

PHP

Introduction

PHP

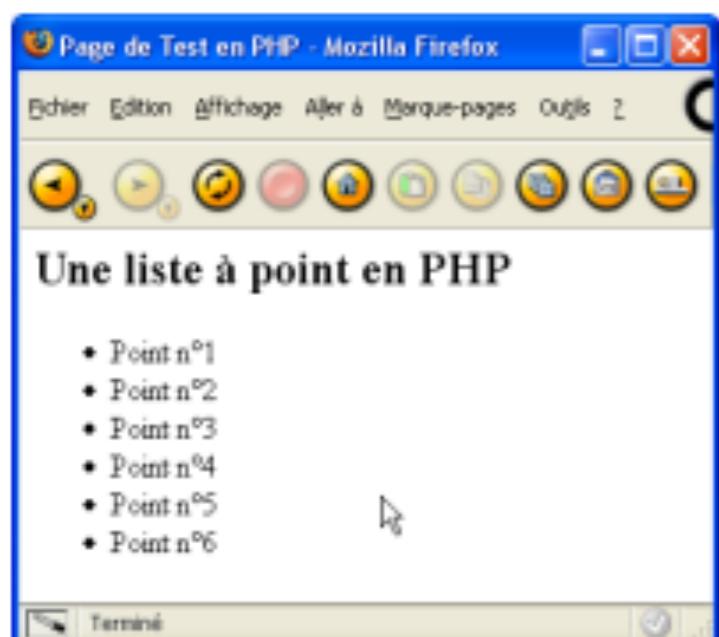
- Langage interprété
- Exécuté sur un serveur web
- Faiblement typé
- Particulièrement adapté au développement web

Requête

 http://www.nom_de_domaine.fr/fichier.php



Poste Client



Affichage du navigateur

Serveur Web



Réponse

Interprétation du fichier PHP

```
<html>
<head>
    <title>Page de Test en PHP</title>
</head>
<body>
    <h2>Une liste à graver: point en PHP</h2>
    <ul>
        <?
            for ($x=1; $x<7; $x++)
            {
                echo "<li>Point n°". $x. "</li>";
            }
        ?>
    </ul>
</body>
</html>
```

```
<html>
<head>
    <title>Page de Test en PHP</title>
</head>
<body>
    <h2>Une liste à graver: point en PHP</h2>
    <ul>
        <li>Point n°1</li>
        <li>Point n°2</li>
        <li>Point n°3</li>
        <li>Point n°4</li>
        <li>Point n°5</li>
        <li>Point n°6</li>
    </ul>
</body>
</html>
```

Génération
d'une page HTML

Principe d'une requête en PHP

IL faut donc un serveur web

Les options d'exécutions

- Logiciels tout en un (XAMPP...)
- Serveur interne (depuis php 5.5)
- Installation manuelle
- Virtualisation (Vagrant, Docker)
- Serveur distant

Premiers pas

Première page

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <title>Première page PHP</title>
</head>
<body>
    <h1>
        Bonjour nous sommes le
        <?php
            echo date( 'd/m/Y' );
        ?>
    </h1>
</body>
</html>
```



Remarques

- Le code php est encadré entre balises <?php et ?>
- Les instructions se terminent par un point-virgule
- Le code php est mélangé avec le code html

Les variables

- Commence par \$
- Pas de déclaration de type pour les variables scalaires
- Typage dynamique

Exemple

```
$nom = 'MORIN';
$prenom = "Isabelle";
$age = 25;
$prix = 5.3;
$distance = 5e16;
$test = true;
```

Commentaires

```
//commentaire monoligne  
  
/*  
 * commentaire  
 * multiligne  
 */  
  
$test = 'b'; //commentaire de fin de ligne
```

La concaténation

```
echo $nom . ' ' . $prenom. '<br>';

echo "$prenom $nom <br>";

printf(
    "Bonjour %s %s vous avez %d ans<br>",
    $prenom, $nom, $age
);
```

La concaténation

```
$fruit = "poire";  
  
echo "J'ai acheté des {$fruit}s";
```

La concaténation

```
//Concaténation d'une clef de tableau  
$person = ["name"=>"Jean", "age"=>34];  
  
$hello = "Hello {$person['name']}";  
  
echo $hello;
```

Chaîne multi-ligne

```
$lang = "PHP";  
  
$phrase = <<<EOD  
Voici une chaîne de caractère multi-ligne  
utilisant le principe d'interpolation  
de $lang  
EOD;  
  
echo $phrase;
```

Infos de debug

```
$nom = 'MORIN';
$prix = 15.5;
$qt = 3;
$majeur = true;

var_dump($nom);
var_dump($prix);
var_dump($qt);
var_dump($majeur);
```

```
string 'MORIN' (length=5)

