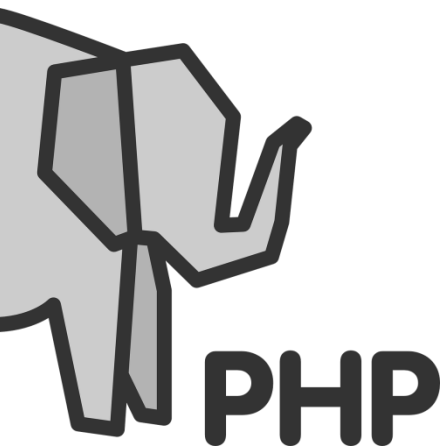


# PHP & MySQL



**PHP (PHP Hypertext Preprocessor)** : Langage de scripts généralistes et Open Source, spécialement conçu pour le développement d'applications web. Il est facilement intégrable au HTML. Il permet de produire des pages web dynamiques. C'est un langage qui peut être orienté objet (POO : Programmation Orientée Objet) sans y être obligé (contrairement au langage Java).

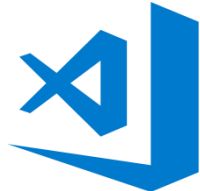
**SQL (Structured Query Language)** : Langage de requêtes permettant d'échanger des informations avec une base de données (interroger, enregistrer, modifier, supprimer). Ce langage est normé mais possède quelques altérations suivant le système de base de données employé.



# Installation

Pour utiliser PHP vous aurez besoin :

1. Serveur web
2. Éditeur de texte ou IDE **Integrated Development Environment** (en français « Environnement de Développement Intégré » ou EDI)
3. MySQL (pour la suite du cours)



XAMPP



## Echo : Hello World !

```
<div class="album py-5 bg-light">
  <div class="container">
    <div class="row">
      <div class="col-md-12">
        <!-- ICI COMMENCE NOTRE PHP -->
        <?php echo 'Hello World !'; ?>
      </div>
    </div>
  </div>
</div>
```



## Les commentaires

```
<div class="col-md-12">  
  <!-- ICI COMMENCE NOTRE PHP -->  
  <?php echo 'Hello World !'; ?>  
  <?php // Voici un commentaire sur une ligne en PHP  
    /*  
      Ici un commentaire  
      multiligne  
    */  
  ?>  
</div>
```



## Les variables

Une variable peut contenir plusieurs types de données (string, array, int, bool, float, NULL)

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    $maVariable = 'Toto'; // J'affecte une valeur à maVariable
    echo $maVariable;
  ?>
</div>
```

**⚠ Astuce**  
**Utilisez les règles de nommages :**  
maVariable (camelCase) ou ma\_variable



## Les variables (rappels)

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    $maVariable = 'Toto'; // J'affecte une valeur à maVariable
    echo $maVariable;

    // Rappels
    $texte      = "du texte"; // STRING
    $chiffre     = 18; // INT
    $nombre_virgule = 3.14; // FLOAT
    $booleen     = true; // BOOLEEN
  ?>
</div>
```

### ⚠ Rappels

- ✓ String = chaîne de caractères
- ✓ Integer (int) = chiffres
- ✓ Float = Nombre à virgule
- ✓ Booleen = Vrai ou Faux ?



# Les constantes

Une **constante** permet de stocker en mémoire une valeur qui ne sera pas modifiable par vos scripts (on peut aussi parler de « lecture seule »).

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Je définie une constante :
    define('AGE', 21);
    echo AGE; // 21
    // Je définie une constante en PHP V5.3+
    const NEW_AGE = 5 + AGE;
    echo NEW_AGE; // 26
  ?>
</div>
```

## ⚠️ Rappels

- ✓ Define('AGE', 21); // définit une constante
- ✓ Depuis PHP 5.3 : const NEW\_AGE = 5 + AGE;
- ✓ S'écrit toujours en MAJUSCULE





# Les tableaux

Les tableaux permettent de stocker un grand volume de données rangées de manière ordonnées sous le format clé/valeur.

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Ancienne version de déclaration de tableau
    $anciens_jeux = array();
    // Version à privilégier
    $jeux = ['The Witcher', 'Counter Strike', 'Dragon Ball FighterZ'];
    echo '<pre>'; // Balise HTML de présentation
    var_dump($jeux);
    echo '</pre>';
  ?>
</div>
```

```
array(3) {
  [0]=>
  string(11) "The Witcher"
  [1]=>
  string(14) "Counter Strike"
  [2]=>
  string(20) "Dragon Ball FighterZ"
}
```

## ⚠️ Rappels

- ✓ Définir un tableau via la fonction array() ou [];
- ✓ Indice (index) pour désigner le numéro de ligne du tableau
- ✓ Clé pour désigner le rang d'un tableau au format texte

## ⚠️ Astuces

- ✓ var\_dump() permet de debugger un élément
- ✓ <pre></pre> en HTML permet de présenter un résultat



# Les tableaux

## Les tableaux indexés

```
echo '<pre>'; // Balise HTML de présentation  
var_dump($jeux[1]);  
echo '</pre>';
```

string(14) "Counter Strike"



# Les tableaux

## Les tableaux associatifs

```
// Tableau avec des clés/valeurs  
$buletin_notes = ['Toto' => 12]; // Déclaration d'un tableau  
echo '<pre>'; // Balise HTML de présentation  
var_dump($buletin_notes);  
echo '</pre>';
```

```
array(1) {  
    ["Toto"]=>  
    int(12)  
}
```



# Les tableaux

## Les tableaux associatif

```
// Notes de jeux vidéos
$notes_jeux_videos = [
    'The Witcher'           => 20,
    'Counter Strike'       => 16,
    'Dragon Ball FighterZ' => 18
];
echo '<pre>'; // Balise HTML de présentation
    var_dump($notes_jeux_videos);
echo '</pre>';
```

```
array(3) {
    ["The Witcher"]=>
    int(20)
    ["Counter Strike"]=>
    int(16)
    ["Dragon Ball FighterZ"]=>
    int(18)
}
```



# Les tableaux

## Les tableaux multidimensionnels

```
// Tableau multidimensionnel
$equipe_rugby = [ // Tableau
    'Avant' => [ // Première partie de mon tableau
        1 => 'pillier',
        2 => 'talonneur',
        3 => 'pillier',
        4 => 'lock',
        5 => 'lock',
        6 => 'flankers',
        7 => 'flankers',
        8 => 'flankers'
    ],
    'Arrière' => [ // Deuxième partie de mon tableau
        9 => 'Demi de mêlée',
        10 => 'Demi ouverture',
        11 => 'Trois-quart',
        12 => 'Centre',
        13 => 'Centre',
        14 => 'Ailier',
        15 => 'Arrière'
    ]
];
```

```
echo '<pre>';
    var_dump($equipe_rugby);
echo '</pre>';
```



