



Les bases du web

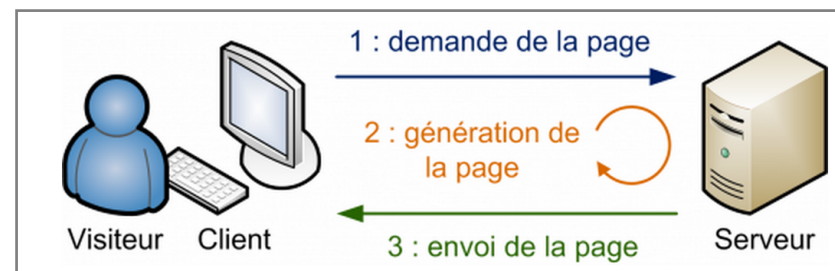
Le langage PHP

- Langage serveur
- Permet de créer des pages web dynamiques
- version 8

site statique (pages html)

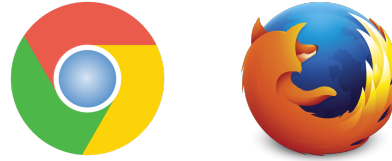


site dynamique (pages php)



Préparer son ordinateur

- Un navigateur
 - Chrome ou Firefox



- Editeur de texte
 - SublimeText



- Serveur local MAMP
 - Serveur Apache
 - Serveur PHP
 - Serveur SQL



Premiers pas avec PHP

- Langage serveur
- Permet d'écrire du html. Tout le code php reste sur le serveur web. Seul ce qui est imprimé est visible dans la page web.
- Permet de consulter, modifier une base de données.

Premiers pas avec PHP

- Extension **.php** et balise php

<?php //code php// **?>**

- Ecrire du contenu dans des balises html

<p><?php echo "Texte"; ?></p>

<p><?= "Texte"; ?></p>

- Ecrire du contenu html

echo "<p>Texte.</p>";

Les variables

- Déclaration et assignation de valeur

\$variable = value;

Plusieurs types de valeurs

String : chaîne de caractères

Integer : nombre entier

Float : chiffres à virgule

Boolean : true/false

Array : tableau

Object : objet

NULL : aucune valeur définie

- Concaténation

echo 'Le visiteur a ' . \$age . ' ans';

echo "Le visiteur a \$age ans"; // Plus simple avec les guillemets doubles.

Les conditions

```
if(condition) { ... }  
else{ ... }
```

Les opérateurs de comparaison

```
==    >=    <=    !=    ===    !==
```

Les opérateurs logiques

```
&&    ||
```

Le cas du booléen

```
if($value){ ... }
```

Les conditions

L'opérateur ternaire

condition ? valeur_si_true : valeur_si_false

```
$egal = ($val1 == $val2) ? 'oui' : 'non';
```

L'alternative

```
switch($value)
```

```
{
```

```
  case 1 :
```

```
    echo "Un";
```

```
    break;
```

```
  case 2 :
```

```
    echo "Deux";
```

```
    break;
```

```
  default :
```

```
    echo "Autre valeur";
```

```
}
```


Les tableaux

```
$array = [valeur1, valeur2, valeur3];
```

```
echo $array[0]; // renvoie valeur1
```

```
$array = [  
    $key1 => $value1,  
    $key2 => $value2  
];
```

Les tableaux

Parcourir un tableau

```
foreach($array as $key => $value){ ... }
```

Des fonctions spécifiques

`in_array()`

`array_key_exists()`

Les boucles

while (condition){ // code à exécuter // }

for(\$i = 0 ; \$i < 10 ; \$i ++){ // code à exécuter // }

foreach()

L'inclusion

Permet la factorisation du code

include

```
include("header.php");
```

Les fonctions

- Déclaration d'une fonction

```
function myFunction($arg) {  
    ...  
    return $val;  
}
```

- Appel d'une fonction

```
myFunction($arg);
```

- De nombreuses fonctions prêtes à l'emploi

- strlen(), trim(), str_replace(), date() ...

