Les bases de PHP

PLAN

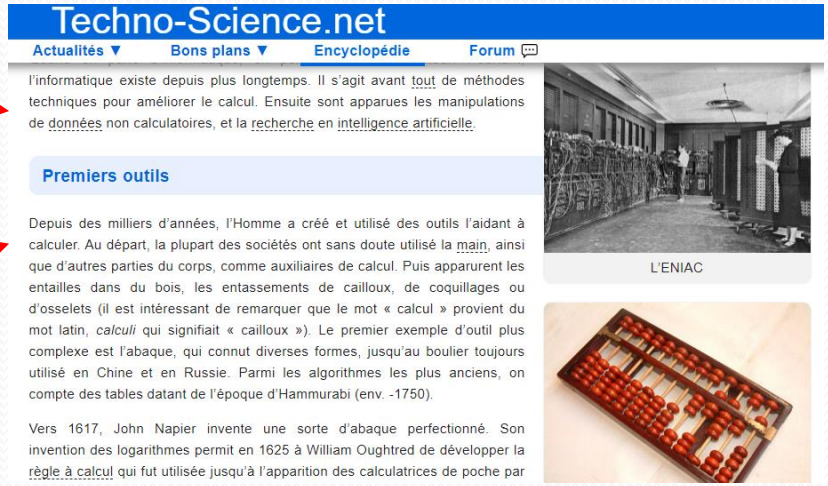
- Présentation de PHP
- Les sites statiques et les sites dynamiques
- Fonctionnement d'un langage coté serveur
- Les bases du langages PHP
  1. Syntaxe
  2. Affichage du texte
  3. Les variables
  4. Les types des variables
  5. Le casting
  6. Connaitre le type d'une variable
  7. La concaténation
  8. Les opérateurs
  9. Les structures conditionnelles
  10. Les boucles
- Fonction de chaines de caractères utiles

Présentation de PHP

- PHP (Hypertext PreProcessor) est un langage utilisé principalement pour produire des pages **Web dynamiques** via un **serveur web**, on dit que c'est un langage **côté serveur,**
- Il peut également fonctionner comme n'importe quel langage **interprété** de façon locale.
- PHP est un langage aussi **orienté objet**.
- Les avantages :
  - Open source
  - Gratuit
  - Multi plate-forme

Les sites statiques

- Ce sont des sites réalisés uniquement à l'aide des langages **HTML**, **CSS** et des **langages côté client** comme le **JavaScript**.
- Si on veut ajouter des nouveautés; des données par exemple dans ces sites, **Il faut que le propriétaire du site (le webmaster) modifie le code source pour faire l'ajout de ces parties.**



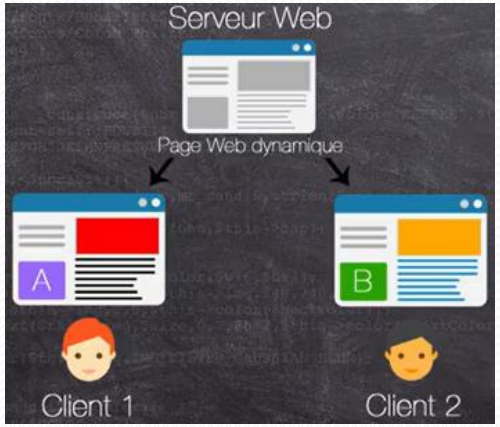
Tous les clients voient la même chose

Les sites dynamiques

- Un site est dit « **dynamique** » parce que son **contenu** peut **changer** sans l'intervention du webmaster, c'est-à-dire sans modification de code. Et on peut avoir un contenu différent selon le client qui est connecté au site, et le contenu change automatiquement d'un temp à autre.

