# Cours - PHP

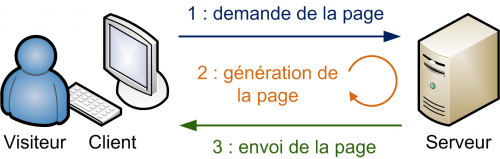
Moussa CAMARA

PHP (officiellement, ce sigle est un acronyme récursif pour PHP Hypertext Preprocessor) est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement **d'applications web**. Il peut être intégré facilement au HTML.

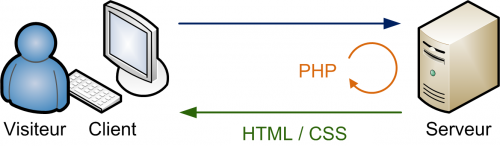
Front HTML, CSS, Angular

Back derrière manipulation du serveur

Full START fait les deux



URL (uniform ressource locator)



<html>

<?php> code php entre les valises html ?>

</html>

Toute variable il faut la déclarer avec un $ devant

<?php>

$maVariable = "hola"; ?>

Les types de variables

$maConnexion = mysql\_connect("monserveur","monNom","monPass");

Interpolation

Porte de variable

Variable globale

Variable locale

Fonction statique

Constantes tout le temps en majuscules

<?php>

class MaClasse {

const CONSTANTE = "Text ici";

function afficher()

{

echo self::CONSTANTE;

}

}

$instance = new MaClasse;

$instance->afichier;

?>

Ne jamais commencer par un chiffre

On utilise une constante Pour donner une valeur que ne change pas

## Les opérations de comparassions

==

Opérations logiques

&& et

|| ou

Conditions = Vérifie certains données

If else

Switch

<?php

$nbClick = 2;

switch ($nbClick) {

case 1:

echo 'passera pas dedans'

break;

case 2:

echo 'passera pas dedans'

break;

case 3:

echo 'passera pas dedans'

break;

default:

# code...

break;

}

?>

Ternaires

Declarer une variable

Verifier si

C’est la police (17)

C’est les pompiers (18)

C’est une mobile (06 ou 07)

C’est une fixe (01, 02, 08, 09)

Etranger

# PSR

« Propose a Standards Recommendation »

A ce jour, les recommandations PSR validées sont aux nombres de 4 :

PSR-0 : qui traite du chargement des classes PHP et de l’autoloading ;

PSR-1 : qui fixe les conventions minimales de codage ;

PSR-2 : qui défini le style et l’organisation du code ;

PSR-3 : qui s’occupe de l’interface des loggers.

BOUCLES

Les tablaux

Tp-php-veronica

Git remote add add

Concaténation

Interpolation

Commentaire naturel

$ toto = 5 ;

Commentaire

Boucles

If

10< admin

10> non admin

Liste des etudiants

|  |  |  |  |
| --- | --- | --- | --- |
| Id | prénom | nom | Qualification |
|  |  |  |  |
|  |  |  |  |

# Machine virtuel

PHP 7

# GIT créer branches :

Git remote –v

Git init

Git remote add veronica (url page Github)

Git remote –v

Git status

Git add –A

Git commit –m "tets"

*git checkout -b tp-php-veronica (pour la branche)*

git push tp-php-Veronica (peut être origin) tp-php-veronica

**(ou git push origin tp-php-veronica)**

# FORMULAIRES

## Méthode GET

PHP admid une base de données

Raf : reste à faire

Formulaire

Nom

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Prénom | Nom | Tel | email |
|  |  |  |  |  |
|  |  |  |  |  |

Prenom

Ajouter

Tel

Mail

Liste des etudiants

envoyer

# Connection à la base de données mysql

Nom fichier : db.php

<?php

$servername = "localhost";

$username = "root";

$password = "";

$db = "poec-symfony";

// Create connexion

define('DBSERVER', 'localhost');

define('DBNAME','poec symfony');

define('DBUSER','root');

define('DBPASS','');

//$connection = new mysqli($servername, $username, $password, $db);

$connection = mysqli\_connect(DBSERVER, DBUSER, DBPASS, DBNAME);

//

if (!$connection) {

die('<strong> Impossible de se connecter, veuillez revoir les acces</strong><br>' .mysqli\_connect\_error());

}else{

//echo '<strong>Super! j\'ai réussi à me connecter à ma basse de données</strong>';

}

//include 'exercise5.php';

?>

# Création de formulaire

Nom fichier : index.php

<?php

require\_once "db.php"