# Les tableaux

## Les tableaux multidimensionnels

Si je souhaite accéder à une valeur il me suffit de parcourir le tableau en utilisant leurs indexes.

```
// Afficher la seconde partie du tableau  
echo '<pre>';  
    var_dump($equipe_rugby['Arrière']);  
echo '</pre>';  
  
// Afficher la seconde partie en ciblant le numéro 10  
echo '<pre>';  
    var_dump($equipe_rugby['Arrière'][10]);  
echo '</pre>';
```

```
array(7) {  
    [9]=>  
        string(15) "Demi de mêlée"  
    [10]=>  
        string(14) "Demi ouverture"  
    [11]=>  
        string(11) "Trois-quart"  
    [12]=>  
        string(6) "Centre"  
    [13]=>  
        string(6) "Centre"  
    [14]=>  
        string(6) "Ailier"  
    [15]=>  
        string(8) "Arrière"  
}  
  
string(14) "Demi ouverture"
```



# Les tableaux

Fonctions utiles :

- ✓ `in_array()` : vérifier qu'une valeur existe dans un tableau
- ✓ `array_push()` : empile un (ou plusieurs) élément(s) à la fin du tableau
- ✓ `array_pop()` : dépile un élément à la fin du tableau
- ✓ `array_shift()` : extrait et enlève un élément au début du tableau
- ✓ `array_unshift()` : empile un (ou plusieurs) élément(s) au début du tableau
- ✓ `count()` : compte les éléments d'un tableau
- ✓ `explode()` : découpe une chaîne en segments (pour les placer dans un tableau par exemple)
- ✓ `implode()` : rassemble les éléments d'un tableau en une chaîne, reliés par une glue



# Les opérateurs

Opérateur d'affectation par valeur

```
<div class="col-md-12">  
  <!-- ICI COMMENCE NOTRE PHP -->  
  <?php  
    // Les opérateurs  
    // Affectation  
    $a = 1;  
    $b = $a;  
    echo $a; // 1  
    echo $b; // 1  
  ?>  
</div>
```





# Les opérateurs

## Opérateur d'affectation par référence

```
// Affectation par référence
$prenom = 'Jérôme';
$reference_prenom = &$prenom;

echo '<pre>';
    echo $prenom; // Jérôme
    echo '<br>';
    echo $reference_prenom; // Jérôme
echo '</pre>';
```



# Les opérateurs

Opérateur d'affectation par référence

```
$prenom = 'JD'; // Je réaffecte une valeur à $prenom  
echo '<pre>';  
    echo $prenom; // JD  
    echo '<br>';  
    echo $reference_prenom; // JD  
echo '</pre>';
```



# Les opérateurs

## Opérateurs arithmétiques

```
// Les opérateurs arithmétiques

$x = 1;
$y = 5;

echo $x + $y;    // Somme : 6
echo $y - $x;    // Soustraction : 4
echo $y * $x;    // Multiplication : 5
echo $y / $x;    // Division : 5
echo $y ** $x;   // Exponentiation : 5
echo $y % $x;    // Modulo : 0
echo -$y;        // Opposé : -5
echo ++$x;       // Pré-incrémentation : 2
echo $y++;       // Post-incrémentation : 5
echo $y;         // $y vaut maintenant 6
echo --$x;       // Pré-décrémentation : 1
echo $y--;       // Post-décrémentation : 6
echo $y;         // $y vaut maintenant 5
```



# Les opérateurs

## Opérateurs de chaînes

```
// Les opérateurs de chaîne
$string_1 = 'Bonjour, ';
$prenom   = 'Jérôme ';
$age      = 30;
$string_2 = ' tu as ';
$string_3 = ' ans.';

echo $string_1 . $prenom . $string_2 . $age . $string_3;
```

Bonjour, Jérôme tu as 30 ans.



# Les opérateurs

## Opérateurs de comparaisons

```
// Les opérateurs de comparaisons
echo '<pre>';
var_dump($y == $x);    // Égalité
var_dump($y === $x);   // Égalité et identique
var_dump($y != $x);    // Différent
var_dump($y !== $x);   // Différent ou de type différent
var_dump($y < $x);     // Inférieur
var_dump($y <= $x);    // Inférieur ou égale
var_dump($y > $x);     // Supérieur
var_dump($y >= $x);    // Supérieur ou égale
echo '</pre>';
```



# Les opérateurs

## Opérateurs logiques

```
// Les opérateurs logiques
$y && $x; // Et / AND
$y || $x; // Ou / OR
$y xor $x; // si $x OU $y est TRUE, mais pas les deux en même temps
!$y;      // Non logique
```



# Les opérateurs

## Opérateur ternaire

```
// Opérateur ternaire
// Si $variable1 est égale à true alors
// J'affiche $variable1
// Sinon j'affiche $variable 2
$variable1 = true;
$variable2 = false;

echo $variable1 ? $variable1 : $variable2;
```



### Rappels



<https://www.php.net/manual/fr/language.operators.php>



# Les conditions

If, elseif, else permet de contrôler sous condition un test.

Si mon résultat est vrai alors je peux déclencher une instruction, sinon une autre instruction sera proposée.

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Les conditions
    $true   = true;
    $false  = false;
    $cinq   = 5;
    $deux   = 2;

    if($cinq > $deux){
      echo $cinq . ' est bien supérieur à ' . $deux;
    }
  ?>
</div>
```





# Les conditions

## ⚠ Astuce

**Vous pouvez utiliser le ternaire**

Observer bien la ponctuation :

? veut dire ALORS

: Veut dire SINON

```
// Condition ternaire  
echo ($cinq > $deux) ? $cinq . ' est bien supérieur à ' . $deux : '';
```

5 est bien supérieur à 2



# Les conditions

Vous pouvez aussi mixer l'HTML et le PHP pour plus de lisibilité.