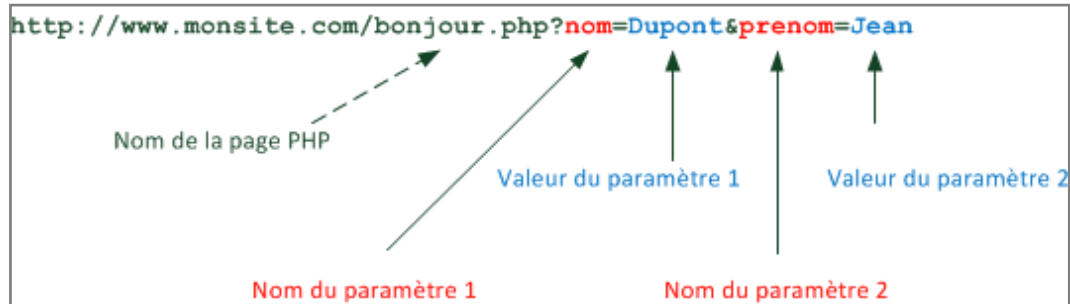
Les variables superglobales

Majuscules

Commence par \$_

\$_GET, \$_POST, \$_COOKIE, \$_SESSION, \$_FILES, \$_SERVER...

Envoyer des données via l'url



`$_GET['nom']`

`$_GET['prenom']`

Sécurité

`is_numeric()`, `isset()`, `empty()`...

Envoyer des données via un formulaire

```
1 <form action="cible_envoi.php" method="post" enctype="multipart/form-data">
2     <p>
3         Formulaire d'envoi de fichier :<br />
4         <input type="file" name="monfichier" /><br />
5         <input type="submit" value="Envoyer le fichier" />
6     </p>
7 </form>
```

```
<form action="cible.php" method="post">
  <input type="text" name="nom" value="" placeholder="Votre nom">
  <input type="submit">
</form>
```

\$_POST et \$_FILES

Sécurité, faille XSS : htmlspecialchars()

Envoyer des données via un formulaire

Paramètres POST

Envoyés par un formulaire avec l'attribut **method="post"**

Pas visibles dans l'url (ne demeurent pas dans l'historique)

`$_POST['nom']`

Paramètres FILES

Envoyés par un formulaire avec l'attribut **enctype="multipart/form-data"**

Contiennent les fichiers envoyés avec le formulaire.

`<input type="file">`

`$_FILES['monFichier']`

\$_SESSION

- Les sessions

Permet de stocker une variable sur toutes les pages du site.

L'utilisateur n'y a pas accès.

Numéro unique dans le navigateur en liaison avec le serveur

session_start();

Fonction appelée avant tout rendu html.

session_destroy();

`$_SESSION['prenom'] = 'Jean';`

\$_COOKIE

- Les cookies

Stocké du côté utilisateur (sécurité)

```
echo $_COOKIE['pseudo'];
```

```
setcookie('pseudo', 'Utilisateur123', time() + 365*24*3600);
```

Fonction appelée avant la balise <html> du document.

Fonction complète : `bool setcookie (string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false [, bool $httponly = false]]]]])`

Note :

Pour supprimer un cookie, on lui donne une date d'expiration dans le passé.

Ex : `setcookie('pseudo', 'Utilisateur123', time() - 3600);`

Redirection php

Permet d'afficher une page web.

```
header('location: maPage.php');  
exit;
```

Fonction appelée avant tout rendu html.