- Les sites **dynamiques utilisent des bases de données relationnelle crées avec des SGBDR comme mysql, sqlserver, …**

ou des bases de données stockées dans des fichiers **XML**, **JSON**…



Langage côté client et langage côté serveur

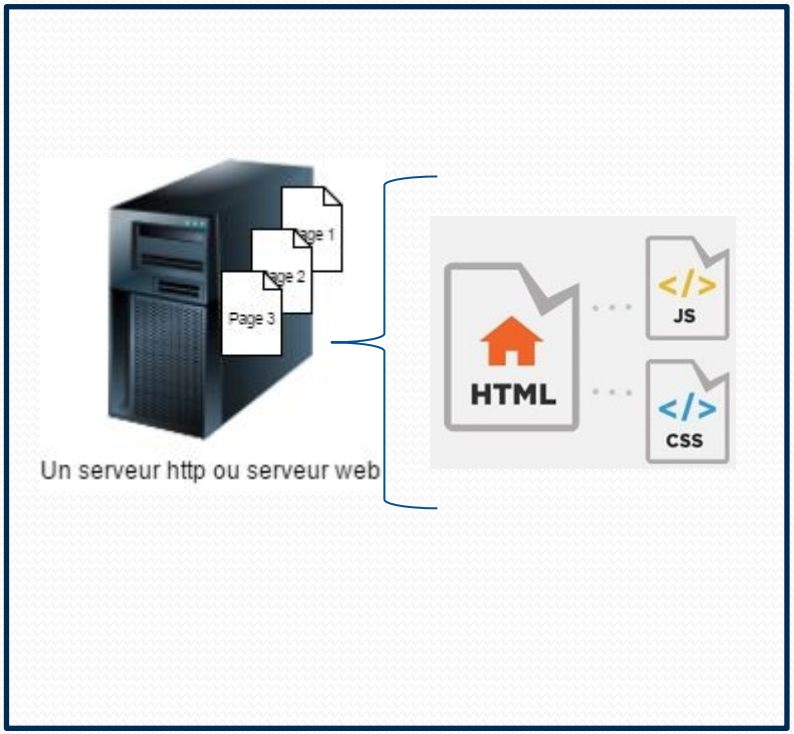
Internet est un réseau composé d'ordinateurs en communication. Ceux-ci peuvent être classés en deux catégories:

1. **Les clients : ce sont les ordinateurs des internautes**. Votre ordinateur fait donc partie de la catégorie des clients. Chaque **client** représente un **visiteur** d'un site web. (il doit y installer navigateur web : chrome, opera, firefox , ...)

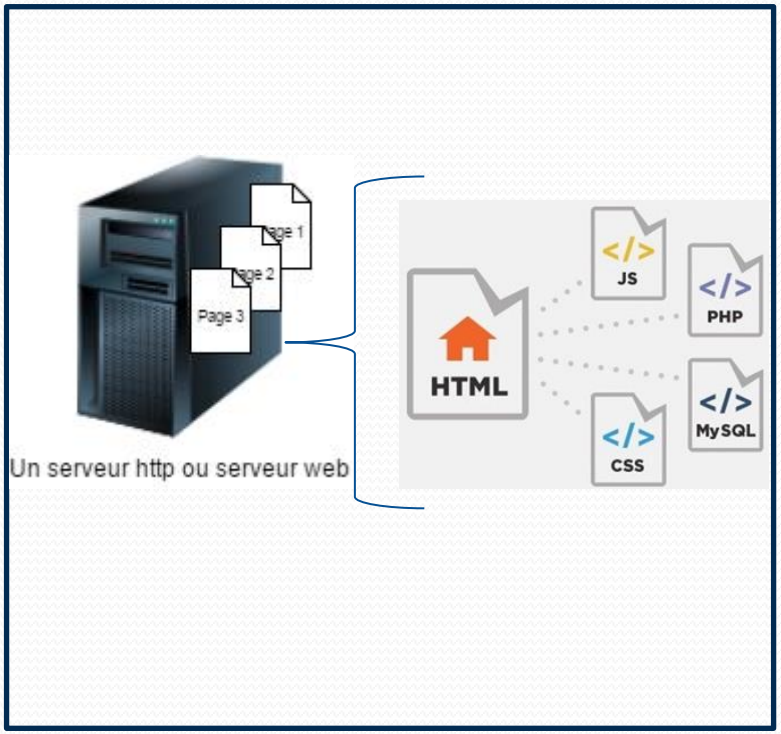
2. **Les serveurs :** ce sont des **ordinateurs puissants** qui **stockent** et **délivrent** des sites web aux internautes, c'est-à-dire aux clients. (on doit y installer un serveur web: apache, IIS, tomcat)