float 15.5

int 3

boolean true
```

Test des variables

fonction	description
isset(\$var)	true si \$var est initialisée et que sa valeur est différente de NULL
is_null(\$var)	true si \$var est NULL
is_bool(\$var)	true si \$var est booléen
is_numeric(\$var)	true si \$var peut être évaluée comme un numérique
is_float(\$var)	true si \$var est de type float
is_int(\$var)	true si \$var est de type entier
is_string(\$var)	true si \$var est de type string
is_array(\$var)	true si \$var est de type array
is_object(\$var)	true si \$var est de type object
empty(\$var)	true si \$var est évaluée à false ("","", 0, NULL, False, tableau vide, valeur non définie)

Conversions avec fonctions

fonction	description
intval(\$var, [\$base])	conversion en entier
floatval(\$var)	conversion en décimal
boolval(\$var)	conversion en booléen
strval(\$var)	conversion en string

Conversions avec opérateurs

fonction	description
(int) \$var	conversion en entier
(float) \$var	conversion en décimal
(bool) \$var	conversion en booléen
(string) \$var	conversion en string
(array) \$var	conversion en tableau
(object) \$var	conversion en objet

Destruction d'une variable

```
//Destruction d'une variable scalaire
$var = 5;
unset($var);
var_dump($var);

//Destruction d'une clef d'un tableau
$letters = [
    'a', 'b', 'c', 'd'
];

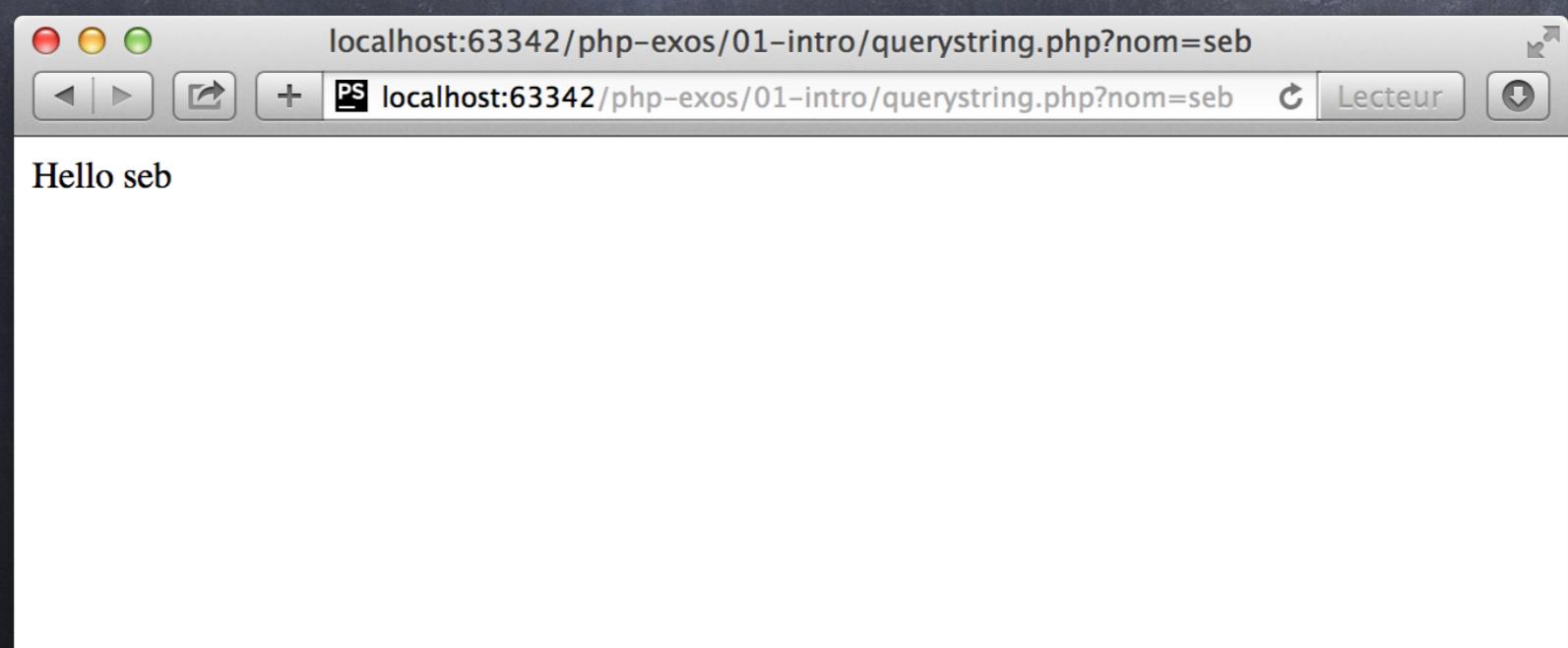
var_dump($letters);

unset($letters[2]);

var_dump($letters);
```

Passage de Paramètres

```
//$_GET contient les variables  
//passées dans l'URL  
$nom = $_GET["nom"];  
  
echo "Hello $nom";
```



Les superglobales

Collection	description
<code>\$_GET</code>	Variables transmises en GET (dans l'URL)
<code>\$_POST</code>	Variables transmises en POST (dans l'en-tête http)
<code>\$_COOKIE</code>	Les cookies
<code>\$_SESSION</code>	Les variables de session
<code>\$_REQUEST</code>	La combinaison de <code>\$_GET</code> , <code>\$_POST</code> et <code>\$_COOKIE</code>
<code>\$_FILES</code>	Les fichiers télétransmis
<code>\$_SERVER</code>	Informations sur le serveur et le contexte d'exécution du script
<code>\$_ENV</code>	Les variables d'environnement



Spaceship operator

