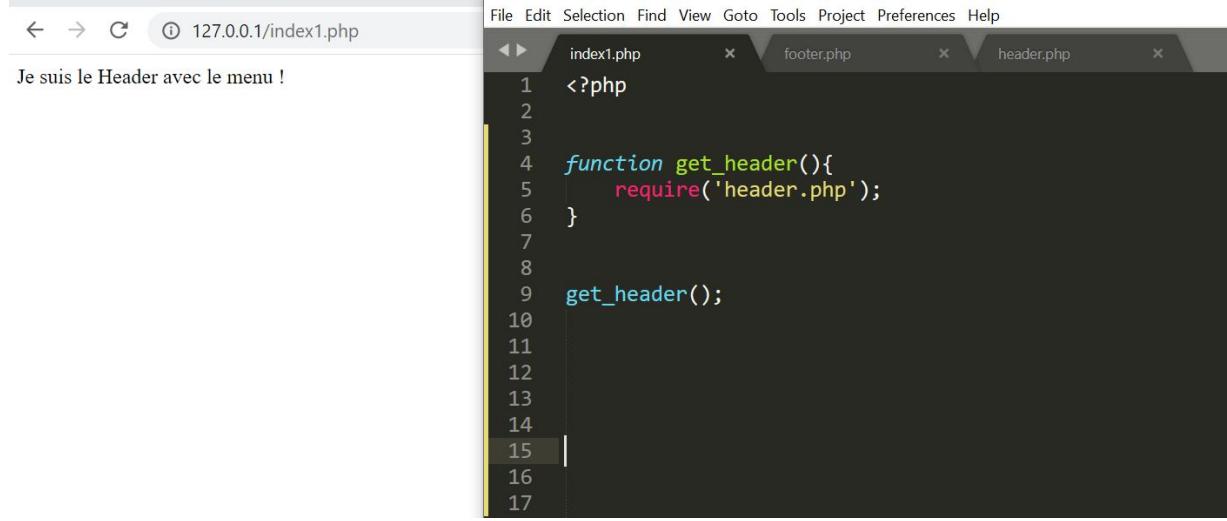
```
index1.php      x  footer.php      x  header.php      x
1  <?php
2
3
4  include('header.php');
5
6  echo "<br><br>Je suis la page index !<br><br>";
7
8  require('footer.php');
9
10
11
12
13
14
15
16
17
18  ?>
19
20
```

Donc, quand j'appelle le fichier index1.php, il m'affiche aussi le header.php et le footer.php car ils sont appelés dans la page.

```
index.php      x   footer.php      x   header.php      x
1 <?php
2
3
4 include('header.php');
5
6 echo "<br><br>Je suis la page index !<br><br>";
7
8 require('footer.php');
9
10
11
12
13
14
15
16 |
17
18 ?>
19
20
```

=

```
index.php      x   footer.php      x   header.php      x
1 <?php
2
3
4 echo 'Je suis le Header avec le menu !';
5
6 echo "<br><br>Je suis la page index !<br><br>";
7
8 echo "Je suis le Footer !";
9
10
11
12
13
14
15
16
17
18 ?>
19
20
```