Lire et écrire dans un fichier

Autoriser l'écriture

ouverture du fichier `$monfichier = fopen('fichier.txt', 'a+')`

lecture de la première ligne `fgets($monfichier);`

déplacer le curseur : `fseek($monfichier, 0);`

écriture : `fputs($monfichier, 'texte');`

fermeture du fichier : `fclose($monfichier);`

Encore plus simple : `file_put_contents('mon_fichier.txt', $output)`

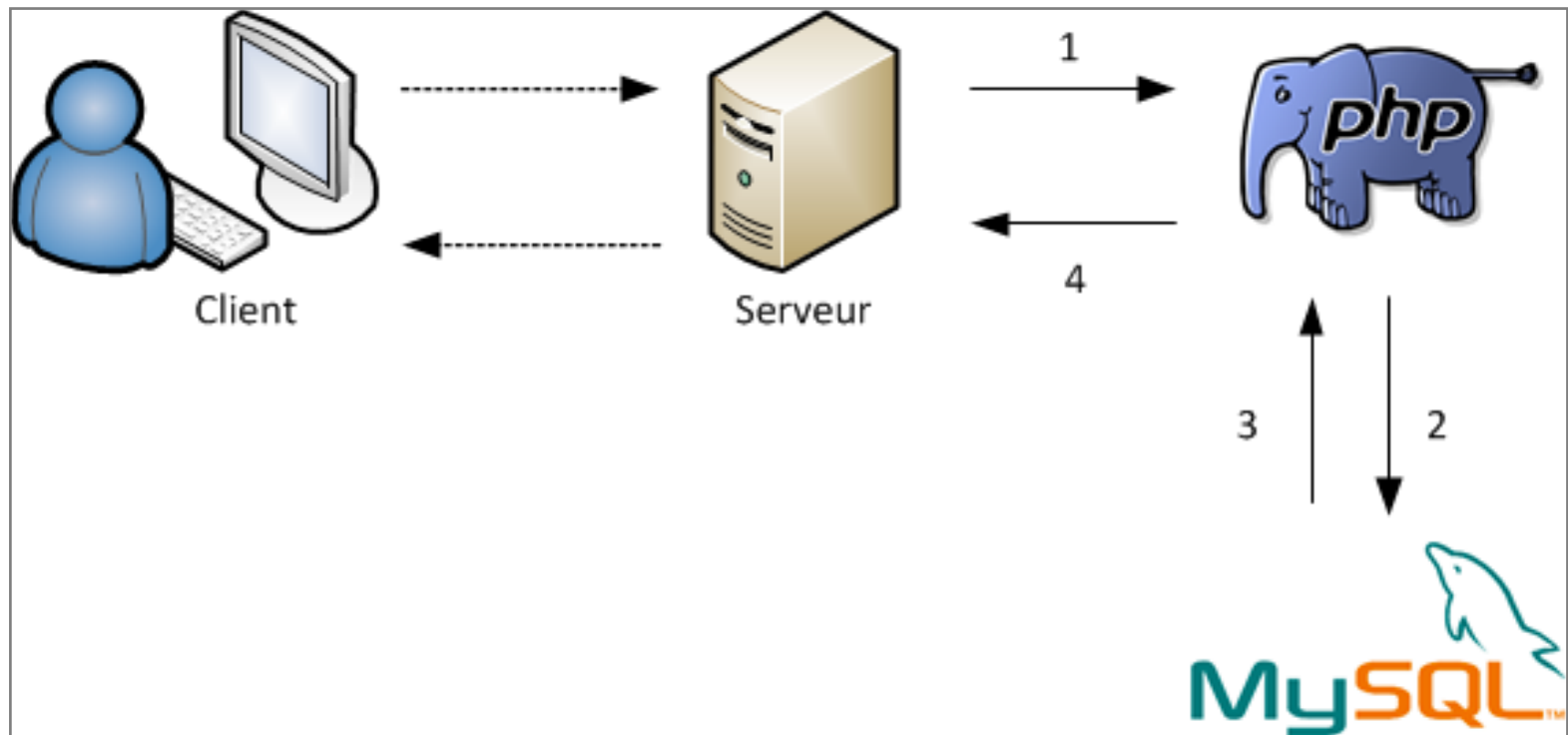
Insertion en base de données

- Les SGBD
- Phpmyadmin
- Lire des données
- Écrire des données
- Les fonctions SQL
- Les date en SQL
- Les jointures

