

PHP

Passez à la vitesse supérieure avec PHP

Plan du cours

- Introduction à PHP
 - C'est quoi ?
 - Utilisation dans le développement web
- Architecture Client / Server
- Communication entre Client et Serveur
- Fonctionnement de site web statique et dynamique
- Environnement de développement
 - Téléchargement et installation d'un serveur (XAMPP, WAMP, MAMP,...)
- Premier pas avec PHP
 - Structure d'une page PHP
 - Affichage de texte
 - Commentaires

Plan du cours

- Variables et types de données
 - Déclaration de variable
 - Types de données de base
 - Opération sur les données
- Les structures conditionnelles
 - if, else, elseif, switch
- Les structure itératives
 - for, while, do while
- Inclusions de fichiers
 - include
 - require
- Les fonctions

Plan du cours

- Les tableaux
 - Tableaux indicés
 - Tableau associatifs
 - Parcours et affichage des éléments d'un tableau
- Formulaires HTML et Traitement PHP
 - Création des formulaire HTML
 - Soumission de formulaire et traitement des données
 - Validation des données côté serveur
- Interaction avec une Base de données CRUD
 - Requête d'insertion
 - Requête de lecture
 - Requête de mise à jour
 - Requête de de suppression

Plan du cours

- Sessions et Cookies
 - Utilisation des sessions pour maintenir l'état des utilisateurs
 - Gestion des cookies
- Gestion de fichiers
 - Lecture
 - Ecriture
- Hébergement d'un site

Introduction à PHP

- date de création 1994 (Rasmus Lerdorf)
- Signifie Personal Home Page, puis Hypertext Preprocessor
- Langage interprété et dynamique
- Principalement utilisé pour générer des page web dynamique, manipuler des données et interagir avec des base de données
- Langage serveur

Le langage PHP est un langage de programmation informatique **interprété** et **open source**. Il permet d'écrire des scripts qui s'exécutent sur un **serveur** pour générer des pages web de manière **dynamique**.

Architecture client serveur

Le web c'est une communication qui fonctionne sur la base d'une architecture **client / serveur**.

- **Client**

Le client c'est nous, c'est le **navigateur Web** qui grâce à des règles établies va nous permettre de communiquer avec une source distante afin de consulter des pages **Web** depuis un ordinateur, un smartphone ou une tablette...

- **Server**

Et le serveur, c'est un ordinateur puissant qui stocke et **héberge des sites Web**. C'est sur cet ordinateur que se trouvent les **pages Web**, c'est-à-dire tous les fichiers du **site internet** auquel on veut accéder.

Un serveur doit rester allumé 24/24 pour que les sites qu'il héberge soient toujours accessibles.

Communication entre le client et le serveur

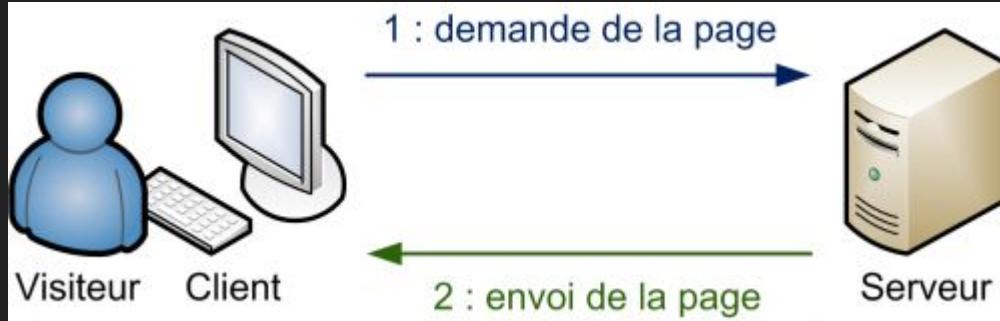
Le client accède à une **page Web** en utilisant son adresse web – appelée **URL** – dans son **navigateur**. Par exemple, l'URL

http://monsite.fr/mondossier/mapage.html

1. “**http**”, c’est le nom du **protocole de communication** entre le client et le serveur,
2. “**monsite.com**” est le **nom de domaine du site Web** auquel on veut accéder,
3. “**mondossier/mapage.html**” est l’endroit où se trouve la **page dans le serveur**.