```
$a = 5; $b = 3; $c = 5;  
//retourne 1  
echo $a <=> $b;  
  
//retourne -1  
echo $b <=> $a;  
  
//retourne 0  
echo $a <=> $c;
```



NULL coalesce

```
//Valeur par défaut
$name = $_GET["name"] ?? "inconnu";

//Equivalent à ce code
if(isset($_GET["name"])){
    $name = $_GET["name"];
} else {
    $name = "inconnu";
}
//Ou à ce code avec un expression ternaire
$name = isset($_GET["name"]) ? $_GET["name"] : "inconnu";
```

Constantes

```
//Définition d'une constante
define("ADMIN_EMAIL", "admin@site.com");

//Utilisation de la constante
echo ADMIN_EMAIL;

if(! defined(ADMIN_EMAIL)){
    echo "L'email est indéfini";
}

//Impossible de redéfinir une constante
define("ADMIN_EMAIL", "admin@site.com");
```

Les structures algorithmiques

Condition

```
//--- Condition simple
if($age >= 18){
    $message = "majeur";
} else {
    $message = "mineur";
}

echo 'vous êtes ' . $message. '<br>';
```

Condition

```
//--- Notation ternaire  
$message = $age>=18?'majeur':'mineur';  
echo 'vous êtes ' . $message. '<br>;  
  
//ou  
  
($age>=18)?$message='majeur':$message='mineur';  
echo $message.<br>;  
  
//ou  
($age>=18)?print 'majeur':print 'mineur';  
echo '<br>;
```

Switch

```
$score = 8;  
  
switch($score){  
    case 1:  
        $message = 'E';  
        break;  
    case 2:  
    case 3:  
        $message = 'D';  
        break;  
    case 4:  
    case 5:  
        $message = 'C';  
        break;
```

```
    case 6:  
    case 7:  
        $message = 'B' ;  
        break;  
    case 8:  
    case 9:  
        $message = 'A' ;  
        break;  
    case 10:  
        $message = 'A+' ;  
        break;  
default:  
        $message = 'Vous avez triché';  
}  
  
echo $message. '<br>';
```

Switch (Hack)

```
$age = 10;

switch(true){
    case ($age <=5):
        echo 'bébé';
        break;
    case ($age>5 and $age <=12):
        echo 'enfant';
        break;
    case ($age >12 and $age <=17):
        echo 'adolescent';
        break;
    case ($age >17 and $age <= 26):
        echo 'jeune adulte';
        break;
    case ($age >26 and $age <= 70):
        echo 'adulte';
        break;
    case ($age >70 and $age <= 90):
        echo 'vieil adulte';
        break;
    case ($age >90):
        echo 'adulte d\'âge canonique';
        break;
}
```

Boucle simple

```
for ($i = 1; $i <= 10; $i++) {  
    echo 'ligne' . $i . '<br>;  
}
```

```
for ($i = 10; $i >= 1; $i--) {  
    echo $i . ' ';  
}  
  
echo 'ignition <br>;
```

Double boucle

```
<table>
    <?php
    for ($i = 1; $i <= 10; $i++) {
        if ($i % 2) {
            echo '<tr style="background-color:#CCC;">';
        } else {
            echo '<tr>';
        }

        for ($k = 1; $k <= 10; $k++) {
            if ($k % 2 == 0) {
                echo '<td style="background-color:#888;">' . $i * $k . '</td>';
            } else {
                echo '<td>' . $i * $k . '</td>';
            }

            echo '</tr>';
        }
    }
    ?>
</table>
```

Boucle while

```
$compteur = 1;  
  
while($compteur <= 10){  
    echo $compteur. '<br>';  
    $compteur ++;  
}
```

PHP HTML

```
<?php if(isset($user)) : ?>
    <h1>Vous êtes authentifié</h1>
<?php else: ?>
    <h1>Authentifiez-vous</h1>
<?php endif ?>
```

PHP HTML

```
<ul>
    <?php for ($i = 1; $i <= 10; $i++) : ?>
        <li>item <?= $i ?></li>
    <?php endfor; ?>
</ul>
```

Les formulaires

Formulaire simple

```
<form method="post"
      action="traitement-formulaire.php">
    <label for="age">Votre Age</label>
    <input type="number" id="age" name="age">

    <button type="submit">Valider</button>
</form>
```



Traitement

```
<?php
//Affichage des données transmises
var_dump($_POST);

//Récupération des données postées
if(isset($_POST['age'])){
    $age = $_POST['age'];
} else {
    $message = "Vous devez passer par le formulaire pour saisir un age";
}

//Test des données
if(! is_numeric($age)){
    $message = "Vous devez saisir un nombre";
}elseif ($age <18){
    $message = "vous avez $age ans, vous êtes mineur";
} else {
    $message = "vous avez $age ans, vous êtes majeur";
}

//Affichage du message
echo $message;
```

Formulaire

```
<form method="post" action="traitement-formulaire.php">
    <fieldset>
        <legend>Contact</legend>