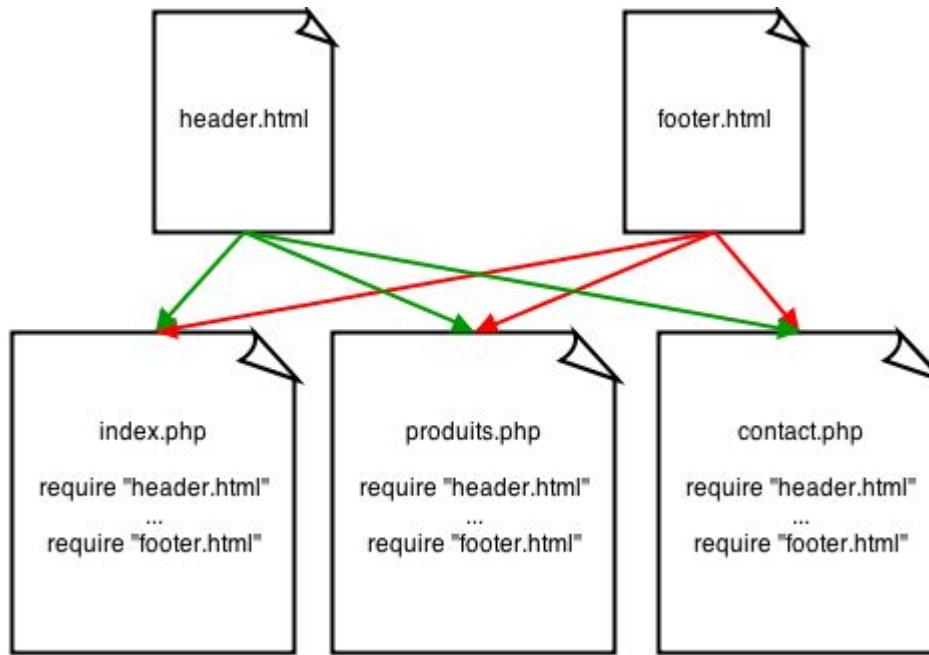


Je suis le Header avec le menu !

```
< ?php
function get_header(){
    require('header.php');
}
get_header();

```

D'ailleurs, si vous utilisez wordpress, il y a une fonction qui s'appelle `get_header()`, ce n'est ni plus, ni moins qu'une fonction qui inclus un `include` (ou `require`).



En fait, c'est l'équivalent de

```
<link rel="stylesheet" media="screen" href="lien_vers_mon_css.css"/>
```

pour le style

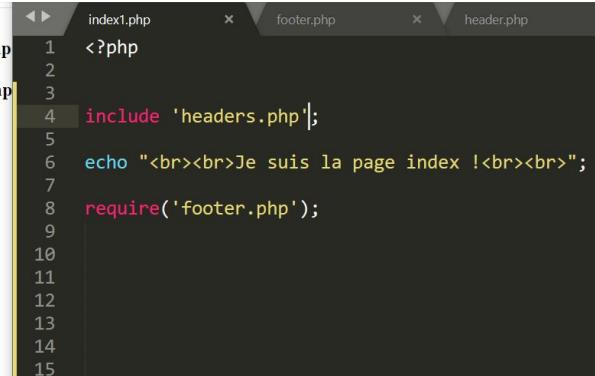
Donc, si vous avez 500 pages sur votre site et que vous avez besoin d'ajouter un item dans le menu, vous n'avez qu'à modifier 1 fichier et l'effet se fera sur tout le site.

**Warning:** include(headers.php): failed to open stream: No such file or directory in C:\xampp4\htdocs\index1.php

**Warning:** include(): Failed opening 'headers.php' for inclusion (include\_path='C:\xampp4\php\PEAR') in C:\xampp\

Je suis la page index !

Je suis le Footer !



```
index1.php          footer.php           header.php
1 <?php
2
3
4 include 'headers.php';
5
6 echo "<br><br>Je suis la page index !<br><br>";
7
8 require('footer.php');
9
10
11
12
13
14
15
```

Si j'appelle un fichier inexistant (headers.php au lieu de header.php), le include va m'afficher la page mais va m'afficher une erreur....

The screenshot shows a browser window at the top with the URL 127.0.0.1/index1.php. Below it is an IDE interface with a menu bar: File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, Help. There are three tabs open: index1.php, footer.php, and header.php. The index1.php tab contains the following code:

```
<?php  
require('headers.php');  
echo "<br><br>Je suis la page index !<br><br>";  
require('footer.php');
```

En revanche, si j'appelle un fichier inexistant avec un “require”, il va m'afficher une erreur mais il va aussi stopper l'affichage de la page.

Je suis le Header avec le menu !

Je suis la page index !

Je suis le Footer !



The screenshot shows a code editor with three tabs at the top: 'index1.php', 'footer.php', and 'header.php'. The 'index1.php' tab is active. The code in 'index1.php' is:

```
1 <?php
2
3
4 include_once('header.php');
5
6 echo "<br><br>Je suis la page index !<br><br>";
7
8 require_once('footer.php');
9
10
11
12
```

Vous pouvez aussi utiliser le “include\_once” et “require\_once”.  
Ils ont la particularité d’appeler qu’une seul fois un fichier.

Je suis le Header avec le menu ! Je suis le Header avec le menu !

Je suis la page index !

