



PHP

FONCTIONS / INCLUSIONS



Aurélien Delorme
OCT 2022



Pourquoi utiliser des fonctions

Fonctions natives sur les tableaux

Fonction natives sur les variables

Fonctions natives sur les dates

Création de fonction php personnalisées

Include / Require / Require once



Pourquoi utiliser des fonctions

Permet de réutiliser son code source

Gain en terme de temps de développement

Gain en terme de maintenance

Prend une ou
plusieurs données
en entrée



Peut retourner une
donnée en sortie ou
simplement exécuter
une action

Effectue un traitement sur ses données

Une fonction se veut REUTILISABLE



Fonctions natives sur les tableaux

```
$identity = ['nom' => 'Delorme', 'prenom' => 'Aurelien', 'age' => 28, 'dateNaissance' => date_create('30-03-1993')->format('d/m/Y')];
```

array_key_exists

Vérifie qu'une clé est bien présente dans un tableau

```
var_dump(array_key_exists('adresse', $identity));
```

```
/var/www/html/index.php:3:boolean false
```

```
var_dump(array_key_exists('nom', $identity));
```

```
/var/www/html/index.php:3:boolean true
```

count

Retourne le nombre d'éléments d'un tableau

```
var_dump(count($identity));
```

```
/var/www/html/index.php:3:int 4
```

in_array

Vérifie qu'une valeur est présente dans un tableau

```
var_dump(in_array('nom', $identity));
```

```
/var/www/html/index.php:3:boolean false
```

```
var_dump(in_array('Aurelien', $identity));
```

```
/var/www/html/index.php:3:boolean true
```



Fonctions natives sur les tableaux

```
$listeCourse = ['pomme', 'poire', 'chocolat', 'poire', 'pomme'];
```

array_search

Recherche dans un tableau la clé associé à la première valeur

```
var_dump(array_search('pomme', $listeCourse));
```

← → ↻ ⓘ localhost

/var/www/html/index.php:3:int 0

```
var_dump(array_search('poire', $listeCourse));
```

← → ↻ ⓘ localhost

/var/www/html/index.php:3:int 1

array_unique

Retourne un tableau en supprimant les doublons

```
var_dump(array_unique($listeCourse));
```

```
/var/www/html/php-backpack/index.php:5:
array (size=3)
  0 => string 'pomme' (length=5)
  1 => string 'poire' (length=5)
  2 => string 'chocolat' (length=8)
```

array_push

Permet d'ajouter un élément à notre tableau

```
array_push($listeCourse,
'Nutella');
// Equivalent de :
$listeCourse[] = 'Nutella'
```

```
/var/www/html/php-backpack/index.php:8:
array (size=6)
  0 => string 'pomme' (length=5)
  1 => string 'poire' (length=5)
  2 => string 'chocolat' (length=8)
  3 => string 'poire' (length=5)
  4 => string 'pomme' (length=5)
  5 => string 'Nutella' (length=7)
```

<https://www.php.net/manual/fr/ref.array.php> ... Il en existe pleins d'autres !

Fonctions sur les variables

gettype

Recherche dans un tableau la clé associé à la première valeur

```
<?php
$variable = 'Aurélien';
var_dump(gettype($variable));
?>
```

← → ↻ ⓘ localhost

/var/www/html/index.php:3:string 'string' (length=6)

```
<?php
$variable = true;
var_dump(gettype($variable));
?>
```

← → ↻ ⓘ localhost

/var/www/html/index.php:3:string 'boolean' (length=7)

unset

Détruit une variable

```
$variable = true;
unset($variable);
var_dump($variable);
```

(!) Warning: Undefined variable \$variable in /var/www/html/php-backpack/index.php on line 15

Call Stack				
#	Time	Memory	Function	Location
1	0.0019	419968	{main}()	.../index.php:0