        <label>Civilité</label><br>
        <input type="radio" id="civF" name="contact[civilite]" value="madame">
        <label for="civF">Madame</label><br>
        <input type="radio" id="civM" name="contact[civilite]" value="monsieur">
        <label for="civM">Monsieur</label><br>

        <label for="nom">Nom</label>
        <input type="text" id="nom" name="contact[nom]"><br>

    </fieldset>

    <fieldset>
        <legend>Adresse</legend>

        <label for="voie">Adresse</label>
        <input type="text" id="voie" name="adresse[voie]"><br>
        <label for="code_postal">Code postal</label>
        <input type="text" id="code_postal" name="adresse[code_postal]"><br>
        <label for="ville">Ville</label>
        <input type="text" id="ville" name="adresse[ville]"><br>
    </fieldset>

    <button type="submit">Valider</button>
</form>
```

localhost:8000/formulaire-organisation/formulaire.html

localhost:8000/formulaire-organisation/formulaire.html Lecteur

Contact

Civilité

Madame

Monsieur

Nom

Adresse

Adresse

Code postal

Ville

Valider

Traitement

```
//Affichage des données transmises  
var_dump($_POST);  
  
$contact = $_POST['contact'];  
var_dump($contact);  
  
$adresse = $_POST['adresse'];  
var_dump($adresse);
```

Filtrage des variables

```
$age = filter_input(  
    INPUT_POST,  
    'age',  
    FILTER_VALIDATE_INT,  
    $options  
) ;
```

constante représentant
la collection source

la clef dans la
collection

le ou les filtres
de validation

un tableau des options
(facultatif)

Les collections

INPUT_GET

INPUT_POST

INPUT_COOKIE

INPUT_SERVER

INPUT_ENV

INPUT_SESSION était prévu
mais n'a jamais été
implémenté, la constante a
été supprimée depuis PHP
8.0.0

Les filtres de nettoyage

FILTER_SANITIZE_EMAIL

FILTER_SANITIZE_FULL_SPECIAL_CHARS

FILTER_SANITIZE_NUMBER_INT

FILTER_SANITIZE_NUMBER_FLOAT

FILTER_SANITIZE_URL

Liste des filtres

Les filtres de validation

FILTER_VALIDATE_EMAIL

FILTER_VALIDATE_BOOLEAN

FILTER_VALIDATE_INT

FILTER_VALIDATE_FLOAT

FILTER_VALIDATE_URL

FILTER_VALIDATE_REGEXP

Liste des filtres

Nettoyage

```
$nom = filter_input(INPUT_GET, 'nom',
    FILTER_SANITIZE_FULL_SPECIAL_CHARS);
//Récupère le nom sans les balises éventuelles
echo $nom;

$age = filter_input(INPUT_GET, 'age',
    FILTER_SANITIZE_NUMBER_INT);
//Ne récupère que les chiffres dans la saisie
echo $age;
```

Validation

```
$email = filter_input(INPUT_POST, 'email',  
    FILTER_VALIDATE_EMAIL);  
  
// $email = false si la validation échoue  
if (!$email){  
    echo "Vous devez saisir une adresse email valide";  
}
```

Filter callback

```
function convertSpace($value) {
    return str_replace(" ", "_", $value);
}

$string = "super PHP";

$string = filter_input(
    INPUT_GET,
    'string',
    FILTER_CALLBACK,
    array("options"=>"convertSpace")
);
```

Collection

Centres d'intérêt

+  localhost:8000/formulaire-collection  Lecteur 

Vos centres d'intérêt

- photo
- dessin
- programmation
- céramique

Valider

Le formulaire

```
<form method="post" action="traitement-formulaire.php">
    <h3>Vos centres d'intérêt</h3>

    <div>
        <input type="checkbox" id="photo" name="interet[]" value="photo">
        <label for="photo">photo</label><br>
        <input type="checkbox" id="dessin" name="interet[]" value="dessin">
        <label for="dessin">dessin</label><br>
        <input type="checkbox" id="programmation" name="interet[]" value="programmation">
        <label for="programmation">programmation</label><br>
        <input type="checkbox" id="ceramique" name="interet[]" value="ceramique">
        <label for="ceramique">céramique</label><br>
    </div>

    <button type="submit">Valider</button>
</form>
```

Le traitement

```
$interests = filter_input(  
    INPUT_POST,  
    'interet',  
    FILTER_DEFAULT,  
    FILTER_REQUIRE_ARRAY  
);  
  
var_dump($interests);
```

Test d'une variable

```
if (!filter_has_var(INPUT_GET, "email")) {  
    echo("Aucun email saisi");  
} else {  
    echo("Vous avez saisi un email");  
}
```

Tests multiples

```
$filters = array
(
    "name" => array
    (
        "filter"=>FILTER_CALLBACK,
        "options"=>"ucwords"
    ),
    "age" => array
    (
        "filter"=>FILTER_VALIDATE_INT,
        "options"=>array
        (
            "min_range"=>7,
            "max_range"=>77
        )
    ),
    "email"=> FILTER_VALIDATE_EMAIL,
);
$data = (filter_input_array(INPUT_POST, $filters));
var_dump($data);
```

Tester une variable