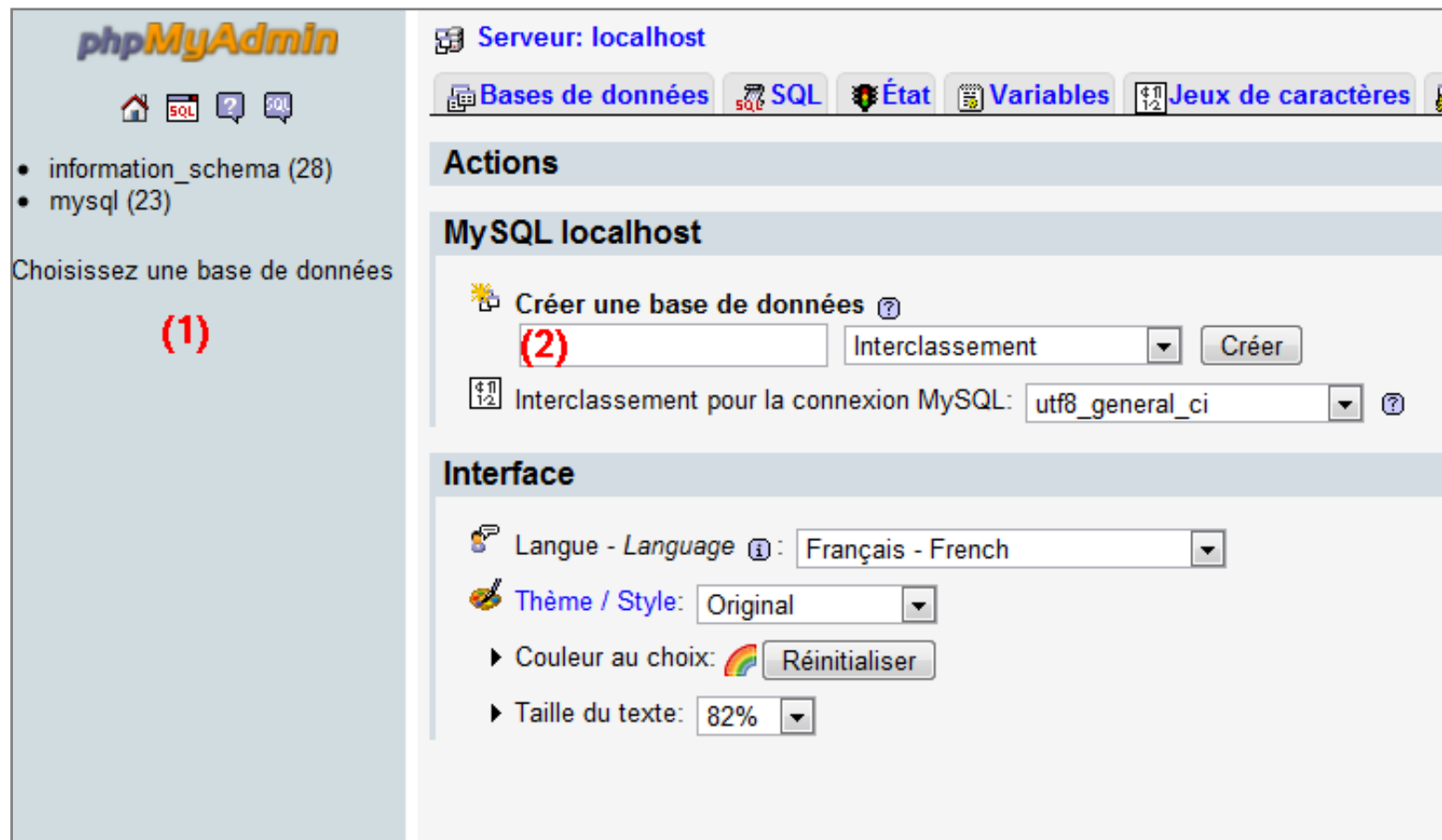
Bases de données / les SGBD

- Mysql : libre et gratuit, le SGBD le plus connu
- **postgresql** : libre et gratuit, plus de fonctionnalités mais moins connu
- **sQlite** : libre et gratuit mais peu de fonctionnalités
- **Oracle** : Le plus complet mais le plus cher
- **Microsoft SQL** serveur : Microsoft ...

Les SGBD



PhpMyAdmin




The screenshot displays the PhpMyAdmin interface. On the left sidebar, the 'phpMyAdmin' logo is at the top, followed by navigation icons (home, SQL, help, etc.) and a list of databases: 'information_schema (28)' and 'mysql (23)'. Below this list is the text 'Choisissez une base de données' and a red '(1)' indicating the database selection area.


The main content area is titled 'Serveur: localhost' and contains a navigation bar with tabs: 'Bases de données', 'SQL', 'État', 'Variables', and 'Jeux de caractères'. The 'Bases de données' tab is active, showing the 'MySQL localhost' section. This section includes a 'Créer une base de données' option with a red '(2)' marking the input field for the database name. Below this, there is a dropdown menu for 'Interclassement' and a 'Créer' button. Further down, there is a section for 'Interclassement pour la connexion MySQL' with a dropdown menu set to 'utf8_general_ci' and a help icon.

The 'Interface' section at the bottom allows users to configure their viewing preferences, including 'Langue - Language' (set to 'Français - French'), 'Thème / Style' (set to 'Original'), 'Couleur au choix' (with a 'Réinitialiser' button), and 'Taille du texte' (set to '82%').