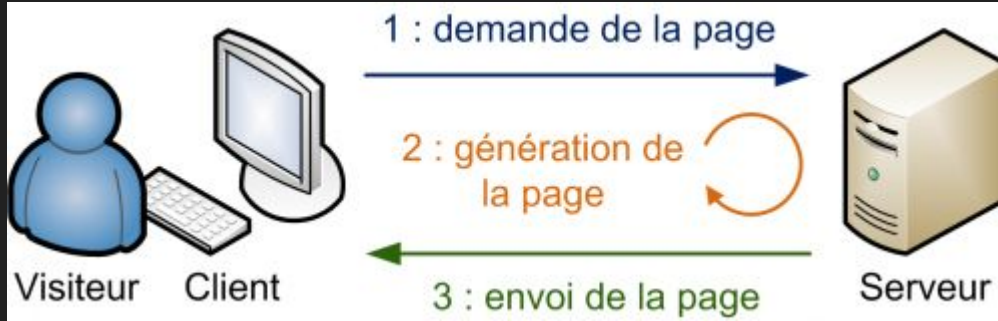
HTTP: ***Hypertext Transfer Protocol*** est le protocole de communication entre un client et un serveur pour le web. Il existe une version sécurisée de ce protocole appelé **HTTPS** ***Hypertext Transfer Protocol Secure***.

Fonctionnement d'un site web statique



Le client envoie une requête au serveur qui se contente de renvoyer la page demandée si elle existe sinon retourne une réponse 404 synonyme de page non existante.

Fonctionnement d'un site web dynamique



Lorsque le **client** commande une page au **serveur**, ce dernier prépare cette page. Si la page nécessite des données spécifiques (à partir d'une base de données), le serveur récupère ces données et les traite avant de générer la page et la retourner au client.

Environnement de développement



Premier pas avec PHP

```
<?php
    // code PHP
?>
<!DOCTYPE html>
<html>
<head>
    <title>Titre de la page</title>
</head>
<body>
    <?php
        // code PHP
    ?>
    <p>
        <?php /** code PHP */?>
    </p>
</body>
</html>
```

Premier pas avec PHP

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Titre de la page</title>
```

```
  </head>
```

```
  <body>
```

```
    <?php echo "<h1>Bonjour le monde</h1>"; ?>
```

```
  </body>
```

```
</html>
```

Premier pas avec PHP

```
<!DOCTYPE html>
<html>
<head>
  <title>Titre de la page</title>
</head>
<body>
  <?php
    // commentaire sur une ligne
    # commentaire sur une ligne autrement
    /* commentaire
    sur plusieurs
    lignes */
  ?>
</body>
</html>
```

Variables et type de données

En programmation, les variables sont des emplacements de mémoire qui sont utilisés pour stocker et manipuler des données. Chaque variable a un nom unique qui la distingue des autres variables, et elle peut contenir différentes valeurs en fonction du type de données qu'elle contient. Les types de données déterminent la nature de la valeur qu'une variable peut contenir et les opérations que l'on peut effectuer sur cette valeur.

Variables et type de données


En PHP pour déclarer une variable on met le symbole “\$” devant le nom de la variable.


Le nom est sensible à la **casse** c'est-à-dire fait la différence entre les majuscules et les miniscule.

Les noms de variables doivent respecter certaines règles de nommage.

Un nom de variable valide doit commencer par une lettre ou un souligné (_), suivi de lettres, chiffres ou soulignés.

Variables et type de données

\$prenom;  déclaration d'une variable nommée prenom

\$prenom = "Dupond";  affectation de la valeur Dupond à la variable prenom

pour affecter une valeur à une variable on fait suivre le nom de la variable le signe "**=**" et puis après préciser la valeur qu'on souhaite donner à la variable.

Variables et type de données

```
<?php
// Entiers (integers)
$age = 25;
$count = 10;

// Décimaux (floats)
$prix = 19.99;
$pi = 3.14159;

// Chaînes de caractères (strings)
$nom = "Alice";
$message = 'Bonjour, comment ça va ?';

// Booléens
$estVrai = true;
$estFaux = false;
```

Variables et type de données

```
<?php
// Tableaux (arrays)
$nombre = array(1, 2, 3, 4, 5);
$personne = array("nom" => "Jean", "âge" => 30, "ville" => "Paris");

$nombre = [1, 2, 3, 4, 5];
$personne = ["nom" => "Jean", "âge" => 30, "ville" => "Paris"];
```

Opérations sur les chaînes de caractères

```
// Concaténation
$prenom = "John";
$nom = "Doe";
$nomComplet = $prenom . " " . $nom; // Résultat : "John Doe"

// Longueur d'une chaîne
$texte = "Bonjour";
$longueur = strlen($texte); // Résultat : 7

// Extraction de sous-chaînes
$phrase = "La programmation est amusante";
$sousPhrase = substr($phrase, 3, 13); // Résultat : "programmation"