Le serveur web

Site statique



Site dynamique



Echange d'informations entre le client et le serveur

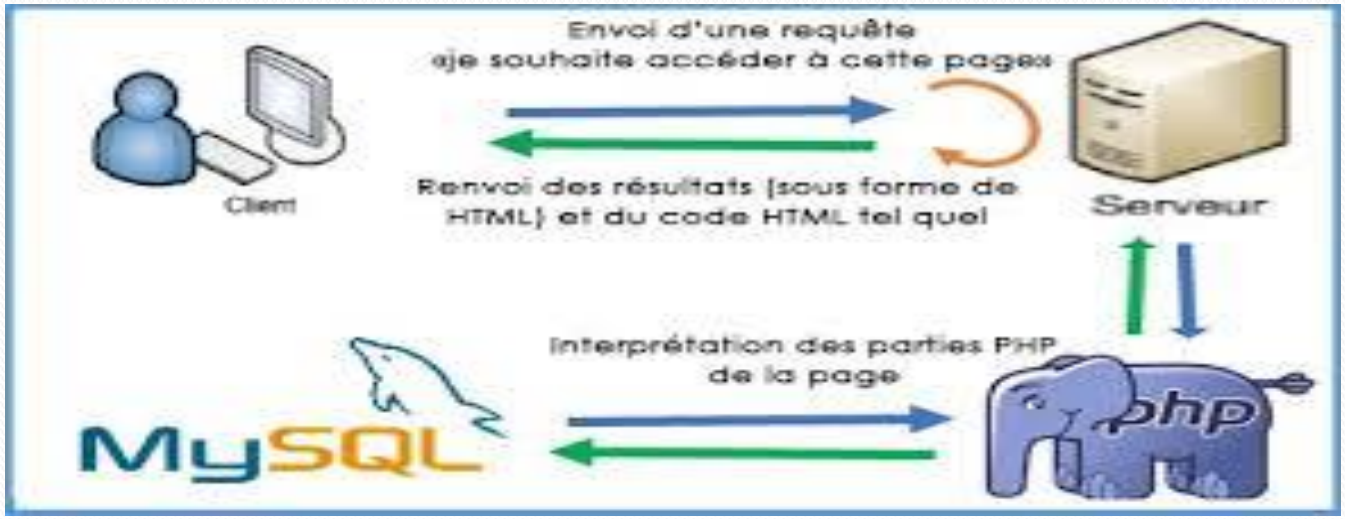
1. **Site statique**:
   - le client demande au serveur à voir une page web, par exemple : **https://MonSite.com/pagex.html**
   - le serveur cherche la page demandée, et lui répond en lui **envoyant** la page réclamée. Le serveur stocke des pages web et les envoie aux clients qui les demandent **sans les modifier.**

Echange d'informations entre le client et le serveur

2. **Site dynamique:**

- Le client demande au serveur à voir une page web dynamique **https://Monsite.com/pagex.php**

- Le serveur prépare la page **spécialement pour le client** et lui envoie la page qu'il vient de **générer**.
- Le client ne connait que le HTML et le CSS et langages côté client. Seul le serveur est capable de lire du code **PHP**



Quel outils pour le client et le serveur?