?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<link rel="stylesheet" type="text/css" href="">

<link rel="stylesheet" href="formulaire.css" />

<link rel="stylesheet" type="text/css" href="bootstrap/css/bootstrap.css">

<title>Formulaire</title>

</head>

<body>

<main>

<h2>Formulaire avec la methode POST</h2>

<form action="exercise5.php" method="POST">

<div class="form-row">

<div class="form-group">

<label for="nom">Nom :</label>

<input type="test" name="nom" id="nom" placeholder="nom"><br>

</div>

<div class="form-group">

<label for="prenom">Prenom :</label>

<input type="test" name="prenom" id="prenom" placeholder="prenom"><br>

</div>

<div class="form-group">

<label for="email">Email :</label>

<input type="test" name="email" id="email" placeholder="email"><br>

</div>

<div class="form-group">

<label for="adresse">Adresse :</label>

<input type="test" name="adresse" id="adresse" placeholder="adresse"><br>

</div>

<div class="form-group">

<label for="fonction">Fonction :</label>

<input type="test" name="fonction" id="fonction" placeholder="fonction"><br>

</div>

<div>

<input type="submit" class="btn btn-primary">

</div>

</div>

</form>

</main>

</body>

</html>

# Insertion à la base de données

Nom fichier : exercise5.php

<?php

require\_once "db.php"

?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<link rel="stylesheet" type="text/css" href="bootstrap/css/bootstrap.css">

<title>Formulaires POST</title>

</head>

<body>

<?php

//mettre dans un autre fichier

if (count($\_POST) >0){

$nom = $\_POST['nom'];

$prenom = $\_POST['prenom'];

$email = $\_POST['email'];

$adresse = $\_POST['adresse'];

$fonction = $\_POST['fonction'];

$isExist ="select nom from user where nom='".$nom."'";

//var\_dump($nom);die;

$exist = mysqli\_query($connection,$isExist);

/\* \*/

/\* Détermine le nombre de lignes du jeu de resultats\*/

$var = mysqli\_num\_rows($exist);

if ($var > 0) {

echo "<div class='alert alert-danger' role='alert'>";

}else {

/\*

\* Instertion dans la base de données

\*/

}

echo $\_POST['nom'];

echo $\_POST['prenom'];

echo $\_POST['email'];

echo $\_POST['adresse'];

echo $\_POST['fonction'];

/\*

Pour faire la requete

\*/

$sql= "INSERT INTO user(nom, prenom, email, adresse, fonction) VALUES ('".$nom."','".$prenom."','".$email."','".$adresse."','".$fonction."')";

/\* Quand la requête est bonne j'efface var dump\*/

// connexion avec ma base de données et avec ma requête

$insert = mysqli\_query ($connection, $sql);

if ($insert) {

echo "<div class='alert alert-success'role='alert'>ok!</div>";

} else {

echo "<div class='alert alert-danger' role='alert'>Oups! Insertion échoué</div>";

}

//Se deconnecte

mysqli\_close($connection);

}

/\*

Une autre manière de faire le code avec la commande "isset".

if(isset($\_POST['nom']) && isset($\_POST['prenom']) && isset($\_POST['adresse']) && isset($\_POST['fonction']) && isset($\_POST['email'])){

}

\*/

?>

</body>

</html>

# Faire une liste avec ma base de données

Nom fichier : liste.php

## Pour me connecter à ma base de données on utilise require\_once et le nom de mon fichier de connexion est db.php

<?php

require\_once "db.php"

?>

## Constitution de table

<table class="table table-striped" style="width: 50%">

<thead>

<caption style="font-size: 1.2em">Liste des étudiantes</caption>

<tr>

<th scope="col">Id</th>

<th scope="col">Prenom</th>

<th scope="col">Nom</th>

<th scope="col">Email</th>

<th scope="col">Adresse</th>

<th scope="col">Foction</th>

</tr>

</thead>

## Remplir ma table avec php

<?php

$sql = "SELECT \* FROM USER"; (ma requête)

// Accès à ma base de données et ma requête

$resultat = mysqli\_query($connection, $sql);

/\*

\* mysqli\_fetch\_row()result->fetch\_row() Récupere une ligne de resultat sous forme de tableau associatif

\*/

//j'utilise ma boucle white et une table assotiative

while ($row = mysqli\_fetch\_assoc($resultat)) {

?>

<tr>

<td>

<?php

//echo $row[0];

echo $row['Id'];

?>

</td>

<td>

<?php

//echo $row[1];

echo $row['nom'];