// Recherche de texte
$texte = "Bonjour tout le monde";
$position = strpos($texte, "tout"); // Résultat : 8 (position du mot "tout")
```

Opérations sur les chaînes de caractères

```
// Remplacement de texte
$texte = "Les chats sont adorables";
$nouveauTexte = str_replace("chats", "chiens", $texte); // Résultat : "Les chiens sont adorables"
// Conversion de casse
$texte = "Hello World";
$minuscules = strtolower($texte); // Résultat : "hello world"
$majuscules = strtoupper($texte); // Résultat : "HELLO WORLD"
// Conversion en majuscule/minuscule initiale
$texte = "hello world";
$premiereMaj = ucfirst($texte); // Résultat : "Hello world"
$premiereMin = lcfirst($texte); // Résultat : "hello world"
// Suppression des espaces
$texte = "    Bonjour    ";
$texteSansEspaces = trim($texte); // Résultat : "Bonjour"
```

Opérations sur les chaînes de caractères

```
// Conversion en tableau
$liste = "pomme,orange,banane";
$tableau = explode(",", $liste); // Résultat : ["pomme", "orange", "banane"]
// Jointure de tableau en chaîne
$tableau = ["pomme", "orange", "banane"];
$liste = implode(", ", $tableau); // Résultat : "pomme, orange, banane"
// Vérification de présence de sous-chaîne
$texte = "Bonjour à tous";
$contientBonjour = strpos($texte, "Bonjour") !== false; // Résultat : true
```

Existence d'une variable

La fonction `isset()` en PHP est utilisée pour vérifier si une variable est définie et a une valeur non nulle. Elle renvoie `true` si la variable est définie et `false` si elle ne l'est pas ou si elle a la valeur `null`.

Obtenir le type d'une variable

- `gettype()`
- `is_int()`, `is_double()`, `is_string()`, `is_array()`, `is_object()`, `is_bool()`

Exemple

```
$a = 1;           echo gettype($a);    // Affiche integer
$b = 1.5;         echo gettype($b);    // Affiche double
$c = NULL;        echo gettype($c);    // Affiche NULL
$d = new Persone(); echo gettype($d);    // Affiche object
$e = "Bonjour";   echo gettype($e);    // Affiche string
```

```
is_int($a);      // Renvoie TRUE
is_int($b);      // Renvoie FALSE
```


Conversion de type

Transtypage = conversion de type

Via la fonction **settype()**

```
$nbr = 10;  
echo gettype($nbr);           // Affiche : integer  
settype($nbr, "double");  
echo gettype($nbr);           // Affiche : double
```

Via un **cast** : (int), (integer), (bool), (boolean), (float), (double), (real),(string)

```
$str = "10 maisons";  
$nbr = (int) $str;  
echo "$nbr = $nbr ; son type est : " . gettype($nbr);  
// Affiche : $nbr = 10 ; son type est : integer
```

Conversion de type

Certaines conversions sont automatiques :

```
$foo = "0";           // $foo est une chaîne de caractères  
$foo += 2;            // $foo est maintenant un entier (2)  
$foo = $foo + 1.3;    // $foo est maintenant un nombre à virgule flottante (3.3)  
$foo = 5 + "10";      // $foo est un entier (15)
```

Les constantes

une constante est une variable qui une fois déclaré ne change pas de valeur c'est une variable dont la valeur reste constante durant l'exécution du script.

Définition:

```
define("G", 9.8);  
define("JAUNE", "#FFFF00");  
define("PI", 3.14);
```

Utilisation d'une constante