Internet est un réseau composé d'ordinateurs. Ceux-ci peuvent être classés en deux catégories:

1. **Les clients : ce sont les ordinateurs des internautes**. Votre ordinateur fait donc partie de la catégorie des clients. Chaque **client** représente un **visiteur** d'un site web. (il doit y installer navigateur web)

2. **Les serveurs :** ce sont des ordinateurs **puissants** qui **stockent** et **délivrent** des sites web aux internautes, c'est-à-dire aux clients. La plupart des internautes n'ont jamais vu un serveur de leur vie. Pourtant, les serveurs sont indispensables au bon fonctionnement du Web. (on doit y installer le serveur web: apache, IIS)

Syntaxe du langage PHP

- Le code PHP est délimité par les balises **<?php** et **?>**
- le code PHP peut s'écrire au milieux de la page HTML.
- Le code PHP peut s'écrire dans une page PHP pure (sans HTML)
- Le code PHP peut s'écrire même à l'intérieur de balise HTML
- Le code HTML peut s'écrire à l'intérieur du code PHP,

```
<html>
<head><title> .....</title><meta charset="utf-8" />
</head>
<body>
 <?php

 /*  du code PHP  */

 ?>

</body>
</html>
```

Syntaxe du langage PHP

- Chaque instruction se termine par ;
- Le langage est **sensibles à la casse: fait la difference entre majuscule et minuscules**
- Le commentaire d'une ligne c'est: // commentaire
- Le commentaire de plusieurs lignes c'est:

```
/*
Ligne 1
Ligne 2
.....
*/
```

Affichage d'un texte

- Pour afficher un texte en PHP, on utilise l'instruction **echo:**

```
<?php  echo  "Ceci est du texte" ;  ?>
```

- **Remarque**: si on veut afficher des guillemets "" dans une phrase, il faut la précéder par \

```
<?php  echo  "Ceci est du <strong> texte </strong>"  ;  ?>
```

- Les balises et sont interprétés par PHP.

Les variables

- Une variable est un élément de la programmation que l'on nomme par une suite de caractères. On associe une valeur à une variable pour l'utiliser en programmation.

- Une variable doit obligatoirement commencer par une lettre ou _

- Le caractère **$** est toujours placée avant le nom de la variable.

Les variables

- ***Exemple 1:***

```php
<?php

$a=1;
$x=true;
$y="bonjour";
$z=5.5;

 ?>
```

- Lorsqu'on affecte une valeur à une variable, on dit que la variable est **chargée (is set : en anglais)**
- Le faite d'affecter une valeur à une variable, c'est la déclaration de la variable.

Effacer le contenu d'une variable

- Pour effacer la valeur contenue dans une variable (ou **décharger** une variable de sa valeur), on utilise la fonction **unset**

```
unset($x);
unset($x, $y, $z);
```

- Une variable déchargée de sa valeur devient **indéfinie**

Les types des variables

- Les chaines de caractères (string): les chaines de caractères sont délimitées par des guillemets " " ou des apostrophes ' '

  'bonjour' ou "bonjour" =>c'est la même chose.

- Les entiers (**int** ou **integer**)
- Les décimaux (**float** ou **double**)
- Les booléens (**bool** ou **boolean**)
- Tableaux (**Array**)
- **Object**
- Rien (**NULL**)

Changement de type de variable

- Une variable peut changer de type par une simple affectation, php est donc un langage à typage dynamique.

```
<?php

$x="o";          //la variable contient une
chaine
$x=2;            //la variable contient un  entier
$x=5.5;          //la variable contient un réel

 ?>
```

Le transtypage (casting)

- La conversion de type en PHP fonctionne de la même manière qu'en C : le nom du type désiré est écrit entre parenthèses devant la variable à transtyper

```
<?php

$x= 10;                          // $x est un entier
$y= (float) $x;                  // $y est un float

?>
```

Le transtypage (casting)

- Les conversions autorisées sont :

1. (int) , (integer) - type entier
2. (bool) , (boolean) - booléen
3. (double) , (float) , (real) - type réel
4. (string) - type chaîne de caractère
5. (array) - type tableau
6. (object) - type objet