```
$email = "john.doe@example.com";

if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
    echo("$email est une adresse valide");
} else {
    echo("$email est invalide");
}
```

Tester plusieurs variables

```
$arr = [
    "name" => "sébastien maloron",
    "age" => "41",
    "email" => "smaloron@orange.fr",
];

$filters = [
    "name" => ["filter"=>FILTER_CALLBACK,
                "options"=>"ucwords"
            ],
    "age" =>  [ "filter"=>FILTER_VALIDATE_INT,
                "options"=>["min_range"=>1,
                            "max_range"=>120]
            ],
    "email"=> FILTER_VALIDATE_EMAIL,
];

$data = filter_var_array($arr, $filters);
var_dump($data);
```

Upload formulaire

```
<form method="post" action="upload.php"
      enctype="multipart/form-data"
```

Upload \$_FILES

```
var_dump($_FILES);
/*
array (size=1)
  'fichier' =>
    array (size=5)
      'name'          => string 'Four_gaz_4.jpg' (length=14)
      'type'          => string 'image/jpeg' (length=10)
      'tmp_name'      => string '/private/var/folders/
                                80/1n9sjk8j6gvct1n0fn0fxzsh0000gn/T/phpgqktDC' (length=66)
      'error'         => int 0
      'size'          => int 76745
*/
```

Les clefs de \$_FILES

clef	description
name	Le nom d'origine du fichier
type	Le type Mime du fichier
tmp_name	Nom et chemin du fichier temporaire sur le serveur
size	La taille du fichier en octet

Upload traitements

```
$targetPath = getcwd().'/images/';
//Attribution d'un nom unique au fichier
$filePath = $targetPath.uniqid('image_').'.'.$ext;

if(!move_uploaded_file($upload['tmp_name'], $filePath)){
    $message = "Impossible de déplacer le fichier temporaire";
    $ok = false;
} else {
    $message = "Upload réussi";
}
```

Les tableaux

Déclaration

```
$liens = ['Accueil', 'Contact', 'Recherche', 'A propos'];  
  
$fruits = array('pommes', 'poires');
```

Accès

```
//modification de la clef 0  
$fruits[0] = "oranges";  
  
//ajout de la clef 2  
$fruits[2] = "pastèques";  
  
//ajout de la clef 9 donc trous  
//dans la séquence  
$fruits[9] = "framboises";  
  
var_dump($fruits);
```

Boucle

```
//Boucle incrémentée
//pour les tableaux sans trous
for($i=0; $i<count($fruits); $i++){
    echo $fruits[$i];
}
//Fonctionne avec des trous
//dans la séquence
foreach ($fruits as $item){
    echo $item;
}
```

Les fonctions d'itération

fonction	description
current(\$arr)	Retourne l'élément en cours du tableau \$arr
next(\$arr)	Avance le curseur et retourne l'élément en cours du tableau \$arr
prev(\$arr)	Recule le curseur et retourne l'élément en cours du tableau \$arr
end(\$arr)	Positionne le curseur à la fin du tableau
reset(\$arr)	Positionne le curseur au début du tableau

Boucle avec itérateur

```
//Itérateurs sur les tableaux
for($i=1; $i<count($liens); $i++) {
    echo $i.' = '.current($liens).'  
';
    next($liens);
}

//Réinitialisation de l'itérateur
reset($liens);
//Itérateur avec boucle while
while(!empty(current($liens))){ 
    echo $i.' = '.current($liens).'  
';
    next($liens);
}
```

Quelques fonctions

fonction	description
array_sum()	Somme de tous les éléments d'un tableau
array_product()	Produit de tous les éléments d'un tableau
in_array(\$var,\$arr)	True si la valeur \$var est présente dans \$arr
sort(\$arr)	Tri le tableau \$arr par ordre ascendant
rsort(\$arr)	Tri le tableau \$arr par ordre descendant
natsort(\$arr)	Tri le tableau selon un ordre de tri naturel (2 inférieur à 10)
shuffle(\$arr)	Mélange aléatoirement les valeurs de \$arr
array_rand(\$arr)	Retourne une clef aléatoirement sélectionné au sein du tableau
array_unique(\$arr)	Supprime les doublons dans le tableau \$arr

fonctions de gestion de pile

fonction	description
array_push(\$arr,\$val)	Ajoute \$val à la fin du tableau \$arr et retourne la nouvelle taille du tableau
\$arr[] = \$val	Fait la même chose que array_push
array_unshift(\$arr,\$val)	Ajoute \$val au début du tableau \$arr et retourne la nouvelle taille du tableau
array_pop(\$arr)	Supprime le dernier élément d'un tableau et retourne ce élément supprimé
array_shift(\$arr)	Supprime le premier élément d'un tableau et retourne ce élément supprimé

Les tableaux associatifs

```
$person = [  
    "nom" => "Sébastien",  
    "age" => 46  
];  
//récupération d'une valeur  
echo $person["nom"];  
  
//Modification d'une clef  
$person["age"] = 47;  
  