```
<?php if($cinq > $deux) : // Mixe des écritures ?>  
    <p>Instruction 1 : Je suis dans le premier cas</p>  
<?php endif; ?>
```

Instruction 1 : Je suis dans le premier cas

## ⚠️ Rappels

**Observer l'ouverture et la fermeture  
des balises PHP**

## ⚠️ Rappels

Vous pouvez utiliser :

If : endif;

If : elseif : endif;



# Les conditions

Plusieurs conditions à la suite.

Prenons l'exemple suivant :

Si (*cinq est inférieur à deux*){  
    ALORS j'exécute une première instruction

}

SINON SI (*cinq est égale à deux*){  
    ALORS j'exécute une deuxième instruction

}

SINON {  
    j'exécute une instruction par défaut  
}

```
<?php
if ($cinq < $deux) {
    // Pour échapper des caractères utiliser comme ci-dessous l'antislash
    echo 'On est d\'accord pour dire que c\'est faux ?';
}elseif ($cinq == $deux) {
    echo 'Là aussi c\'est faux';
}else{
    echo 'Aucun de ces tests n\'est juste !';
}
?>
```

Aucun de ces tests n'est juste !



## La boucle « tant que » while

La boucle while permet de lancer une action répétée tant que la condition n'est pas réunie. Si nous voulons stopper la boucle, il faut que la condition soit valide.

```
<?php
// Les boucles
// La boucle WHILE
/*
    Tant que tu n'as pas atteint la valeur 5 tu ajoutes
    une unité au chiffre précédent.
*/
$chiffre = 0;
echo '<ul>';
while ($chiffre <= 5) {
    echo '<li>Numéro du tour de boucle : ' . ($chiffre + 1) . '</li>';
    $chiffre++; // J'incrmente de 1 le chiffre en base
}
echo '</ul>';
?>
```



## La boucle « tant que » while

Vous pouvez aussi utiliser cette syntaxe :

```
<ul>
  <?php $chiffre = 0; ?>
  <?php while ($chiffre <= 5) : ?>
    <li>Numéro du tour de boucle : <?= ($chiffre + 1); ?> </li>
    <?php $chiffre++; // J'incrmente de 1 le chiffre en base ?>
  <?php endwhile; ?>
</ul>
```



# La boucle «fait une fois puis tant que » do while

La boucle exécutera le script une fois puis testera la condition.

```
<?php
// La boucle DO WHILE
$im_a_dev = "Je suis un dev !";
$length   = strlen($im_a_dev);
$i         = 0;
do {
    // Affiche la première lettre en majuscule suivit d'un point
    echo strtoupper($im_a_dev[$i] . '-') ;
    $i++;
} while ( $i < $length);
?>
```



## La boucle for

La boucle exécute une instruction de manière itérative.

La boucle for prend 3 paramètres qui serviront à initialiser la boucle puis à tester la condition d'exécution et enfin incrémenter la boucle si la condition n'est pas réussie.

```
<?php
// La boucle FOR
for ($i=0; $i < 10 ; $i++) {
    echo '<p>Tour de boucle numéro ' . $i . '</p>';
}
// On peut aussi écrire la boucle FOR sous ce format :
?>
<?php for ($i=0; $i < 10; $i++) : ?>
    <p>Tour de boucle numéro <?= $i; ?></p>
<?php endfor; ?>
```

### ! Astuces

Vous pouvez utiliser <?= à la place de <?php echo.



## La boucle switch

La boucle switch peut être utilisée pour comparer plusieurs cas, un peu comme le if et elseif. Elle sera très utile quand nous serons en présence de cas pré-programmés.

```
<?php
// La boucle SWITCH
$i = 0;
switch ($i) {
    case 1: // Dans le cas ou $i est égale à 1
        echo 'La variable vaut : ' . $i;
        break; // Si c'est bien ce cas je stoppe le switch
    case 2: // Dans le cas ou $i est égale à 1
        echo 'La variable vaut : ' . $i;
        break; // Si c'est bien ce cas je stoppe le switch

    default:
        echo 'Je n\'ai pas trouvé de valeur correspondante';
        break;
}
?>
```





# La boucle foreach

La boucle foreach sera une des boucles les plus utilisées en PHP. Elle sert à parcourir des tableaux ou objets afin d'en extraire les données.

```
<?php
// La boucle FOREACH
$notes_jeux_videos = [
    'The Witcher'      => 20,
    'Counter Strike'   => 16,
    'Dragon Ball FighterZ' => 18
];
foreach ($notes_jeux_videos as $nom_du_jeu => $note) {
    echo '<p>' . $nom_du_jeu . ' a obtenu la note de : ' . $note . '</p>';
}
?>
```

## ! Rappels

- ✓ Contrairement à while ou for, foreach n'a pas besoin d'itération.
- ✓ Vous pouvez utiliser aussi la syntaxe avec les :.
- ✓ Foreach (Tableau à parcourir AS clé => valeur)

```
<?php foreach ($notes_jeux_videos as $nom_du_jeu => $note) : ?>
    <p> <?= $nom_du_jeu; ?> a obtenu la note de : <?= $note; ?></p>
<?php endforeach; ?>
```



# Les inclusions de fichiers

Une bonne pratique dans PHP est le fait de ne pas réutiliser le code inutilement. Prenons l'exemple d'un site web très basique. Celui-ci comportera toujours 3 zones au minima bien distinctes : HEADER – MAIN – FOOTER.

```
<!-- Appel du HEADER -->
<?php include 'header.php'; ?>

<main role="main">

  <section class="jumbotron text-center">
    <div class="container">
      <h1 class="jumbotron-heading">Cours de PHP & MySQL</h1>
      <p class="lead text-muted">Modèle pour notre cours.</p>
    </div>
  </section>

  <div class="album py-5 bg-light">
    <div class="container">
      <div class="row">
        <div class="col-md-12">
          <!-- ICI COMMENCE NOTRE PHP -->
        </div>
      </div>
    </div>
  </div>

</main>
<!-- Appel du FOOTER -->
<?php include 'footer.php'; ?>
```

**⚠️ Rappels**  
**Les fonctions include, require, include\_once et require\_once**  
✔️ Permettent d'inclure des fichiers dans d'autres fichiers

📄 footer.php  
📄 header.php  
📄 index.php



# Les fonctions

Avec PHP vous allez pouvoir définir vos propres fonctions. Vous avez déjà utilisé des fonctions mais celle-ci étaient programmées par le langage et donc fournies nativement. Ici nous allons parler de fonctions utilisateur (définie par l'utilisateur).

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Les fonctions
    // Une fonction se déclare avec le mot clé "function"
    // suivie d'un nom de fonction.
    function identification(){
      return "Bonjour, votre nom est Jérôme !";
    }
    // Pour exécuter une fonction, il faut l'appeler.
    identification();
  ?>
</div>
```

## ⚠️ Rappels

- ✓ Mot clé « function »
- ✓ Nom de ma fonction
- ✓ Retourne un résultat
- ✓ Exécution

## ⚠️ Rappels

### Listes des fonctions PHP

- ✓ <https://www.php.net/manual/fr/funcref.php>



# Les fonctions

Avec PHP vous allez pouvoir définir vos propres fonctions. Vous avez déjà utilisé des fonctions mais celle-ci étaient programmées par le langage et donc fournies nativement. Ici nous allons parler de fonctions utilisateur (définie par l'utilisateur).