/var/www/html/php-backpack/index.php:15:null

is_null

Retourne la nullité de la variable

```
<?php
$variable = 'toto';
var_dump(is_null($variable));
$variable = null;
var_dump(is_null($variable));
?>
```

← → ↻ ⓘ localhost

/var/www/html/index.php:3:boolean false

/var/www/html/index.php:5:boolean true

Fonctions sur les variables

empty

Indique si une variable est vide ou pas (")

```
<?php
$variable = 'aurelien';
var_dump(empty($variable));
?>
```

```
← → ↻ ⓘ localhost
/var/www/html/index.php:3:boolean false
```

```
<?php
$variable = "";
var_dump(empty($variable));
?>
```

```
← → ↻ ⓘ localhost
/var/www/html/index.php:3:boolean true
```

is_string

Indique si une variable est de type string

```
<?php
$variable = 10;
var_dump(is_string($variable));
?>
```

```
← → ↻ ⓘ localhost
/var/www/html/index.php:3:boolean false
```

```
<?php
$variable = 'Aurélien';
var_dump(is_string($variable));
?>
```

```
← → ↻ ⓘ localhost
/var/www/html/index.php:3:boolean true
```

isset

Indique si une variable existe

```
<?php
$variable = 'Aurélien';
var_dump(isset($variable));
unset($variable);
var_dump(isset($variable));
?>
```

```
← → ↻ ⓘ localhost
/var/www/html/index.php:3:boolean true
/var/www/html/index.php:5:boolean false
```

<https://www.php.net/manual/fr/ref.var.php> et pleins d'autres !!

Fonctions sur les dates

date(format)

Retourne la date du jour sous forme de chaîne de caractère. Prend en paramètre le format de retour souhaité

```
<?php
$date = date('d/m/Y');
echo('Nous sommes le '. $date);
?>
```

← → ↻ ⓘ localhost

Nous sommes le 20/02/2020

date(format, timestamp)

Retourne la date sous forme de chaîne de caractère. Prend en second paramètre le timestamp recherché

```
<?php
$date = date('d/m/Y', 733486810);
echo('Nous sommes le '. $date);
?>
```

← → ↻ ⓘ localhost

Nous sommes le 30/03/1993

date_create

Permet de créer une date depuis le format suivant :

yyyy-mm-dd

Equivalent à new DateTime()

```
<?php
$date1 = date_create("2013-03-15");
$sameDate = new DateTime('2013-03-15');
var_dump($sameDate);
?>
```

← → ↻ ⓘ localhost

```
/var/www/html/index.php:4:
object(DateTime)[1]
  public 'date' => string '2013-03-15 00:00:00.000000' (length=26)
  public 'timezone_type' => int 3
  public 'timezone' => string 'UTC' (length=3)

/var/www/html/index.php:5:
object(DateTime)[2]
  public 'date' => string '2013-03-15 00:00:00.000000' (length=26)
  public 'timezone_type' => int 3
  public 'timezone' => string 'UTC' (length=3)
```


Fonctions sur les dates

date(format)

```
<?php
$today = date_create("now");
echo('Aujourd'hui, nous sommes le : ' . $today->format('d/m/Y').<br>');
echo('Demain, nous serons le : ' . date_modify($today,'1 day')->format('d/m/Y'));
?>
```

← → ↻ ⓘ localhost

Aujourd'hui, nous sommes le : 20/02/2020
Demain, nous serons le : 21/02/2020

date_diff

Permet de calculer la différence entre 2 dates

```
<?php
$birthDate = date_create("1993-03-30");
$today = date_create("now");
$diff = date_diff($birthDate, $today);
var_dump($diff);
echo("Je suis né il y a $diff->y ans, $diff->m mois et $diff->d soit $diff->days jours");
?>
```