//Création d'une nouvelle clef  
$person["taille"] = 170;
```

Boucle sur tableaux associatifs

```
//Avec foreach
foreach ($person as $key=>$val){
    echo "$key = $val";
}

//Avec un tableau des clefs
$keys = array_keys($person);
for($i=0; $i < $keys; $i++){
    echo $person[$keys[$i]];
}
```

L'export des tableaux

```
$person = ["Seb", 46, "bleu"];  
list($name, $age, $color) = list($person);  
echo $name;
```

```
$person = [  
    "name" => "Seb",  
    "age" => 46  
];  
extract($person);  
echo "$name $age";
```

Les fonctions

Déclaration

```
//Définition de la fonction
function hello($name){
    echo "hello $name";
}

//Appel de la fonction
hello("Seb");
```

Valeur par défaut

```
//Définition de la fonction
function hello($name = "toto") {
    echo "hello $name";
}

//Appel de la fonction
hello();
```

Les arguments par défaut doivent être à la fin de la liste des arguments

Nombre d'arguments variable

A partir de
PHP 5.6

```
function add(...$numbers){  
    $total = 0;  
    foreach ($numbers as $n){  
        $total += $n;  
    }  
    return $total;  
}  
  
echo add(5,8,2,8);
```

Nombre d'arguments variable

Avant
PHP 5.6

```
function sum() {  
    $total = 0;  
    foreach (func_get_args() as $n) {  
        $total += $n;  
    }  
    return $total;  
}  
  
echo sum(1, 2, 3, 4);
```

Passage d'arguments

A partir de
PHP 5.6

```
function add($a, $b) {  
    return $a + $b;  
}  
  
echo add(...[1, 2]);  
  
$a = [1, 2];  
echo add(...$a);
```

Transforme un tableau
en une liste d'arguments

Passage d'arguments par référence

```
function addString(&$str){  
    $str .= " dans un pays fort lointain";  
}  
  
$str ="Il était une fois";  
addString($str);  
  
echo $str;
```

Par défaut tous les arguments sont passés par valeur à l'exception des objets



Type des arguments

```
function add(int $n, int $total){  
    return $total + $n;  
}  
//Ok conversion implicite  
echo add("5", 2);  
  
//Type Error  
echo add("a", 8);
```

Les types

int

float

bool

string

array

callable

class/interface



Mode strict

```
declare(strict_types=1);

function add(int $n, int $total){
    return $total + $n;
}

//Type Error
echo add("5", 2);

//Type Error
echo add("a", 8);
```

Valeur de retour

```
//Définition de la fonction
function hello($name = "toto"){
    return "hello $name";
}

//Appel de la fonction
echo hello();
```



Type de retour

```
declare(strict_types=1);

function add(int $n, int $total):int{
    return $total + $n;
}

echo add(12, 2);
```

Variables fonctions

```
function hello($name){  
    echo "Hello $name";  
}  
  
$func = "hello";  
  
$func("seb");
```

Fonctions anonymes

```
$greet = function ($name){  
    echo "Hello $name";  
};  
  
$greet("seb");
```

Importation des variables du parent scope

```
$message = "Hello";  
  
$greet = function ($name) use ($message){  
    echo "$message $name";  
};  
  
$greet("seb");
```

Closure

```
function greetingFunction($timeOfDay = "morning") {  
  
    return ( function( $name ) use ( &$timeOfDay ) {  
        return ( "Good $timeOfDay, $name!" );  
    } );  
};  
  
$goodMorning = greetingFunction();  
$goodAfternoon = greetingFunction("afternoon");  
  
echo $goodAfternoon("Seb");
```

Nouvelle syntaxe

```
$message = 'hello';
$greet = static fn ($text) => "$message $text";

echo $greet('world');
```

Inutile d'importer le
contexte avec cette
nouvelle syntaxe

Les inclusions

Utilité

- ⊕ Chargement des dépendances
(bibliothèques de fonction)
- ⊕ Configuration de l'application
- ⊕ Découpage des vues en éléments indépendants

Les fonctions d'inclusion

fonction	description
include(\$path)	Inclus le contenu du fichier \$path et retourne un warning si le fichier est absent
include_once(\$path)	Comme include, mais n'inclus le fichier qu'une seule fois
require(\$path)	Inclus le contenu du fichier \$path et retourne une erreur si le fichier est absent (donc interrompt le traitement)
require_once(\$path)	Comme require, mais n'inclus le fichier qu'une seule fois

retour des inclusions

```
//return.php
<?php
$var = 'PHP';

return $var;
?>
```

```
//norelease.php
<?php
$var = 'PHP';
?>
```

```
<?php

$foo = include 'return.php';

echo $foo; // affiche 'PHP'