Je suis le Footer !



```
index.php          footer.php          header.php
1  <?php           x   x   x
2
3
4  include_once('header.php');    include('header.php');
5  include('header.php');        require_once('footer.php');
6
7  echo "<br><br>Je suis la page index !<br><br>";
8
9  require_once('footer.php');  require_once('footer.php');
10
11
12
13
14
15
16
17
```

En fait, le `*_once` va empêcher l'inclusion que si le fichier précédemment appelé a été avec une fonction `*_once`.

The screenshot shows a web browser window at 127.0.0.1/index1.php displaying the output of the PHP code. Below it is a code editor showing the source code of index1.php.

Output in browser:

```
Je suis le Header avec le menu ! Je suis le Header avec le menu !
Je suis la page index !
Je suis le Footer !
```

Code in editor:

```
index1.php
<?php
include_once('header.php');
include('header.php');

echo "<br><br>Je suis la page index !<br><br>";

include_once('footer.php');
require_once('footer.php');
```

Peu importe que le premier fichier soit appelé avec un require\_once ou un include\_once, il sera ignoré par l'autre fonction \*\_once car déjà inclus.

# **GET, POST, SESSION, COOKIE**

**`$_GET`**

Ces variables servent à passer des informations de page en page. Elles peuvent être ensuite enregistrées (ou non) dans la base de données.



MYDIGITALSCHOOL FORMATIONS MÉTIERS ÉCOLES PÉDAGOGIE ADMISSION

ESPACE ENTREPRISES

DOCUMENTATION

CANDIDATURE



PRÉSENTATION

VIE DE L'ÉCOLE

FORMATIONS

PÉDAGOGIE

ACTUALITÉS

AGENDA

STAGE / ALTERNANCE

ACCÈS / CONTACT

ACCUEIL / CAEN / PRÉSENTATION

## SE FORMER AUX MÉTIERS DU DIGITAL À MYDTGTTAI SCHOOL CAEN



DOCUMENTATION



CANDIDATURE



La requête GET est la plus utilisée. Le fait d'aller sur un site web est une requête GET.

The screenshot illustrates a development environment. On the left, a browser window shows the URL `127.0.0.1/index1.php?nom=jules`. The page content is "Bonjour jules, vous allez bien ?". On the right, a Sublime Text editor window titled "C:\xampp4\htdocs\index1.php - Sublime Text (UNREGISTERED)" displays the following PHP code:

```
<?php  
echo "Bonjour ".$_GET['nom'].", vous allez bien ?";  
?>
```

Il est possible de faire passer des paramètres dans l'url. On récupère les valeurs des paramètres avec la variable `$_GET['clé_de_la_variable']`

A screenshot showing a web browser window and a code editor window side-by-side.

The browser window shows the URL `127.0.0.1/index1.php?nom=Lopez&prenom=Ludivine`. The page content is "Bonjour Lopez Ludivine, vous allez bien ?".

The code editor window shows a file named `index1.php` with the following content:

```
<?php  
echo "Bonjour ".$_GET['nom']." ".$_GET['prenom'].", vous allez bien ?";  
?>
```

Il est possible de chaîner plusieurs paramètres pour passer plusieurs informations...  
`http://127.0.0.1/index1.php?nom=Lopez&prenom=Ludivine`

Array ( [nom] => Lopez [prenom] => Ludivine )

```
1 <?php
2     print_r($_GET);
3 ?>
4
5
6
7 |
```

On peut voir ici que \$\_GET est en fait un tableau...