← → ↻ ⓘ localhost

```
/var/www/html/index.php:5:
object(DateInterval)[3]
  public 'y' => int 26
  public 'm' => int 10
  public 'd' => int 21
  public 'h' => int 10
  public 'i' => int 49
  public 's' => int 51
  public 'f' => float 0.009391
  public 'weekday' => int 0
  public 'weekday_behavior' => int 0
  public 'first_last_day_of' => int 0
  public 'invert' => int 0
  public 'days' => int 9823
  public 'special_type' => int 0
  public 'special_amount' => int 0
  public 'have_weekday_relative' => int 0
  public 'have_special_relative' => int 0
```

Je suis né il y a 26 ans, 10 mois et 21 soit 9823 jours

time

Affiche le timestamp actuel (nombre de secondes écoulées depuis le 1 Janvier 1970 correspondant a la date de début d'un timestamp UNIX)

```
<?php
echo("Le 1 Janvier 1970 à eu lieu il y a 'time().' secondes");
?>
```

← → ↻ ⓘ localhost

Le 1 Janvier 1970 à eu lieu il y a 1582196365 secondes

<https://www.php.net/manual/fr/ref.datetime.php> et pleins d'autres ...



Création de fonctions

- 1) utilisation du mot clé `function nomDeLaFonction()`.
- 2) Une fonction peut prendre 0 ou x paramètres `function test($param1, $param2, ...)`
- 3) Si cette fonction doit retourner une variable, ne pas oublier le mot clé `return` à la fin de la fonction

Exemple 1. Une fonction qui ne retourne rien mais qui dit bonjour !

```
<?php
function sayHello($nom)
{
    echo 'Hello ' . $nom . ' !<br />';
}
sayHello('Aurélien');
sayHello('Everybody');
```



Hello Aurélien !
Hello Everybody !



Création de fonctions

Exemple 2. Une fonction qui jouera le rôle d'une calculette.
Elle prendra en entrée 3 paramètres et retournera un résultat !

```
<?php
function calculette($operator, $nombre1, $nombre2)
{
    switch ($operator) {
        case '+':
            return $nombre1 + $nombre2;
            break;
        case '-':
            return $nombre1 - $nombre2;
            break;
        case '*':
            return $nombre1 * $nombre2;
            break;
        case '/':
            return $nombre1 / $nombre2;
            break;
    }
}

var_dump(calculette('+', 3, 3));
var_dump(calculette('*', 10, 12));
var_dump(calculette('/', 23, 4));
?>
```

 localhost

```
/var/www/html/index.php:19:int 6
```

```
/var/www/html/index.php:20:int 120
```

```
/var/www/html/index.php:21:float 5.75
```



Include / Include once

Permet d'inclure un fichier PHP dans un autre. Très utilisé pour les raisons suivantes :

- Permet de développer un élément et de le réutiliser ailleurs
- Permet ainsi de gagner en temps de développement
- Permet de gagner en terme de maintenabilité.

Exemple : Mon site internet possède 20 pages différentes.

Sur chacune de ses pages, j'ai :

- Un pied de page qui est systématiquement le même
- Un menu qui sera systématiquement le même
- ...

Je vais donc pouvoir créer 2 fichiers (menu.php et footer.php). Ces derniers seront ajoutés sur toutes mes pages grâce à l'instruction Include !

Quand je souhaiterais modifier mon menu ou mon pied de page, j'aurais juste à modifier le fichier menu.php ou le fichier footer.php






Include

Menu

Contenu

Footer

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="utf-8" />
  <title>Un site pratique à utiliser</title>
</head>
<body>
  <?php include("block/menu.php"); ?>
  <div id="corps">
    <h1>Le contenu spécifique de ma page</h1>
    <p>
      C'est bien mieux de réutiliser ce que l'on a déjà fait avant !
    </p>
  </div>
  <?php include("block/footer.php"); ?>
</body>
</html>
```

▼  block footer.php menu.php index.php



Include

block > menu.php >