$bar = include 'norelease.php';

echo $bar; // affiche 1

?>
```

Export des variables

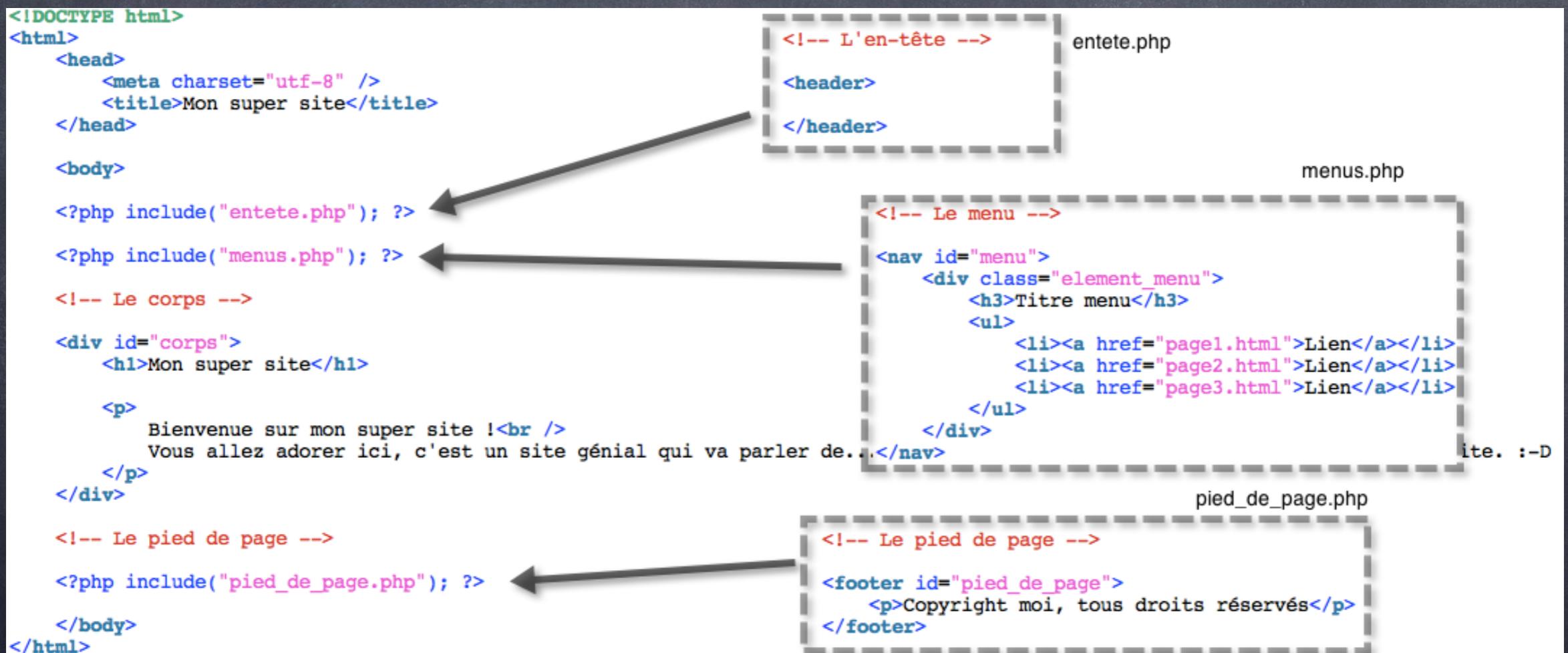
```
//include.php
<?php
$nom .= ' Maloron';
?>
```

```
<?php
$nom = 'Sébastien';

include 'include.php';

// affiche 'Sébastien Maloron'
echo $nom;
?>
```

Inclusions de gabarits



Fonctions relatives aux chemins

fonction	description
getcwd()	retourne le chemin absolu du dossier contenant le script qui exécute cette fonction
realpath(\$path)	traduit un chemin relatif en chemin absolu
__DIR__	retourne le chemin du dossier en cours
dirname(\$path)	retourne le chemin du dossier parent de \$path

Les en-têtes http

SYNTAXE

```
header(header, replace, responseCode)
```

argument	description
header	requis, un chaîne représentant l'en-tête à envoyer
replace	facultatif, un booléen indiquant si l'en-tête doit remplacer un éventuel en-tête existant
responseCode	facultatif, un entier forçant un code de réponse http

réponse http

code	description
20XX	succès
30XX	redirection
40XX	accès impossible à la ressource : non trouvée, accès non autorisé ou mauvaise requête
50XX	erreur interne du serveur

redirections

```
//Redirection relative vers un fichier page.php  
//se trouvant dans le même dossier  
$redirectTarget = "page.php";  
header("location:$redirectTarget");  
  
//Redirection relative vers un fichier page.php  
//se trouvant un dossier au dessus  
$redirectTarget = "../page.php";  
header("location:$redirectTarget");  
  
//Redirection absolue vers un fichier page.php  
//se trouvant dans le dossier racine du serveur web  
//(DOCUMENT_ROOT)  
$redirectTarget = "/page.php";  
header("location:$redirectTarget");  
  
//Redirection adresse absolue  
//vers mon site  
$redirectTarget = "http://www.monsite.com/page.php";  
header("location:$redirectTarget");
```

Forcer un type de contenu

```
header('Content-Type: application/json');
$user = array(
    'name' => 'Maloron',
    'lastname' => 'Sébastien',
    'age' => 87
);

echo json_encode($user);
```

Forcer le téléchargement

```
$fileName = 'participants.csv';

header("Content-type:application/csv");