sdcomm.com/?utm\_source=Ochato&utm\_medium=referral&utm\_campaign=jeuRestaurant

14350 St Martin des Besaces info@sdcomm.com 02 31 67 56 04

ZA Les Blanches Landes

**Horaires**

Lun - Jeu:  
8:00 - 12h / 13h30 - 17h30  
Ven:  
8:00 - 12h / 13h30 - 16h30

**ACCUEIL** NOS PRESTATIONS NOS MACHINES NOS CLIENTS EN PARLENT Contact

Nos catalogues

## NOUVEL INVESTISSEMENT !!!

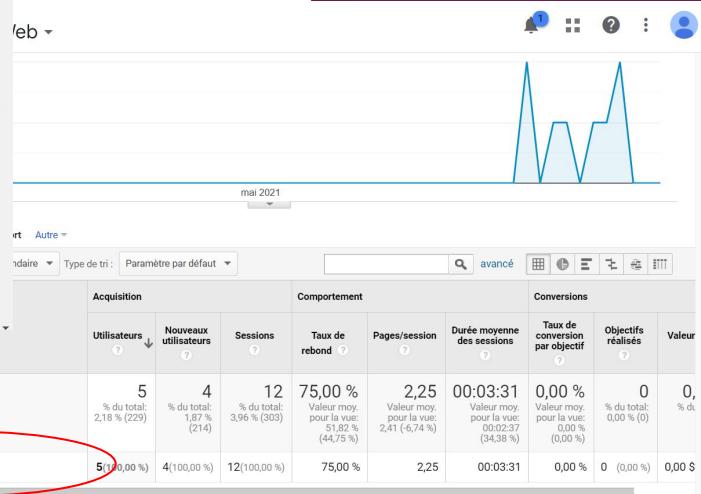
Cela fait maintenant plus d'un an que la situation sanitaire est compliquée pour tous.

Malgré ces difficultés, nous avons décidé de continuer

En attente de sdcomm.com... rendez-vous.

Search Console  
Réseaux sociaux  
**Campagnes**  
Toutes les campagnes  
Mots-clés achetés  
Attribution BETA

1. Ochato / referrer



Les paramètres des requêtes GET sont visibles de tous dans l'url. Il est préférable de les crypter (quand elles sont sensibles). En général, on utilise ce type de requête pour traquer un lien/utilisateur. Aussi utiliser dans les emails ou lors de lien de confirmation.

[https://sdcomm.com/?utm\\_source=Ochato&utm\\_medium=referral&utm\\_campaign=jeuRestaurant](https://sdcomm.com/?utm_source=Ochato&utm_medium=referral&utm_campaign=jeuRestaurant)

# thruuu

Please verify your thruuu account. Use the button below and click on it.

Verify your account

Thanks,  
The thruuu Team

© 2021 thruuu. All rights reserved.



<https://app.samuelschmitt.com/verify-user/eyJhbGciOiJIUzI1NilsInR5cCI6IkpXVCJ9eyJpZCI6IjYwYjU0MzY3ZWVmNTMxMDAwNDQ4Y2RjZiIsImIhdCI6MTYyMjQ5MjAwNywiZXhwIjoxNjlyNTM1MjA3fQ.7VvEMcNH7be2HiaVbBeijMS3R-tAWjQLgaND4sbRi-Q>

The screenshot shows a Microsoft Edge browser window with a purple header bar containing standard navigation icons (back, forward, search, etc.). The main content area has a light gray background. At the top left, there's a yellow icon of a person reading a book next to the text "LES LECTURES". Below it, a blue pencil icon is next to the word "CONTENT". A red arrow points downwards from this section towards the bottom of the page.

**Les types de contenus qui suscitent le plus d'engagement et de liens**

C'est toujours difficile de trouver un type de contenu à créer. D'autant plus, qu'il faut produire un contenu qui attire et suscite de l'engagement. Pour vous aider, cet article vous propose 7 types de contenus reconnus pour susciter de l'engagement.

**ADS**

**[TUTO] Apprenez à voir les publicités de vos concurrents !**

Le benchmark concurrentiel est très important pour se démarquer des concurrents ! Mais, il est parfois difficile de trouver toutes les informations que l'on veut. Pourtant, leurs publicités sont à votre disposition sur leurs réseaux sociaux et c'est une grande source pour vous. Avec cet article, vous saurez l'importance d'espionner les publicités de vos concurrents et comment faire !

**FACEBOOK**

**Économisez de l'argent grâce au test A/B de Facebook !**

Les publicités Facebook demandent un certain coût, du plus petit au plus considérable. Il faut être sûr de votre coup pour bien toucher votre cible. Avec l'A/B test, vous pouvez déterminer à

<https://deux.maillist-manage.com/click.zc?od=3z9ebdc8bc443a9654c2c238278c9a3b26f5330da49f8e5e72f1f4884190a0c96b&repDgs=16608a188c8e28ab&linkDgs=16608a188c8e1bab&mrd=16608a188c8e22a7&m=1>

# GET, POST, SESSION, COOKIE

`$_POST`

nom
prenom
Envoyer

```
<?php
    if($_POST){

        print_r($_POST);
    }
?>

<form action="" method="POST">
    <input type="text" name="nom" placeholder="nom">
    <br>
    <input type="text" name="prenom" placeholder="prenom">
    <br>
    <input type="submit" value="Envoyer">
</form>
```