```
<nav id="menu">
  <div class="element_menu">
    <h3>Ici mon menu</h3>
    <ul>
      <li><a href="page1.html">Mes restos préférés</a></li>
      <li><a href="page2.html">Les restos à découvrir</a></li>
      <li><a href="page3.html">Scanner un code !</a></li>
    </ul>
  </div>
</nav>
```

block > footer.php >

```
<nav id="footer">
  <div class="element_menu">
    <h3>Ici mon footer</h3>
    <ul>
      <li><a href="page1.html">Lien</a></li>
      <li><a href="page2.html">Lien</a></li>
      <li><a href="page3.html">Lien</a></li>
    </ul>
  </div>
</nav>
```

Ici mon menu

- [Mes restos préférés](#)
- [Les restos à découvrir](#)
- [Scanner un code !](#)

Le contenu spécifique de ma page

C'est bien mieux de réutiliser ce que l'on a déjà fait avant !

Ici mon menu

- [Lien](#)
- [Lien](#)
- [Lien](#)



include / include_once

Ici mon menu

- [Mes restos préférés](#)
- [Les restos à découvrir](#)
- [Scanner un code !](#)

Ici mon menu

- [Mes restos préférés](#)
- [Les restos à découvrir](#)
- [Scanner un code !](#)

Le contenu spécifique de ma page

C'est bien mieux de réutiliser ce que l'on a déjà fait avant !

```
<body>
<?php
    include("block/menu.php");
    include("block/menu.php")
?>
<div id="corps">
    <h1>Le contenu spécifique de ma page</h1>
    <p>
        C'est bien mieux de réutiliser ce que l'on a déjà fait avant !
    </p>
</div>

<?php include("block/footer.php"); ?>
```

Ici mon menu

- [Mes restos préférés](#)
- [Les restos à découvrir](#)
- [Scanner un code !](#)

Le contenu spécifique de ma page

C'est bien mieux de réutiliser ce que l'on a déjà fait avant !

```
<body>
<?php
    include_once("block/menu.php");
    include_once("block/menu.php")
?>
<div id="corps">
    <h1>Le contenu spécifique de ma page</h1>
    <p>
        C'est bien mieux de réutiliser ce que l'on a déjà fait avant !
    </p>
</div>
```




require / require_once

Similaire à include. En revanche, si le fichier à inclure n’est pas trouvé, une erreur sera lancée. La ou include se contente d’un warning et affichera le reste de la page, require lancera une erreur et stoppera le script.

Avec include et un fichier inexistant

```
<?php include("block/unfichierinexistant.php"); ?>
```

 Warning: include(block/unfichierinexistant.php): failed to open stream: No such file or directory in /var/www/html/index.php on line 10				
Call Stack				
#	Time	Memory	Function	Location
1	0.0052	359528	{main}()	../index.php:0

 Warning: include(): Failed opening 'block/unfichierinexistant.php' for inclusion (include_path='.:usr/local/lib/php') in /var/www/html/index.php on line 10				
Call Stack				
#	Time	Memory	Function	Location
1	0.0052	359528	{main}()	../index.php:0

LE TEMPS C'EST DE L'ARGENT

C'est bien mieux de réutiliser ce que l'on a déjà fait avant !


Ici mon footer !

- [Mentions légales](#)
- [A propos](#)
- [Me suivre sur Facebook !](#)

Avec require et un fichier inexistant

```
<?php require("block/unfichierinexistant.php"); ?>
```

 Warning: require(block/unfichierinexistant.php): failed to open stream: No such file or directory in /var/www/html/index.php on line 10				
Call Stack				
#	Time	Memory	Function	Location
1	0.0044	359512	{main}()	../index.php:0

 Fatal error: require(): Failed opening required 'block/unfichierinexistant.php' (include_path='.:usr/local/lib/php') in /var/www/html/index.php on line 10				
Call Stack				
#	Time	Memory	Function	Location
1	0.0044	359512	{main}()	../index.php:0

Comme include_once, require_once chargera le fichier seulement si ce dernier n’a pas été déjà chargé.



Echanges / Questions