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Les fonctions
    // Une fonction se déclare avec le mot clé "function"
    // suivie d'un nom de fonction.
    function identification(){
      return "Bonjour, votre nom est Jérôme !";
    }
    // Pour exécuter une fonction, il faut l'appeler.
    identification();
  ?>
</div>
```

## ! Rappels

- ✓ Mot clé « function »
- ✓ Nom de ma fonction
- ✓ Retourne un résultat
- ✓ Exécution

## ! Rappels

### Listes des fonctions PHP

- ✓ <https://www.php.net/manual/fr/funcref.php>



# Les fonctions

Vos fonctions vont pouvoir intégrer des paramètres. Les paramètres vont vous donner la possibilité de personnaliser vos fonctions. Vous pouvez aussi dans vos fonctions attribuer des paramètres par défaut.

```
// Fonction avec paramètres
function operation($premier_chiffre, $deuxieme_chiffre){
    $resultat = $premier_chiffre * $deuxieme_chiffre;
    return $resultat;
}
echo operation(10, 5);
```



# Les fonctions

Fonctions avec des paramètres par défaut.

```
// Fonction avec paramètres par défaut
function connexion($user = "root", $password = ""){
    if($user === "root"){
        return "Bonjour, $user, vous êtes administrateur";
    }else{
        return "Bonjour, $user, vous êtes un abonné";
    }
}
// Utilisation des paramètres par défaut
echo connexion();
// Utilisation en remplaçant les paramètres par défaut
echo connexion("Jérôme", "J'AIME PHP");
```

## ! Astuces

- ✓ Vous remarquerez, que les variables peuvent être utilisées à l'intérieur des doubles côtes.



# Les dates

PHP permet toute sorte de manipulation avec le temps via les dates. Avec PHP nous pouvons donc manipuler les dates avec un TIMESTAMP (en informatique et dans le web le TIMESTAMP est défini au 1<sup>er</sup> janvier 1970).

```
<div class="col-md-12">
  <!-- ICI COMMENCE NOTRE PHP -->
  <?php
    // Les dates
    // Nous allons définir un fuseau horaire
    date_default_timezone_set('Europe/Paris');
    // Afficher une date
    echo date("d-m-Y H:i:s");
    // jour-mois-année (4 chiffres)
    // heures:minutes:secondes
  ?>
</div>
```

## ⚠ Astuces

- ✓ Depuis PHP 5.1 nous pouvons définir un fuseau horaire à nos scripts
- ✓ <https://www.php.net/manual/fr/function.date.php>

## ⚠ Définition

- ✓ [https://fr.wikipedia.org/wiki/Heure\\_Unix](https://fr.wikipedia.org/wiki/Heure_Unix)



# Les dates

Manipulations avec les dates : checkdate()

```
// Vérifier une date valide
if(checkdate(month, day, year)){
    echo "Date valide";
}else{
    echo "Date non valide";
}
```

```
// Vérifier une date valide
if(checkdate(12, 24, 2020)){
    echo "Date valide";
}else{
    echo "Date non valide";
}
```





# Les dates

Manipulations avec les dates : strftime()

```
// Utilisation de strftime()  
setlocale(LC_ALL, 'fr_FR');  
echo strftime('%A %d %B %Y');  
// en US  
setlocale(LC_ALL, 'us_US');  
echo strftime('%A %d %B %Y');
```

## ⚠ Astuces

🔗 <https://www.php.net/manual/fr/function.strftime.php>



# Les dates

Exercice : Utilisation de mktime()

Exercice : Quel jour serons-nous le 22 janvier 2059 ?

- ✓ Astuce 1 : Utiliser mktime(hour, minute, second, month, day, year);
- ✓ Astuce 2 : date() peut convertir un mktime en date standard
- ✓ date("d-m-Y", mktime());
- ✓ Astuce 3 : Essayez de trouver le jour de la semaine avec date('w');
- ✓ Astuce 4 : Créer un tableau de semaine

Résultat attendu : Le 22 janvier 2059, nous serons un mercredi



# Les dates

## Exercice

```
/*
Exercice : Quel jour serons-nous le 22 janvier 2059 ?
Astuce 1 : Utiliser mktime(hour, minute, second, month, day, year);
Astuce 2 : date() peut convertir un mktime en date standard
            date("d-m-Y", mktime());
Astuce 3 : Essayez de trouver le jour de la semaine avec date('w');
Astuce 4 : Créer un tableau de semaine
Résultat attendu : Le 22 janvier 2059, nous serons un mercredi
*/
$jour = mktime(0,0,0,1,22,2059);
$semaine = [" dimanche ", " lundi ", " mardi ", " mercredi ", " jeudi ", "
            vendredi ", " samedi "];
echo "<p>Le      " . date("d M Y ", $jour) .
      " nous serons un      " . $semaine[date('w', $jour)] . "</p>";
```



# Les formulaires

Un formulaire prend plusieurs attributs :

- ✓ Action : va désigner la page qui va traiter notre formulaire
- ✓ Method : permet de choisir le mode de transmission des données du formulaire
  - ✓ GET : les données sont transmises via l'url [www.url.com/?toto=123](http://www.url.com/?toto=123)
  - ✓ POST : les données sont transmises dans la requête côté serveur

```
<form action="traitement.php" method="POST">
```

## ⚠ Rappels

- ✓ Si on ne met pas de page de traitement, la page en cours sera chargée de traiter le formulaire
- ✓ Par défaut la « method » GET est appliquée



# Les formulaires

Faisons quelques traitements pour notre futur formulaire de connexion

## ! Astuces

- ✓ Les fonctions de bases comme `empty()` et `isset()` permettent de savoir si une donnée est vide et définie.
- ✓ N'oubliez pas de définir le « **name** » dans vos formulaires

## ! Rappels

- ✓ Une fois le bouton envoyer cliqué, nos données seront transmises au `traitement.php`
- ✓ Pour accéder aux données lorsque l'on se trouve sur `traitement.php`, nous pouvons utiliser **`$_POST`** ou **`$_GET`** (**variables globale**) en fonction de la méthode



# Les formulaires

Faisons quelques traitements pour notre futur formulaire de connexion

```
<?php

// Traitement de notre formulaire
// On accède aux données via le tableau $_POST
echo '<pre>'; // Balise HTML de présentation
var_dump($_POST); // Debug des données
echo '</pre>';
```

```
array(2) {
    ["email"]=>
    string(19) "maxu@mailinator.com"
    ["password"]=>
    string(9) "Pa$$w0rd!"
}
```



# Les formulaires

Faisons quelques traitements pour notre futur formulaire de connexion