```
echo 'l'intensité de la pesanteur g égale' . G;
```

Les opérateurs arithmétiques

Exemple	Nom	Résultat
$-\$a$	Négation	Opposé de $\$a$.
$\$a + \b	Addition	Somme de $\$a$ et $\$b$.
$\$a - \b	Soustraction	Différence de $\$a$ et $\$b$.
$\$a * \b	Multiplication	Produit de $\$a$ et $\$b$.
$\$a / \b	Division	Quotient de $\$a$ et $\$b$.
$\$a \% \b	Modulo	Reste de $\$a$ divisé par $\$b$.
$\$a ** \b	Exponentielle	Résultat de l'élévation de $\$a$ à la puissance $\$b$. Introduit en PHP 5.6.

Les opérateurs d'affectation et opérateur combinés

Opérateur	Définition
.=	Concatène puis affecte le résultat
+=	Additionne puis affecte le résultat
-=	Soustrait puis affecte le résultat
*=	Multiplie puis affecte le résultat
/=	Divise puis affecte le résultat
%=	Calcule le modulo puis affecte le résultat
**=	Élève à la puissance puis affecte le résultat

Les opérateurs de comparaisons

Exemple	Nom	Résultat
<code>\$a == \$b</code>	Egal	TRUE si <i>\$a</i> est égal à <i>\$b</i> après le transtypage.
<code>\$a === \$b</code>	Identique	TRUE si <i>\$a</i> est égal à <i>\$b</i> et qu'ils sont de même type.
<code>\$a != \$b</code>	Différent	TRUE si <i>\$a</i> est différent de <i>\$b</i> après le transtypage.
<code>\$a <> \$b</code>	Différent	TRUE si <i>\$a</i> est différent de <i>\$b</i> après le transtypage.
<code>\$a !== \$b</code>	Différent	TRUE si <i>\$a</i> est différent de <i>\$b</i> ou bien s'ils ne sont pas du même type.
<code>\$a < \$b</code>	Plus petit que	TRUE si <i>\$a</i> est strictement plus petit que <i>\$b</i> .
<code>\$a > \$b</code>	Plus grand	TRUE si <i>\$a</i> est strictement plus grand que <i>\$b</i> .
<code>\$a <= \$b</code>	Inférieur ou égal	TRUE si <i>\$a</i> est plus petit ou égal à <i>\$b</i> .
<code>\$a >= \$b</code>	Supérieur ou égal	TRUE si <i>\$a</i> est plus grand ou égal à <i>\$b</i> .

Les opérateurs logiques

Mot-clé	Signification	Symbole équivalent
AND	Et	&&
OR	Ou	

Les structures conditionnelles

Comme tout langage, PHP dispose d'instructions conditionnelles qui permettent d'orienter le déroulement d'un script en fonction de la valeur des données.

Un script PHP est exécuté dans l'ordre de la première ligne du fichier jusqu'à la dernière ligne, On a parfois besoin d'afficher des choses différentes en fonction de certaines données.

C'est là qu'interviennent les conditions. Elles permettent de donner des ordres différents à PHP selon le cas.

Les structures conditionnelles

- **if** pour exécuter un bloc de code, si une condition spécifiée est vraie.
- **else if** pour spécifier une nouvelle condition à tester, si la première condition est fausse et exécuter le bloc de code adéquat.
- **else** pour exécuter un bloc de code, si la / les condition (s) sont fausses
- **switch** pour spécifier de nombreux blocs de code alternatifs à exécuter

Les structures conditionnelles

```
if (condition) {  
    // instructions à exécuter si la condition est vrai  
}
```

exemple

```
if($a > $b){  
    echo "$a est supérieur à $b"  
}
```

Les structures conditionnelles

```
if (condition) {
```

```
    // instructions à exécuter si la condition est vrai
```

```
}else{
```

```
    // instructions à exécuter si la condition est vrai
```

```
}
```

Les structures conditionnelles

exemple

```
if($a > $b){  
    echo "$a est supérieur à $b";  
}  
else{  
    echo "$a n'est pas supérieur à $b";  
}
```

Les structures conditionnelles

```
if (condition) {  
    // instructions à exécuter si la condition est vrai  
}  
else if(condition){  
    // instructions à exécuter si la condition est vrai  
}  
else{  
    // autres instructions  
}
```

Les structures conditionnelles

exemple

```
if($a > $b){  
    echo "$a est supérieur à $b";  
}  
else if($a < $b){  
    echo "$a est inférieur à $b";  
}else{  
    echo "$a est égale à $b";  
}
```

Les structures conditionnelles

```
switch(expression) {  
  
    case x:  
  
        // instructions à exécuter pour ce cas  
  
        break;  
  
    case y:  
  
        // instructions à exécuter pour ce cas  
  
        break;  
  
    default:  
  
        // instructions à exécuter par défaut  
  
}
```

Les structures conditionnelles

```
$jour = 5;
switch ($jour) {
    case 0:
        echo "Dimanche";
        break;
    case 1:
        echo "Lundi";
        break;
    case 2:
        echo "Mardi";
        break;
    case 3:
        echo "Mercredi";
        break;
    case 4:
        echo "Jeudi";
        break;
    case 5:
        echo "Vendredi";
        break;
    case 6:
        echo "Samedi";
        break;
    default:
        echo "Jour inconnu";
}
```


Les structures conditionnelles

`opérateur = condition ? val1 : val2`

Si la condition vaut true, `opérateur` vaudra `val1`. Sinon elle vaudra `val2`.

Exemple :

```
$age = 12;
```

```
$statut = ($age >= 18) ? "adulte" : "mineur";
```