`$_POST` est un tableau aussi... Cette méthode est très souvent utilisée pour un formulaire..

nom
prenom
<input type="button" value="Envoyer"/>

```
1  <?php
2
3  if($_SERVER['REQUEST_METHOD'] == "POST"):
4      echo "Nom de l'utilisateur : ".$_POST['nom']."<br>";
5      echo "Prenom de l'utilisateur : ".$_POST['prenom'];
6
7  endif;
8
9
10 ?>
11
12 <form action="" method="POST">
13     <input type="text" name="nom" placeholder="nom">
14     <br>
15     <input type="text" name="prenom" placeholder="prenom">
16     <br>
17     <input type="submit" value="Envoyer">
18 </form>
19
```

Pour sélectionner une valeur particulière, on procède comme `$_GET` (comme un tableau)

`$_POST['clé_de_la_variable']`

The screenshot shows a web development interface with two tabs: "index1.php" and "post.php".

**index1.php:**

```
<form action="post.php" method="POST">
    <input type="text" name="nom" placeholder="nom">
    <br>
    <input type="text" name="prenom" placeholder="prenom">
    <br>
    <p>Bleu : <input type="radio" name="couleur" value="bleu"> Rouge : <input type="radio" name="couleur" value="rouge"></p>
    <br>
    <input type="submit" value="Envoyer">
</form>
```

**post.php:**

```
1  <?php
2
3  if($_SERVER['REQUEST_METHOD'] == "POST" && $_SERVER['HTTP_REFERER'] == "http://127.0.0.1/index1.php"):
4      echo "Nom de l'utilisateur : ".$_POST['nom']."<br>";
5      echo "Prenom de l'utilisateur : ".$_POST['prenom']."<br>";
6      echo "Couleur de l'utilisateur : ".$_POST['couleur'];
7
8  endif;
9
10
11 ?>
```

A red arrow points from the "couleur" radio button in the index1.php code to the "couleur" variable in the post.php code.

Exemple d'un formulaire qui envoie les données sur une autre page que la sienne  
(post.php)

nom
prenom

Envoyer

```
1
2
3    <form action="post.php" method="POST">
4        <input type="text" name="nom" placeholder="nom">
5        <br>
6        <input type="text" name="prenom" placeholder="prenom">
7        <br>
8        <input type="hidden" name="secret" value="sdksldkslmdksmldksd">
9        <br>
10       <input type="submit" value="Envoyer">
11   </form>
12
```

The screenshot shows a browser developer tools window with the Network tab selected. A single request to 'post.php' is listed, showing the following details:

- Name:** post.php
- Headers:** Referer: http://127.0.0.1/index1.php, sec-ch-ua: " Not A Brand";v="99", "Chromium";v="99", "Google Chrome";v="99", sec-ch-ua-mobile: ?0, Sec-Fetch-Dest: document, Sec-Fetch-Mode: navigate, Sec-Fetch-Site: same-origin, Sec-Fetch-User: ?1, Upgrade-Insecure-Requests: 1, User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/1537.36
- Form Data:** nom: kjkj, prenom: kjkj, secret: sdksldkslmdksmldksd

Les informations d'un formulaire sont généralement connues (car c'est l'utilisateur qui les tape...), cependant, il est possible de faire passer certaines informations en cachées (hidden). Mais même si ces infos sont plus discrètes qu'avec les requêtes GET, elles restent facile à trouver.

# **GET, POST, SESSION, COOKIE**

**`$_COOKIE`**

(PHP 4, PHP 5, PHP 7, PHP 8)

setcookie — Envoie un cookie

## Description

---

```
setcookie ( string $name , string $value = "" , int $expires = 0 , string $path = "" , string  
$domain = "" , bool $secure = false , bool $httponly = false ) : bool
```

Signature alternative disponible à partir de PHP 7.3.0 :

```
setcookie ( string $name , string $value = "" , array $options = [] ) : bool
```

**setcookie()** définit un cookie qui sera envoyé avec le reste des en-têtes HTTP. Comme pour les autres en-têtes, les cookies doivent être envoyés *avant* toute autre sortie (c'est une restriction du protocole HTTP, pas de PHP). Cela vous impose d'appeler cette fonction avant toute balise <html> ou <head> et aussi des caractères d'espacement blanc.