PhpMyAdmin

 Créer une nouvelle table sur la base [test](#)

Nom: Nombre de champs:

Champ	<input type="text" value="id"/>	<input type="text" value="titre"/>	<input type="text" value="contenu"/>
Type 	<input type="text" value="INT"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="TEXT"/>
Taille/Valeurs* ¹	<input type="text"/>	<input type="text" value="255"/>	<input type="text"/>
Défaut ²	<input type="text" value="Aucun"/>	<input type="text" value="Aucun"/>	<input type="text" value="Aucun"/>
Interclassement	<input type="text"/>	<input type="text"/>	<input type="text"/>
Attributs	<input type="text"/>	<input type="text"/>	<input type="text"/>
Null	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index	<input type="text" value="PRIMARY"/>	<input type="text" value="---"/>	<input type="text" value="---"/>
AUTO_INCREMENT	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Commentaires	<input type="text"/>	<input type="text"/>	<input type="text"/>


















PhpMyAdmin







- **Champ** : permet de définir le nom du champ (très important !) ;
- **Type** : le type de données que va stocker le champ (nombre entier, texte, date...) ;
- **Taille/Valeurs** : permet d'indiquer la taille maximale du champ, utile pour le type VARCHAR notamment, afin de limiter le nombre de caractères autorisés ;
- **Index** : active l'indexation du champ. Ce mot barbare signifie dans les grandes lignes que votre champ sera adapté aux recherches. Le plus souvent, on utilise l'index PRIMARY sur les champs de type id ;
- **AUTO_INCREMENT** : permet au champ de s'incrémenter tout seul à chaque nouvelle entrée. On l'utilise fréquemment sur les champs de type id.

PhpMyAdmin

Serveur: localhost ► Base de données: test ► Table: news

[Afficher](#) [Structure](#) [SQL](#) [Rechercher](#) [Insérer](#) [Exporter](#) [Importer](#) [Opérations](#) [Vider](#) [Supprimer](#)

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/>	id	int(11)			Non	Aucun	auto_increment	     
<input type="checkbox"/>	titre	varchar(255)	latin1_swedish_ci		Non	Aucun		     
<input type="checkbox"/>	contenu	text	latin1_swedish_ci		Non	Aucun		     

↑ [Tout cocher](#) / [Tout décocher](#) Pour la sélection :      

[Version imprimable](#) [Suggérer des optimisations quant à la structure de la table](#) ?







[Ajouter](#) 1 champ(s) ☒ En fin de table ☐ En début de table ☐ Après id [Exécuter](#)

Champ	Type	Fonction	Null	Valeur
id	int(11)	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>
titre	varchar(255)	<input type="text"/>	<input type="checkbox"/>	Ma première news
contenu	text	<input type="text"/>	<input type="checkbox"/>	<div>Vous êtes en train de lire ma première news. Bravo !</div>

[Exécuter](#)

☒ Ignorer

Champ	Type	Fonction	Null	Valeur
id	int(11)	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>
titre	varchar(255)	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>
contenu	text	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>

	id	titre	contenu
<input type="checkbox"/>  	1	Ma première news	Vous êtes en train de lire ma première news. Bravo...
<input type="checkbox"/>  	2	Autre news	Ceci est une autre news !
<input type="checkbox"/>  	3	Exclusif !	Ceci est une news !

Lire des données

Se connecter à la base de données

PDO : un outil complet compatible avec tous les SGBD

`$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', 'root');`

```
1 <?php
2 try
3 {
4     $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', '');
5 }
6 catch (Exception $e)
7 {
8     die('Erreur : ' . $e->getMessage());
9 }
10 ?>
```

Lire des données

Traquer les erreurs

Fatal error: Call to a member function fetch() on a non-object in ... on line 13

```
$bdd = new PDO('mysql:host=localhost;dbname=test', 'root', 'root',  
array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION));
```

Lire des données

```
$reponse = $bdd->query('Tapez votre requête SQL ici');
```

Exemple de requête : 'SELECT colonne1, colonne2 FROM table'

Exemple d'utilisation de la réponse à la requête :

```
$donnees = $reponse->fetchAll(PDO::FETCH_ASSOC);
```

```
$reponse->closeCursor();
```

Les critères de sélection

WHERE, AND, OR

ORDER BY

LIMIT 0, 10

Lire des données

La mauvaise idée : concaténer une variable dans une requête.