```
// Si la globale $_POST est définie
// Et qu'elle n'est pas vide
if(isset($_POST) && !empty($_POST)){
    // Si l'email est valide
    if(filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)){
        // Si l'email est valide, je test le password
        if(!empty($_POST['password'])){
            // Je redirige vers mon compte
            header('Location: /mon-compte');
        }else{
            //!!\\ Retourne à la page précédente
            header('Location: ' . $_SERVER['HTTP_REFERER']);
        }
    }else{
        //!!\\ Retourne à la page précédente
        header('Location: ' . $_SERVER['HTTP_REFERER']);
    }
}else{
    //!!\\ Retourne à la page précédente
    header('Location: ' . $_SERVER['HTTP_REFERER']);
}
```

## ⚠️ Rappels

- ✓ Ne négligez pas les contrôles
- ✓ Documentez-vous pour appliquer de nouveau contrôle



# Les formulaires

## Exercice formulaire 1

Créez un formulaire d'inscription comme suit et effectuer le traitement sur une page externe pour afficher le récapitulatif des informations saisies dans un tableau HTML.

### Exercice 1

Firstname

Firstname

Email

Password

### Vos informations

exo1-firstname	Brian
exo1-lastname	Phyllis
exo1-email	kyjimokaq@mailinator.com
exo1-password	Pa\$\$w0rd!





# Les formulaires

## TP formulaire pizza maison

Dans cet exercice vous devez créer une application de création de pizza maison.

- ✓ Vous utiliserez la méthode « **GET** ».
- ✓ Le client pourra voir le résultat de sa pizza dans un tableau récapitulatif (liste et total des ingrédients) ainsi que le prix.
- ✓ Si vous connaissez le JavaScript, n'hésitez pas à donner de l'interactivité à votre formulaire
  - ✓ Des images pré découpées sont disponibles ici : <https://rebrand.ly/jytjrb>



# Les sessions

Les sessions vont vous permettre de faire transiter des données d'une page à l'autre pendant toute la durée de la navigation de l'utilisateur.

L'exemple le plus connu est celui de la connexion à un espace administration.

```
<?php session_start(); ?> <!-- Ici démarre la session -->
<!doctype html>
<html lang="en">
```

## ⚠ Rappels

- ✓ **Session\_start()** permet de démarrer la session
- ✓ La session démarre toujours au début du fichier et dans tous les fichiers où on utilise la session
- ✓ Pour accéder aux données stockées on utilise la globale **\$\_SESSION**
- ✓ Pour vider une session on utilise **unset(\$\_SESSION['non de session']);**
- ✓ **\$\_SESSION** est un tableau



## Les sessions



## Les sessions

```
<?php session_start();  
// Traitement de notre formulaire  
// avec session  
if(! empty($_POST)){  
    $_SESSION['user'] = [  
        'pseudo' => $_POST['email'],  
        'password' => $_POST['password']  
    ];  
    header('Location: mon-compte.php');  
}
```



# Les sessions

```
<div class="album py-5 bg-light">
  <div class="container">
    <div class="row">
      <div class="col-md-12">
        <!-- ICI COMMENCE NOTRE PHP -->
        <p>Vos informations : <?= $_SESSION['user']["pseudo"]; ?></p>
      </div>
    </div>
  </div>
</div>
```



# Les cookies

Les cookies vont vous permettre de stocker des informations directement dans navigateur de l'utilisateur. A travers ces informations vous pourrez stocker tout un tas de données (*attention tout de même à ne pas stocker des informations confidentielles, on appliquera les mêmes règles que pour les sessions*).

Dans l'exemple que nous allons produire ensemble, nous allons créer un formulaire de connexion et sauvegarder le choix de l'utilisateur lorsqu'il coche la case « se souvenir de moi » pour l'authentifier de nouveau.

*setcookie() permet de créer un cookie*  
*\$\_COOKIE superglobale*

```
$value = 'Valeur de test';  
setcookie("TestCookie", $value, time()+3600); /* expire dans 1 heure */
```

Nous profiterons aussi de cet exercice pour encoder un mot de passe (hasher un mot de passe).



# Les cookies

```
// J'ai un utilisateur
$user = ['email' => 'j@aime.php', 'password' => 'PhpCfun'];
// J'encode le mot de passe
// Effectuez un var_dump
password_hash($user['password'], PASSWORD_BCRYPT);
// $2y$10$10xKodpAG0xKs6hNdDzFxuuAira4rqZ1D6qZgaaJRnBf7017Pn2Gu

// Je stocke de nouveau le mot de passe dans l'utilisateur
$user['password'] = '$2y$10$10xKodpAG0xKs6hNdDzFxuuAira4rqZ1D6qZgaaJRnBf7017Pn2Gu';
```



# Les cookies

```
require 'bdd.php';
// Traitement de notre formulaire
if(! empty($_POST)){
    // Si le formulaire est transmis je vérifie que l'utilisateur
    // est bien celui de ma bdd.php et si il a coché la case
    // se souvenir de moi, je stocke un cookie sur son navigateur

    // Un mot de passe hashé ne peut pas être décrypté, seulement
    // comparé avec la fonction password_verify()
    // Cette fonction retourne true si le mot de passe est identique
    $password_verify = password_verify($_POST['password'], $user['password']);

    if( $_POST['email'] === $user['email'] && $password_verify ){
        // Si les identifiants sont ok
        if($_POST['rememberme'] === 'rememberme'){
            // Si il a coché la case, j'enregistre son email dans un cookie