?>

</td>

<td>

<?php

//echo $row[2];

echo $row['prenom'];

?>

</td>

<td>

<?php

//echo $row[3];

echo $row['email'];

?>

</td>

<td>

<?php

//echo $row[4];

echo $row['adresse'];

?>

</td>

<td>

<?php

//echo $row[5];

echo $row['fonction'];

?>

</td>

</tr>

<?php

}

?>

</tbody>

</table>

Bienvenue

Sign in

Sing up

login

email

password

bouton se connecter

index.php seulement de html

bienvenue avec les boutons login et sign up

< hrefer = « login.php>

< hrefer = « inscription.php>

*MON FICHIER INDEX ;PHP*

Menu

connection

inscription

*Conextion*

login

password

*iNSCRIPTION*

*Tabla list*

*Ajouter*

Nom

Prenom

Email

Chekout

*déconnection*

# Création de mon serveur de projet

1. je crée mon commande de serveur d’un projet X dans mon cas je l’appelle veronica.dev

#mon creation serveur

<VirtualHost \*:80>

ServerName veronica.dev

ServerAlias veronica.dev

DocumentRoot "C:/wamp64/www/listeEtudiants/"

ErrorLog "logs/veronica.dev-error.log"

CustomLog "logs/veronica.dev-access.log"common

</VirtualHost>

Je copie <VirtualHost> dans le fichier

C:\wamp64\bin\apache\apache2.4.27\conf\extra\httpd-vhost.conf

2. Dans le fichier

C:\wamp64\bin\apache\apache2.4.27\conf\extra\httpd.conf

J’inclue la commande :

Include cof/extra/httpd-vhosts.cof

3.- Je copie le fichier host de

C:\Windows\System32\drivers\etc\host

Dans une autre dossier par exemple pour le modifier

C:\Users\etudiant13\Desktop\host

Après j’annexe à la fin

127.0.0.1 veronica.dev

Après je le met une autre fois à son lieu d’origine

C :\Windows\System32\drivers\etc\host

J’accepte la modification

Après je démarre le Wampserver

et j’appelle mon serveur dans mon cas s'appelle v*eronica.dev*

# PHP orienté à objet POO

## Objectifs principaux

* Concevoir
* Améliorer
* Simplifier

## Concepts fondamentaux

1. Objet, classe et instance
2. Les membres de classe et d’instance
3. Envoi de message et méthode
4. Héritage, encapsulation et polymorphisme
5. Constructeur et destructeur
6. Classe abstraite / concret

## Terminologie

La classe est la structure d’un objet

Une classe est un ensemble d’attributs et de méthodes

1. Attributs de données spécifiques à cet objet
2. Méthodes ou propriété: les verbes les actions

Un Objet est créé à partir d’une classe (un retour)

Exemples

1. Personne :

Attributs : nom, prénom, taille

Méthodes : actions courir, manger, parler, etc.

1. Email :

Titre, auteur, date, image, message

**Variable constantes** doit être déclarée et initialisée avec sa valeur en même temps et n’a pas le $

Const NOMBRE\_DE\_BRAS = 2

## Instancier une classe

## Affichage des attributs

## La portée des Variables POO

Public (globale)

Private (locale) sont disponibles que pour la classe en elle même

Protecte on va l’utiliser quand on va faire de l’héritage

class Article {

private$\_private = "Je suis private";

protected $\_protected =

}

Modifier une variable

Getter récupérer gat

et setter set modifier

Il est préférable de passer par des méthodes

Exercise

Classe personne

Nom

Prenom

age

## Les annotations

Est un élément permettant d’ajouter des métadonnées à un code source.

## Les namespaces

Permet de regler les problèmes de nom de classe et permet de dire en que on travaille

On crée deux fichiers

class Personne1{

//\*@var string

private $nom;

//\*@var string

private $nom;

//\*@var int

private $age;

}

|  |  |
| --- | --- |
| Namespace personne1 | Namespace personne2 |
| P1  Class personne{  $nom  $prenom ;  $age ;  } | P2  Class personne{  $nom  $prenom ;  $coleur;  $taille;  } |

$personne = new Personne ( ) ;

## Méthode magique constructeur

Un constructeur

## Héritage

L’héritage est un mécanisme qui facilite la réutilisation du code et la gestion de son évolution. Grâce à l’héritage, les objets d’une classe ont accès aux données

extends =>pour indiquer une classe hérite d’une autre. En l’absence de de mot réservé associe à une classe , le compilateur considère la classe Objet comme classe parent.