Risque d'**injection SQL**

Préparation d'une requête :

```
$req = $bdd->prepare('SELECT * FROM table WHERE colonne1 = ?');
```

```
$req->execute(array($_GET['name']));
```

```
$donnees = $req->fetchAll();
```

```
$req->closeCursor()...
```

Préparation avec des marqueurs nominatifs :

```
$req = $bdd->prepare('SELECT * FROM table WHERE name = :name AND surname = :surname');
```

```
$req->execute(array('name' => $_GET['name'], 'surname' => $_GET['surname']));
```

Ecrire des données

INSERT INTO table (prenom, nom) **VALUES** ('Paul', 'Dupont');

UPDATE users **SET** age=30 **WHERE** prenom='Paul'

DELETE FROM users **WHERE** prenom='Paul'

\$bdd->exec('ma requête SQL');

Les fonctions SQL

Fonctions scalaires : transformation

exemple : UPPER(), ROUND(), ...

Fonctions d'agrégat : calcul sur un ensemble

exemple : AVG(), SUM(), ...

Les dates en SQL

- **DATE** : stocke une date au format AAAA-MM-JJ (Année-Mois-Jour) ;
- **TIME** : stocke un moment au format HH:MM:SS (Heures:Minutes:Secondes) ;
- **DATETIME** : stocke la combinaison d'une date et d'un moment de la journée au format AAAA-MM-JJ HH:MM:SS. Ce type de champ est donc plus précis ;
- **TIMESTAMP** : stocke une date et un moment sous le format AAAAMMJJHHMMSS ;
- **YEAR** : stocke une année, soit au format AA, soit au format AAAA.
- FONCTIONS :
 - NOW()
 - DAY(), MONTH(), YEAR() : extraire le jour, le mois ou l'année
 - DATE_FORMAT(date, '%d/%m/%Y %Hh%imin%ss') AS date_formatée

Les jointures

- Liaison de deux tables

- table1 **INNER JOIN** table2 ON table1.id=table2.id
- table1 **LEFT JOIN** table2 ON table1.id=table2.id
- LEFT JOIN : même si aucun résultat

ID	prenom	nom	tel
1	Florent	Dugommier	01 44 77 21 33
2	Patrick	Lejeune	03 22 17 41 22
3	Michel	Doussand	04 11 78 02 00

ID	nom	ID_proprietaire	console	prix	nbre_joueurs_max
1	Super Mario Bros	1	NES	4	1
2	Sonic	2	Megadrive	2	1
3	Zelda : ocarina of time	1	Nintendo 64	15	1
4	Mario Kart 64	1	Nintendo 64	25	4
5	Super Smash Bros Melee	3	GameCube	55	4

Utilisation avancée

- Créer des images
- Les expressions régulières
- La programmation orientée objet
- Architecture MVC

Créer des images

- Activer la bibliothèque GD
- le header
 - `header ("Content-type: image/png");`
- Création de l'image de base
 - `$image = imagecreate(200,50);`
 - `$image = imagecreatefromjpeg('couchersoleil.jpg');`
 - `$image = imagecreatetruecolor (200,50);`
- Affichage de l'image
 - `imagepng($image);` ou `imagepng($image, 'path to save/image.png');`
 - ``

Créer des images

Création des couleurs

- `$orange = imagecolorallocate($image, 255, 128, 0);`
- `$bleu = imagecolorallocate($image, 0, 0, 255);`
- `$bleuclair = imagecolorallocate($image, 156, 227, 254);`
- `$noir = imagecolorallocate($image, 0, 0, 0);`
- `$blanc = imagecolorallocate($image, 255, 255, 255);`
- Le premier `imagecolorallocate` est la couleur de fond

Créer des images

Création de texte

- `imagestring($image, $police, $x, $y, $texte_a_ecrire, $couleur);`
- `imagestringup($image, $police, $x, $y, $texte_a_ecrire, $couleur);`
pour écrire verticalement.
Avec `$police` : entre 1 et 5

Il est conseillé d'utiliser la fonction **`imagettftext()`** qui a un meilleur rendu visuel mais n'utilise que des polices ttf :