            setcookie('rememberme', $user['email'], time()+3600*24*365);
        }
        header('Location: mon-compte.php');
    }else{
        header('Location: index.php');
    }
}
```





# Les bases de données

Une fois les bases de PHP apprissent, viendra le moment où vous souhaitez rendre votre site dynamique notamment en sauvegardant des données. SQL permettra donc de faire cela, PHP peut utiliser plusieurs types de bases de données nativement donc : Oracle, MySQL, Microsoft SQL SERVER...

A travers notre apprentissage, nous utiliserons MySQL qui est le plus connu et le plus rependue.

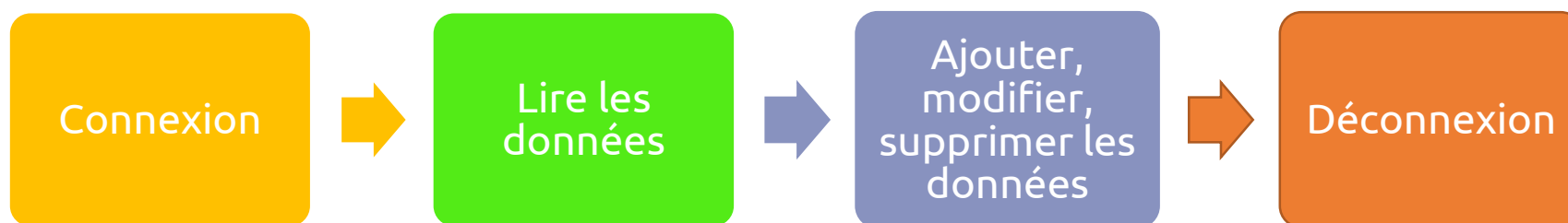
Vous allez pouvoir en autre sauvegarder :

- ✓ Liste d'utilisateurs
- ✓ Liste de produits
- ✓ Liste d'articles de blog
- ✓ Liste de commentaires
- ✓ ...



# Les bases de données

Pour fonctionner avec PHP nous allons procéder de la sorte :



## ! Astuces

- ✓ MySQL est déjà installé sur vos serveurs si vous utilisez l'un des logiciels vus au début



# Les bases de données

## Notions et vocabulaire

**SGBD** : système de gestion de base de données (comme MySQL)

**PhpMyAdmin** : Logiciel interfacé pour gérer une base de données

**HeidiSQL** : Logiciel Windows pour gérer une base de données

**Table** : feuille de stockage présentée en colonnes et lignes

**Select** : commande SQL pour sélectionner des données

**Insert** : commande SQL pour insérer des données

**Delete** : commande SQL pour supprimer des données

**Update** : commande SQL pour mettre à jour des données

**Create** : commande SQL pour créer une base de données

**Drop** : Supprime une base de données

**Where** : extraire des données en suivant une condition

**Orderby** : trie des lignes suivant une condition

**CRUD** : pour create, read, update, delete

**PDO** : PHP Data Objects - Représente une connexion entre PHP et un serveur de base de données

**MySQLi** : Extension PHP pour effectuer connexion entre PHP et un serveur de base de données



# Les types de données

Les types de données les plus courants sont :

- ✓ INT : un nombre positif ou négatif (Quatre octets)
- ✓ BIGINT : un nombre positif ou négatif (Huit octets)
- ✓ SMALLINT : un nombre positif ou négatif (Deux octets)
- ✓ TINYINT : un nombre positif 0 à 255 (Un octet)
  
- ✓ FLOAT : un nombre positif ou négatif avec décimales
  
- ✓ VARCHAR : pour les textes jusqu'à 255 caractères
- ✓ TEXT : pour les textes jusqu'à 65 535 caractères
- ✓ DATE : date au format SQL YYYY-MM-DD
- ✓ DATETIME : date et une heure au format SQL YYYY-MM-DD HH:MM:SS



# Les bases de données

Objectif de ce chapitre :

- ✓ A travers un exercice nous allons parcourir les principales fonctionnalités de MySQL

Brief de l'exercice :

- ✓ Nous allons créer un moteur de recherche de film privé
  - ✓ Pour accéder à la plateforme, il faudra se connecter
  - ✓ On pourra ensuite y voir tous les films sur une page
  - ✓ Voir / Editer un film seul
  - ✓ Ajouter / Supprimer un film
  - ✓ Ajouter un titre, une description et une image à notre film
  - ✓ En dernier nous ferons un champ de recherche qui ira trouver un film

## ⚠ Rappels

- ✓ **Pour plus de rapidité, cet exercice ne couvre pas tous les aspects de sécurité, à vous de prendre en compte cela lors de vos futurs développements**



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

- ✓ Créer un dossier data à la racine de votre projet et créer un fichier data.sql dans ce dossier
- ✓ Ajouter le script suivant dans app/create\_table.php

```
$host      = 'localhost';
$username  = 'root';
$password  = '';
try {
    // PDO est un objet qui prend 3 paramètres minimum
    $connection = new PDO("mysql:host=$host", $username, $password);
    // file_get_contents permet de récupérer et lire le contenu d'un fichier
    $sql = file_get_contents("data/data.sql");
    // exécution de fichier
    $connection->exec($sql);
    echo "Base de données et tables créées !";
} catch(PDOException $error) {
    echo $sql . "<br>" . $error->getMessage();
}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

☑ Créer la requête SQL dans data/data.sql

```
CREATE DATABASE tp_films;
use tp_films;

CREATE TABLE IF NOT EXISTS user (
    id int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    firstname VARCHAR( 255 ) NOT NULL,
    lastname VARCHAR( 255 ) NOT NULL,
    email VARCHAR( 255 ) NOT NULL,
    password VARCHAR( 60 ) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS film (
    id int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR( 255 ) NOT NULL,
    description VARCHAR( 255 ) NOT NULL,
    image VARCHAR( 255 ) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO user(firstname,lastname,email,password)
VALUES ('Jon', 'Snow', 'js@winterfell.com', 'youknownothingjonsnow');
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

☑ Créer la requête SQL

```
CREATE DATABASE tp_films;  
use tp_films;
```





# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

✓ Créer la requête SQL

```
CREATE TABLE IF NOT EXISTS user (  
    id int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    firstname VARCHAR( 255 ) NOT NULL,  
    lastname VARCHAR( 255 ) NOT NULL,  
    email VARCHAR( 255 ) NOT NULL,  
    password VARCHAR( 60 ) NOT NULL,  
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

✓ Créer la requête SQL

```
CREATE TABLE IF NOT EXISTS film (  
    id int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    title VARCHAR( 255 ) NOT NULL,  
    description VARCHAR( 255 ) NOT NULL,  
    image VARCHAR( 255 ) NOT NULL,  
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 1 : Connexion avec PDO et création de la base de données ainsi que les tables USER, FILM

✓ Créer la requête SQL

```
INSERT INTO user(firstname,lastname,email,password)
VALUES ('Jon', 'Snow', 'js@winterfell.com', 'youknownothingjonsnow');
```




# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 2 : Vérifier la connexion grâce à PDO

✓ **Nous avons maintenant un utilisateur dans notre BDD**

 id	firstname	lastname	email	password	created_at
1	Jon	Snow	js@winterfell.com	youknownothingjonsnow	2019-05-08 22:25:48



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 2 : Vérifier la connexion grâce à PDO

- ✓ Nous avons maintenant un utilisateur dans notre BDD
- ✓ **Créons un connecteur BDD avec PDO**

```
<?php
// Connexion à la base de données
// Nous utiliserons aussi le gestionnaire d'erreur
// C'est une bonne pratique d'utiliser le try et catch