Comme en javascript, php peut stocker des valeurs persistantes dans le navigateur pour les récupérer sur n'importe quelle page.



index1.php

```

1 <?php
2
3 setcookie('user', 'nicolas', time() + 60 * 60 * 24);
4
5

```

Elements Console Sources Network Performance Memory Application Lighthouse Adblock Plus

Application

Name	Value	Domain	P...	Expires / Max-Age	Size	Htt...	Sec...	Sam...	Sa...	Prio...
user	nicolas	127.0.0.1	/	2021-06-02T09:58:...	11					Me...

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

<http://127.0.0.1>

Trust Tokens

Cache

- Cache Storage
- Application Cache

Background Services

- Background Fetch
- Background Sync
- Notifications
- Payment Handler
- Periodic Background Sync
- Push Messaging

Console What's New

Select a cookie to preview its value

Network

Filter Preserve log Disable cache No throttling

Blocked Requests

Request URL: http://127.0.0.1/index1.php  
 Request Method: GET  
 Status Code: 200 OK  
 Remote Address: 127.0.0.1:80  
 Referrer Policy: strict-origin-when-cross-origin

Response Headers

Connection: Keep-Alive  
 Content-Length: 0  
 Content-Type: text/html; charset=UTF-8  
 Date: Tue, 01 Jun 2021 09:58:10 GMT  
 Keep-Alive: timeout=5, max=100  
 Server: Apache/2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.11  
 Set-Cookie: user=nicolas; expires=Wed, 02-Jun-2021 09:58:10 GMT; Max-Age=86400  
 X-Powered-By: PHP/7.4.11

7 requests

Console What's New

Highlights from the Chrome 90 update

Les 3 paramètres obligatoires sont `setCookie('clé', 'valeur', 'temps_durée_cookie');`  
 Par défaut, si 'temps\_durée\_cookie' est égal à 0, le cookie durera jusqu'à ce que l'utilisateur ferme son navigateur.

Array ( [user] => nicolas )

```
1 <?php
2
3 //setcookie('user', 'nicolas', time() + 60 * 60 * 24);
4 print_r($_COOKIE);
5
6
7 ?>
```

Pour récupérer un cookie, il faut faire `$_COOKIE`. Et comme GET et POST, on voit que `COOKIE` est un tableau...

The screenshot shows the Chrome DevTools Application tab. On the left, a sidebar lists categories: Application, Storage, Cache, and Background Services. Under Application, 'Manifest', 'Service Workers', and 'Storage' are listed. Under Storage, 'Local Storage', 'Session Storage', 'IndexedDB', and 'Web SQL' are listed, with 'Cookies' expanded. Under Cookies, a list of cookies for the domain 'http://127.0.0.1' is shown. One cookie, 'user' with value 'nicolas', has a size of 11 and an expiration date of 2021-06-02T09:58:... . A red circle highlights the 'Size' column for this cookie. Below the table, a message says 'Select a cookie to preview its value'.

Name	Value	Domain	P...	Expires / Max-Age	Size	Htt...	Sec...	Sam...	Sa...	Prio...
user	nicolas	127.0.0.1	/	2021-06-02T09:58:...	11					Me...

Il faut noter qu'un cookie est très léger. Il peut aller jusqu'à (environ) 5Ko. Cela dépend des navigateurs.

```
index1.php x
1 <?php
2
3 //setcookie('user', 'nicolas', time() + 60 * 60 * 24);
4 print_r($_COOKIE['user']);
5
6
7
8 ?>
```

← → ⌂ 127.0.0.1/index1.php



nicolas

Et donc pour récupérer un cookie en particulier....

The screenshot shows the browser's developer tools open to the Application tab. Under the Storage section, the Cookies category is selected, showing a list of cookies. One cookie for 'http://127.0.0.1' is highlighted. Below the table, a message reads 'Select a cookie to preview its value'.

```
<?php  
unset($_COOKIE['user']);  
setcookie('user', 'nicolas', time() - 10);  
if(isset($_COOKIE['user'])) echo $_COOKIE['user'];  
?  
?>
```

Pour supprimer un COOKIE, il faut faire 2 choses :

- Détruire la clé que l'on souhaite détruire (comme dans tout tableau) avec  
`unset($_COOKIE['user'])`
- Enregistrer ce même cookie dans le passé pour “tuer” sa persistance.