`imagettftext($img , $taille , $angle , $x_debut , $y_debut , $couleur , $police , $str)`

Créer des images

Création d'une forme

- ImageSetPixel (\$image, \$x, \$y, \$couleur);
- ImageLine (\$image, \$x1, \$y1, \$x2, \$y2, \$couleur);
- ImageEllipse (\$image, \$x, \$y, \$largeur, \$hauteur, \$couleur);
- ImageRectangle (\$image, \$x1, \$y1, \$x2, \$y2, \$couleur);
- ImagePolygon (\$image, \$array_points, \$nombre_de_points, \$couleur);

Créer des images

Transformer une couleur en transparent

- `imagecolortransparent($image, $couleur);`

Placer une image par dessus

- `$source = imagecreatefrompng(« logo.png »); $destination = imagecreatefromjpeg(« couchersoleil.jpg »); $largeur_source = imagesx($source);`
- `imagecopymerge($destination, $source, $destination_x, $destination_y, 0, 0, $largeur_source, $hauteur_source, 60);`
- `imagejpeg($destination);`

Les expressions régulières

ou « REGEX »

Vérifier une adresse email, un mot de passe, ...

preg_match() -> Est ce que cela contient un **pattern**

preg_replace() -> Remplace un pattern

Les expressions régulières

Recherche simple :

- `#Ma regex#options`
- Le `#` permet de délimiter la regex et d'y ajouter des options
- Options : `i` -> insensible à la casse

Chaîne	Regex	Résultat
J'aime jouer de la guitare	<code>#Guitare#i</code>	VRAI
Vive la GUITARE !	<code>#guitare#i</code>	VRAI
Vive la GUITARE !	<code>#guitare#</code>	FAUX

Les expressions régulières

Le symbole OU -> | (le pipe)

Chaîne	Regex	Résultat
J'aime jouer de la guitare.	<code>#guitare piano#</code>	VRAI
J'aime jouer du piano.	<code>#guitare piano#</code>	VRAI
J'aime jouer du banjo.	<code>#guitare piano#</code>	FAUX
J'aime jouer du banjo.	<code>#guitare piano banjo#</code>	VRAI

Les expressions régulières

Début et fin de chaîne

- Commence par : ^
- Termine par : \$

Chaîne	Regex	Résultat
Bonjour petit dev	#^Bonjour#	VRAI
Bonjour petit dev	#dev\$#	VRAI
Bonjour petit dev	#^dev#	FAUX
Bonjour petit dev !!!	#dev\$#	FAUX

Les expressions régulières

Les classes de caractères

- `#gr[io]s#`
 - entre crochet, une des lettres peut convenir
 - donc on peut avoir gris ou gros
- `#gr[a-f]s#`
 - le tiret autorise une plage de caractère
 - donc on peut avoir gras, grbs, grcs, grds, ..., grfs.
- `#[A-Z0-9]#`
- `#[^0-9]#` -> Je ne veux pas de numérique

Les expressions régulières

Chaîne	Regex
Cette phrase contient autre chose que des chiffres	<code>#[^0-9]#</code>
cette phrase contient autre chose que des majuscules et des chiffres	<code>#[^A-Z0-9]#</code>
Cette phrase ne commence pas par une minuscule	<code>#^[^a-z]#</code>
Cette phrase ne se termine pas par une voyelle	<code>#[^aeiouy]\$#</code>
ScrrmmmblllGnnngngnngnMmmmmffff	<code>#[^aeiouy]#</code>

Les expressions régulières

Les quantificateurs :

- ▣ ? -> 0 ou 1
- ▣ + -> 1 ou plus
- ▣ * -> 0 ou plus

Chaîne	Regex	Résultat
eeeeee	#e+#	VRAI
ooo	#u?#	VRAI
magnifique	#[0-9]+#	FAUX
Yahoooooooo	#^Yaho+\$#	VRAI
Yahoooooooo c'est génial !	#^Yaho+\$#	FAUX
Blablablablabla	#^Bla(bla)*\$#	VRAI

#bor?is# -> bois ou boris

#Ay(ay)*# -> Ay ou Ayay ou Ayayay ou Ayayayay ...