$dbname = 'tp_films';
$user   = 'root';
$pass   = '';

try {
    $connexion = new PDO('mysql:host=localhost;dbname='.$dbname, $user, $pass);
} catch (PDOException $e) {
    print "Erreur !: " . $e->getMessage() . "<br/>";
    die();
}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 2 : Vérifier la connexion grâce à PDO

- ✓ Nous avons maintenant un utilisateur dans notre BDD
- ✓ Créons un connecteur BDD avec PDO
- ✓ **Essayons de connecter l'utilisateur en vérifiant qu'il existe**

```
require_once 'bdd.php';  
// Je créer une requette pour vérifier si l'utilisateur existe  
// Je prépare la requette  
$sql = $connexion->prepare("SELECT * FROM user WHERE email=:email AND password=  
    :password");  
// J'exécute la requette  
$sql->execute(  
    [  
        'email' => $_POST["email"],  
        'password' => $_POST["password"]  
    ]  
);  
// Je récupère le résultat  
$user = $sql->fetch();
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 2 : Vérifier la connexion grâce à PDO

- ✓ Nous avons maintenant un utilisateur dans notre BDD
- ✓ Créons un connecteur BDD avec PDO
- ✓ Essayons de connecter l'utilisateur en vérifiant qu'il existe
- ✓ **Nous aurons besoin de démarrer une session partout où l'utilisateur devra être connecté pour voir le contenu y compris sur la page de connexion**

```
<?php session_start(); ?>
<!doctype html>
<html lang="en">
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 3 : Affichage de notification grâce aux sessions, il faut créer un fichier notifications.php

```
<!-- Ici nous ajoutons une notification lorsqu'on a une donnée
dans le tableau $_SESSION -->

<?php if( !empty($_SESSION) ) : ?>

    <?php foreach($_SESSION as $k => $v) :?>
        <div class="toast" role="alert" aria-live="assertive" aria-atomic="true" data-autohide="
false" style="position: absolute; top: 0; right: 20px;">
            <div class="toast-header">
                <strong class="mr-auto"><?= $k; ?></strong>
                <button type="button" class="ml-2 mb-1 close" data-dismiss="toast" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="toast-body">
                <?= $v; ?>
            </div>
        </div>
    <?php endforeach; ?>
    <?php // Je vide les sessions j'en créer une autre
        unset($_SESSION['success']);
        unset($_SESSION['errors']); ?>
<?php endif; ?>
```





# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 3.1 : J'appelle le fichier pour afficher les notifications sur les pages dont j'ai besoin de notification

```
<header>
  <?php require_once 'notifications.php'; ?>
</header>
```

```
<script>
$(document).ready(function(){
  $('.toast').toast('show');
});
</script>
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 4 : Je sécurise l'accès depuis le fichier connect.php

```
// Je donne la permission d'accéder à mon utilisateur  
$_SESSION['access'] = true;  
header('Location: ../list-films.php');
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 4.1 : Je crée un fichier app/can-connect.php

```
<?php
// J'autorise l'utilisateur à voir le contenu
// Sinon je le renvoi à la page connexion

if(empty($_SESSION)){

    if( !isset($_SESSION['access']) ) {
        //!\ Retourne à la page précédente
        header('Location: /15_TP_MySQL_liste_films_correction');
    }

}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 4.2 : Je crée un script de déconnexion app/disconnect.php et j'ajoute un bouton dans le header

```
<?php session_start();  
foreach ($_SESSION as $key => $value) {  
    unset($_SESSION[$key]);  
}  
header('Location: /15_TP_MySQL_liste_films_correction');
```

```
<a href="/15_TP_MySQL_liste_films_correction/app/disconnect.php" class="btn btn-danger">Déconnexion</a>
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5 : Ajouter un film (add.php et app/add.php)

```
<?php session_start();  
// Ajout d'un film  
//$_FILES contient les informations du ou des fichiers envoyés  
var_dump($_FILES);  
  
if(!empty($_FILES)){  
    //je vérifie que le transfert s'est bien passé  
    if($_FILES['image']['error'] === 0){  
        /*  
        Error renvoi 0 si il n'y a pas d'erreur  
        1 ko = 1024 octets  
        1 Mo = 1 048 576 octets  
        */  
    }  
    else{  
        $_SESSION['errors'] = 'erreur lors du transfert';  
        header('Location: ../list-films.php');  
    }  
}
```

## ⚠ Rappels

- ✓ Pour transférer des fichiers via un formulaire votre form doit être au format  
enctype="multipart/form-data"



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.1 : Ajouter un film (add.php et app/add.php)

```
// Je limite les fichiers à 1000Ko
$maxSize = 1000 * 1024; //1 048 576 octets
if($_FILES['image']['size'] <= $maxSize){
}
else{
    $_SESSION['errors'] = 'fichier trop gros';
    header('Location: ../list-films.php');
}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.2 : Ajouter un film (add.php et app/add.php)

```
// Je limite les fichiers à 1000Ko
$maxSize = 1000 * 1024; //1 048 576 octets
if($_FILES['image']['size'] <= $maxSize){
    // La taille du fichier est inférieur ou égale

    //je vérifie l'extension du fichier
    $fileInfo = pathinfo($_FILES['image']['name']);
    //var_dump($fileInfo);
    $extension = strtolower($fileInfo['extension']);

    $extensions_autorisees = ['jpg', 'jpeg', 'png', 'svg', 'gif'];

    // Je vérifie si l'extension est ok
    if(in_array($extension, $extensions_autorisees)){ ...
    }
    else{
        $_SESSION['errors'] = 'mauvaise extension';
        header('Location: ../list-films.php');
    }
}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3 : Ajouter un film (add.php et app/add.php)

```
// Je vérifie si l'extension est ok
if(in_array($extension, $extensions_autorisees)){
    // Astuce pour créer un nom unique
    $image_name = md5(uniqid(rand(), true));
    echo $image_name;
```





# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.1 : Ajouter un film (add.php et app/add.php)

```
/*
    Création des miniatures 150px
    On utilise imagecreatefrom...
*/

$config_minature_width = 150;

if($extension == 'jpg' || $extension === 'jpeg'){
    $new_image = imagecreatefromjpeg($_FILES['image']['tmp_name']);
}
elseif($extension === 'gif'){
    $new_image = imagecreatefromgif($_FILES['image']['tmp_name']);
}
elseif($extension === 'png'){
    $new_image = imagecreatefrompng($_FILES['image']['tmp_name']);
}
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.2 : Ajouter un film (add.php et app/add.php)

```
// largeur originale (en px)
$original_width = imagesx($new_image);

// hauteur originale
$original_height = imagesy($new_image);

// calcul de la nouvelle hauteur
$miniature_height = ($original_height * $config_miniature_width) / $original_width;