// filename permet de suggérer un nom de fichier
header("Content-Disposition:attachment;filename='$fileName'");

// Affichage du contenu
echo "nom;age\n";
echo "Pierre;20\n";
echo "Jean;32\n";
echo "Odile;40\n";
```

Les sessions

Principes

- Données persistantes stockées sur le serveur relatives à une session d'utilisation.
- Le serveur génère un identifiant de session envoyé au client.
- Le client retransmet son identifiant de session à chaque requête http
- Les données restent stockées sur le serveur

Variables de session

```
//fichier start.php
session_start();
$_SESSION["couleur"] = "rouge";

header("location:end.php");

//fichier end.php
$couleur = $_SESSION["rouge"] ?? "bleu";

echo $couleur;
```

Authentification

```
session_start();  
  
$userName = filter_input(INPUT_POST, 'userName', FILTER_SANITIZE_STRING);  
$password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);  
  
if($userName=="admin" && $password=="pass") {  
    //génère un nouvel identifiant de session  
    //et détruit l'ancienne session  
    session_regenerate_id(true);  
  
    $_SESSION["user"] = $userName;  
    $_SESSION["IP"] = $_SERVER["REMOTE_ADDR"];  
  
} else {  
    header("location:login.php");  
}
```

Test d'authentification

```
session_start();  
  
$userName = filter_var($_SESSION["user"], FILTER_SANITIZE_STRING);  
$ip = filter_var($_SESSION["IP"], FILTER_VALIDATE_IP);  
  
//Vérification de la validité de la session  
if($userName && $ip == $_SERVER["REMOTE_ADDR"]){  
    session_regenerate_id(true);  
  
    $_SESSION["user"] = $userName;  
    $_SESSION["IP"] = $_SERVER["REMOTE_ADDR"];  
  
} else {  
    header("location:login.php");  
}
```

déconnexion

```
session_start();  
//Génère un nouvel identifiant de session  
//et détruit l'ancienne session  
session_regenerate_id(true);  
  
header("location:index.php");
```

Les fickiers

Lecture/écriture

```
$fichier = 'textes/mon-fichier.txt';
//Lecture du contenu d'un fichier texte
//et stockage dans une variable
$contenu = file_get_contents($fichier);

//Ajout d'une nouvelle ligne
$contenu .= '\nUne nouvelle ligne';

//Enregistrement du nouveau contenu dans le fichier
file_put_contents($fichier, $contenu);
```

Ajout à un fichier

```
$fichier = 'data/contacts.txt';

$data = '\nSébastien Maloron';

//Pose un verrou sur le fichier
//(pour éviter les accès concurrents)
//et ajoute le contenu de $data à la fin du fichier
file_put_contents($fichier, $data, FILE_APPEND | LOCK_EX);
```

Fichier délimité

```
$fichier = 'tableau.csv';
$data = file_get_contents($fichier);

//Découpage du fichier texte dans un tableau indicé
//Chaque occurrence du caractère de saut de ligne
//génère une case de tableau
$lignes = explode('\n', $data);

//Boucle sur chaque ligne
for($i=0; $i<count($lignes);$i++){
    //Découpage d'une ligne
    //en un tableau de colonnes
    //Ici le délimiteur est la tabulation
    $colonnes = explode('\t', $lignes[$i]);
    ...
}
```

Fichier json

```
$file = 'tableau.json';
$data = json_decode(
    file_get_contents($fichier)
);

$data[] = ['name'=>'Olga', 'age'=> 43];

file_put_contents($file, json_encode($data));
```

Lecture d'un dossier

```
$path = 'data';  
  
$content = scandir($path);  
  
var_dump($content);
```

Lecture d'un dossier

```
$path = 'data';
$folderContent = scandir($path);

foreach($folderContent as $item) {

    if ($item == '.' || $item == '..') {

    } elseif (is_file($path . '/' . $item)) {
        echo $item . ' (fichier)';
    } elseif (is_dir($path . '/' . $item)) {
        echo $item . ' (dossier)';
    }
}
```

Filtre dans un dossier

```
foreach (glob("img/*.jpg") as $filename) {  
    echo "$filename size " . filesize($filename) . "\n";  
}
```

Output buffering

Principes

- Le résultat de l'interprétation PHP est stocké en mémoire au lieu d'être immédiatement envoyé dans la réponse
- Il est possible de capturer cette mémoire tampon dans une variable

Exemple

```
<?php
//Activation de la mise en tampon de la sortie (output) PHP
ob_start();

$list = range(0, 100, 10);
?>
<ul>
    <?php foreach ($list as $number): ?>
        <li><?= $number ?></li>
    <?php endforeach; ?>
</ul>

<?php
//Récupération de la sortie dans une variable
$content = ob_get_clean();

//Affichage de la sortie
var_dump($content);
?>
```

Avec un include

```
<?php
//Activation de la mise en tampon de la sortie (output) PHP
ob_start();

$list = range(0, 100, 10);

include 'templates/list-template.php';

//Récupération de la sortie dans une variable
$content = ob_get_clean();

//Affichage de la sortie
var_dump($content);
?>
```