**Warning:** setcookie() expects parameter 2 to be string, array given in **C:\xampp4\htdocs\index1.php** on line **9**

```
1  <?php
2
3  $user = [
4      "nom" => "Julie",
5      "prenom" => "Garcia",
6      "age" => 29
7 ];
8
9  setcookie('user', $user, time() + 60 * 60 * 24);
10
11
12
13
14 ?>
```

On ne peut pas stocker de tableau dans un cookie, seulement un string (chaîne de caractère)

```
string(69) "a:3:{s:3:"nom";s:5:"Julie";s:6:"prenom";s:6:"Garcia";s:3:"age";i:29;}"
Array ( [nom] => Julie [prenom] => Garcia [age] => 29 )
```

```
1 <?php
2
3 $user = [
4     "nom" => "Julie",
5     "prenom" => "Garcia",
6     "age" => 29
7 ];
8
9 setcookie('user', serialize($user), time() + 60 * 60 * 24);
10
11 var_dump($_COOKIE['user']);
12 echo "<br>";
13 print_r(unserialize($_COOKIE['user']));
14
15
16
17 ?>
```

Il y a une fonction native de PHP qui est unserialize() qui transforme les tableaux en chaîne de caractère (string). Pour pouvoir lire ce tableau, il ya la fonction (native aussi) de PHP qui est unserialize() qui change un string en array.



Par contre, si vous voulez protéger votre site avec COOKIE, sachez que ce n'est pas vraiment sécurisé. Imaginons que vous ayez un site avec du contenu réservé aux adultes. Vous demandez l'age de l'internaute au début.

The screenshot shows the Chrome DevTools interface with the Application tab selected. In the left sidebar, under Storage, the Cookies section is expanded, showing a list of cookies for the domain `http://127.0.0.1`. One cookie is listed: `age` with a value of `13`. The cookie table has columns for Name, Value, Domain, Path, Expires / Max-Age, Session, HTTP Only, SameSite, and Persistence. The cookie `age` is listed in the first row of the table.

Il peut entrer dans les paramètres et modifier le cookie pour accéder au contenu protégé.  
(Le plus simple c'est de mentir sur son âge mais c'est un exemple)

# **GET, POST, SESSION, COOKIE**

**`$_SESSION`**



```
index1.php      x  post.php      x
1 <?php
2 session_start();
3 $_SESSION['nom'] = "billy";
```

Une session, c'est comme un cookie mais en secret, pas modifiable par l'utilisateur et sans limite de stockage....

Pour initialiser une session, il faut forcément commencer par la fonction session\_start();  
Ensuite, c'est comme les autres, c'est un tableau.....

The screenshot shows a browser window with developer tools open. The address bar says 127.0.0.1/index1.php. The code editor on the left has index1.php open, containing the following PHP code:

```
<?php  
session_start();  
var_dump($_SESSION);
```

The developer tools have the "Application" tab selected. In the "Storage" section, under "Cookies", there is one entry for the domain http://127.0.0.1. A table below shows the cookie details:

Name	Value	Domain	P..	Expires / Max...	Size	Ht...	Se...	Sa...	Pr...
PHPSESSID	vn5vm5i9q...	127.0....	/	Session	35				M...

A tooltip at the bottom right of the table says "Select a cookie to preview its value".

On peut voir que PHP a créé un cookie avec un identifiant aléatoire.  
Cet identifiant ne contient pas d'information. Elles sont stockées côté serveur.  
Ce cookie met juste en relation le navigateur avec les infos stockées côté serveur.

```
array(2) { ["nom"]=> string(5) "billy" ["usr"]=> array(2) { ["nom"]=> string(4) "john" ["prenom"]=> string(5) "neige" } }
```

```
index1.php      post.php
1  <?php
2  session_start();
3  $usr = [
4      "nom" => "john",
5      "prenom" => "neige"
6  ];
7
8  $_SESSION['usr'] = $usr;
9
10 var_dump($_SESSION);
11
12
```

Voici un exemple qui prouve que l'on peut facilement stocker des tableaux en session



127.0.0.1/index1.php

array(1) { ["nom"]=&gt; string(5) "billy" }

```
index1.php x po
1 <?php
2 session_start();
3
4 unset($_SESSION['usr']);
5
6 var_dump($_SESSION);
7
8 |
```

Un simple `unset()` suffit à détruire une clé de session.