//je crée une nouvelle image avec les dimensions (nouvelle largeur et nouvelle hauteur)
$miniature = imagecreatetruecolor($config_miniature_width, $miniature_height);
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.3 : Ajouter un film (add.php et app/add.php)

```
// on remplit la miniature à partir de l'image originale
imagecopyresampled($miniature,
                   $new_image,
                   0, 0, 0, 0,
                   $config_miniature_width,
                   $miniature_height,
                   $original_width,
                   $original_height);
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.4 : Ajouter un film (add.php et app/add.php)

```
// chemin vers le dossier où je stocke mes miniatures
$folder = '../images/miniatures/';

if($extension === 'jpg' || $extension === 'jpeg')
{
    imagejpeg($miniature, $folder . $image_name . '.' . $extension);
}

elseif($extension === 'png')
{
    imagepng($miniature, $folder . $image_name . '.' . $extension);
}

elseif($extension === 'gif')
{
    imagegif($miniature, $folder . $image_name . '.' . $extension);
}
```

## ⚠️ Rappels

- 👉 Pensez à créer les dossiers avant d'exécuter le script



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.5 : Ajouter un film (add.php et app/add.php)

```
// je peux transférer le fichier sur le serveur  
move_uploaded_file($_FILES['image']['tmp_name'], '../images/' .  
    $image_name . '.' . $extension);
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.5 : Ajouter un film (add.php et app/add.php)

```
// j'ouvre une connexion et j'enregistre les infos
require_once('bdd.php');
// préparation de la requête
$sql = $connexion->prepare('
    INSERT INTO film (title, description, image)
    VALUES(:title, :description, :image)');
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.5.1 : Ajouter un film (add.php et app/add.php)

```
// protection des données enregistrées  
$sql->bindValue(':title', $_POST['title'], PDO::PARAM_STR);  
$sql->bindValue(':description', $_POST['description'], PDO::PARAM_STR);  
$sql->bindValue(':image', $image_name . '.' . $extension, PDO::PARAM_STR);
```



# Les bases de données

Vous aurez besoin de récupérer la base HTML à cette adresse : <https://rebrand.ly/jp1xo0>

Étape 5.3.5.2 : Ajouter un film (add.php et app/add.php)

```
// on exécute la requête
$result = $sql->execute();

if($result){
    $_SESSION['success'] = 'Film ajouté';
    header('Location: ../list-films.php');
}
else{
    $_SESSION['errors'] = 'erreur lors de l\'ajout';
    header('Location: ../list-films.php');
}
```





# Rappels

Nous allons revoir ici les bases des requêtes

Connexion à une base de données avec PDO : <https://www.php.net/manual/fr/pdo.connections.php>

```
<?php
//On lance une connexion à la base de données
$dbh = new PDO('mysql:host=localhost;dbname=revisions_sql', 'root', '');
```



# Rappels

Nous allons revoir ici les bases des requêtes

Insertion de données avec : INSERT

Prépare la requête

```
$req = $dbh->prepare("SELECT * FROM user WHERE lastname = :lastname");  
$req->bindValue(':lastname', $_GET['lastname'], PDO::PARAM_STR);  
$req->execute();  
  
// $resultat = $req->fetch(); // Récupère le premier élément  
$resultat = $req->fetchAll(PDO::FETCH_OBJ); // Récupère tous les éléments
```

Bind : Associe une valeur à un paramètre et le sécurise

Récupère le ou les éléments



# Rappels

Nous allons revoir ici les bases des requêtes

La méthode `fetch()` et `fetchAll()`

```
$resultat = $req->fetch(); // Récupère le premier élément  
$resultat = $req->fetchAll(); // Récupère tous les éléments
```

Retourne un tableau indexé par les noms de colonnes et aussi par les numéros de colonnes, commençant à l'index 0, comme retournés dans le jeu de résultats



# Rappels

Nous allons revoir ici les bases des requêtes

La méthode `fetch()` et `fetchAll()`

`Fetch` et `fetchAll` peuvent prendre un paramètre pour retourner un tableau sous différent format, cela sera très pratique lorsque l'on aura besoin d'un format particulier de données par exemple les objets.

***PDO::FETCH\_BOTH (défaut)***: retourne un tableau indexé par les noms de colonnes et aussi par les numéros de colonnes, commençant à l'index 0, comme retournés dans le jeu de résultats

***PDO::FETCH\_OBJ***: retourne un objet anonyme avec les noms de propriétés qui correspondent aux noms des colonnes retournés dans le jeu de résultats

***PDO::FETCH\_ASSOC***: retourne un tableau indexé par le nom de la colonne comme retourné dans le jeu de résultats

```
$resultat = $req->fetchAll(PDO::FETCH_OBJ); // Récupère tous les éléments
```



# Rappels

Nous allons revoir ici les bases des requêtes

Insertion de données avec : INSERT

Tableau de données

```
$array = [  
    'firstname' => 'Marco',  
    'lastname' => 'Polo',  
    'email' => 'mp@mp.com'  
];  
$insert = $dbh->prepare("INSERT INTO user (firstname, lastname, email)  
                        VALUES (:firstname, :lastname, :email)");  
$insert->bindValue(':firstname', $array['firstname'], PDO::PARAM_STR);  
$insert->bindValue(':lastname', $array['lastname'], PDO::PARAM_STR);  
$insert->bindValue(':email', $array['email'], PDO::PARAM_STR);  
$insert->execute();  
$dbh->lastInsertId(); // Récupère le dernier élément enregistré par son ID
```

Prépare la requête

Bind : Associe une valeur à un paramètre et le sécurise

Récupère le dernier élément inséré



# Rappels

Nous allons revoir ici les bases des requêtes

Insertion de données avec : UPDATE

Tableau de données

Prépare la requête

```
// Modification de données

$array['firstname'] = 'Marc';
$update = $dbh->prepare("UPDATE user
                        SET firstname = :new_firstname
                        WHERE id = :id");
$update->bindValue(':new_firstname', $array['firstname'], PDO::PARAM_STR);
$update->bindValue(':id', 6, PDO::PARAM_INT);
$update->execute();
```

Bind : Associe une valeur à un paramètre et le sécurise

Exécute la requête



# Rappels

Nous allons revoir ici les bases des requêtes

Insertion de données avec : DELETE

```
// Suppression d'une donnée  
$delete = $dbh->prepare("DELETE FROM user WHERE id = :id");  
$delete->bindValue(':id', 6, PDO::PARAM_INT);  
$delete->execute();
```

Prépare la requête

Bind : Associe une valeur à un paramètre et le sécurise

Exécute la requête