Les expressions régulières

Les quantificateurs plus précis :

- $\{3\}$ -> 3 fois
- $\{3,5\}$ -> 3 à 5 fois
- $\{3,\}$ -> 3 fois ou plus

Ecrire les équivalents de ? + et *

$\{0,1\}$

$\{0,\}$

Les expressions régulières

- Les métacaractères

^ \$ () [] { } ? + * . \ |

#Quoi \?# -> Quoi ?

#Quoi ?# ->

Les expressions régulières

Les classes abrégées

Raccourci	Signification
<code>\d</code>	Indique un chiffre. Ça revient exactement à taper <code>[0-9]</code>
<code>\D</code>	Indique ce qui n'est PAS un chiffre. Ça revient à taper <code>[^0-9]</code>
<code>\w</code>	Indique un caractère alphanumérique ou un tiret de soulignement. Cela correspond à <code>[a-zA-Z0-9_]</code>
<code>\W</code>	Indique ce qui n'est PAS un mot. Si vous avez suivi, ça revient à taper <code>[^a-zA-Z0-9_]</code>
<code>\t</code>	Indique une tabulation
<code>\n</code>	Indique une nouvelle ligne
<code>\r</code>	Indique un retour chariot
<code>\s</code>	Indique un espace blanc
<code>\S</code>	Indique ce qui n'est PAS un espace blanc (<code>\t \n \r</code>)
<code>.</code>	Indique n'importe quel caractère. Il autorise donc tout !

Les expressions régulières

- `#^0[1678]([\-\.\.]?[0-9]{2}){4}$#`
- `#^((https?|ftp)://(\w{3}\.?)?(\w+|-?)+\.[a-z]{2,4})$#i`

La programmation orientée objet

- Permet de structurer et de factoriser le code
- Le principe
 - L'architecte crée un plan de maison -> une classe
 - le plan permet de créer des maisons -> des objets
- PDO est une classe
- `$bdd = new PDO(...)` -> `$bdd` est un objet
- Un objet est une instance de classe

La programmation orientée objet

Une fois la classe instanciée, l'objet peut appeler les fonctions de la classe (méthodes).

- `$bdd->query();`
- `$bdd->prepare();`
- `$req->execute();`

La programmation orientée objet

Créer une classe :

```
1 <?php
2 class Membre
3 {
4     private $pseudo;
5     private $email;
6     private $signature;
7     private $actif;
8
9     public function getPseudo()
10    {
11        return $this->pseudo;
12    }
13
14    public function setPseudo($nouveauPseudo)
15    {
16        $this->pseudo = $nouveauPseudo;
17    }
18 }
```

La programmation orientée objet

Inclure une classe :

```
1 <?php
2
3 include_once('Membre.class.php');
4
5 $membre = new Membre();
6 $membre->setPseudo('M@teo21');
7 echo $membre->getPseudo() . ', je vais te bannir !';
8 $membre->bannir();
9
10 ?>
```

La programmation orientée objet

- Constructeur, destructeur et autres fonctions spéciales (« méthodes magiques »)
 - **new class()** lance public function **__construct(\$arg)**
 - **unset(\$objet)** lance public function **__destruct()**
 - public function **__get()** **__set()**
 - ...

La programmation orientée objet

L'héritage

- Le concept le plus important
- Permet de factoriser le code de manière logique
- Il y a héritage quand on peut dire : « A est un B »
- exemple : « une voiture est un véhicule »

```
1 <?php
2 include_once('Membre.class.php');
3
4 class Admin extends Membre
5 {
6
7 }
8 ?>
```

La programmation orientée objet

Les droits d'accès :

- **public** : tout le monde peut accéder à l'élément
- **private** : personne (à part la classe elle-même) n'a le droit d'accéder à l'élément ;
- **protected** : identique à private, sauf qu'un élément ayant ce droit d'accès dans une classe mère sera accessible aussi dans les classes filles.

static : Permet d'accéder à des propriétés ou des méthodes sans instancier la classe.

```
<?php
class Foo
{
    public static function aStaticMethod() {
        // ...
    }
}
```

Architecture MVC

Une bonne pratique de programmation.

- **Modèle** : gère les données de votre site, récupère les informations brutes de la base de données par exemple.
- **Vue** : se concentre sur l'affichage des résultats, aucun calcul complexe, se contente d'afficher des variables dans du code HTML
- **Contrôleur** : gère la logique et prend des décisions, en quelque sorte l'intermédiaire entre la vue et le modèle.