Le transtypage (casting)

- ***Remarque:***

Au lieu de transtyper une variable en chaîne en utilisant (string), vous pouvez aussi l'insérer entre deux guillemets doubles :

```
<?php

$foo = 10;                  // $foo est un entier
$str = "$foo";              // $str est une chaîne
$fst = (string) $foo;       // $fst est aussi une chaîne

?>
```

Connaitre le type d'une variable

- La fonction **gettype()** est une fonction intégrée à PHP utilisée pour obtenir le type d'une variable.
- ***Exemple:***

```php
<?php
$var1 = true;
$var2 = 3;
$var3 = 5.6;
$var4 = "Abc3462";


echo gettype($var1);    /*boolean*/
echo gettype($var2);    /*integer*/
echo gettype($var3);    /*Double*/
echo gettype($var4);    /*string*/

?>
```

Fonction pour vérifier le type de la variable

Fonction	Utilisation
isset	Dès que la valeur a été affectée à une variable, celle-ci devient **set** Exemple: **if(isset($x))** : on veut vérifier si la variable $x est chargée par une valeur ou non. (null est considéré non set)
empty	Pour ne pas être empty, une variable doit contenir autre chose que o et "" Exemple: **if(empty($x))** : **vrai si $x contient o ou "" ou null**
is_numeric	Retourne **true** si la variable est un nombre même s'il est entre ""
is_string	Retourne **true** si la variable est une chaine de caractères (vérifie le type)
is_bool	On vérifie si la valeur passé en paramètre est un booléen ou non.
is_int **is_integer**	On teste si la variable est un entier ou non
is_double **is_float**	On teste si la variable est un réel ou non
is_null	Une variable est de type null si - On lui a affecté la valeur null - On lui a affecté la fonction unset

L'opérateur de concaténation

- Les "" et les '' sont les deux utilisés pour délimiter des chaines de caractères.
- Le caractère de concaténation est le point (.)

```
$x="Talbi";
$y="Mohamed";
$z=$x." " .$y;          // renvoie "Talbi Mohamed"
$z=$x.' ' .$y;          // renvoie "Talbi Mohamed"
$z= " $x $y ";          //renvoie le même résultat
```

L'opérateur de concaténation

- ***Exemple 2:***

```
<?php

$x=20;

echo  " le stagiaire a $x ans ";          //le stagiaire a 20 ans
echo  " le stagiaire a" . $x . " ans ";          //le stagiaire a 20 ans
echo  ' le stagiaire a $x ans ';          //le stagiaire a $x ans
echo  ' le stagiaire a' . $x . 'ans ';    //le stagiaire a 20 ans

?>
```

Les opérateurs arithmétiques

- Les opérateur arithmétiques vont nous permettre de faire des calculs:
- +, -, *, /, %

Les opérateurs d'affectation

- Les opérateurs d'affection permettent d'affecter une valeur ou une expression à une variable
- =, +=, -=, *=, /=, %= , => (utilisé dans les tableaux à voir plus tard) , -> (utilisé dans la POO)

les opérateurs d'incrémentation et décrémentation

- $x++, ou ++$x, $x—et --$x

Exemple:

```
$x=5;
$y=$x++;      //$x=6 et $y=5
$t=++$x;      //$x=7 et $t=7
```

Même raisonnement pour --$x et $x--

Les opérateurs de comparaison

Exemple	Nom	Résultat
$x==$y	Egal	Vrai si les variables $x et $y contiennent des valeurs égaux.
$x===$y	Identique	La valeur de $x égale à $y, en plus $x a le même type que $y
$x!=$y	Non égale	Vrai si les variables $x et $y ne contiennent des valeurs égaux.
$x!==$y	Non identique	La valeur de $x est différente de la valeur de $y, en plus $x n'a le même type que $y
$x<>$y	Non égal	
$x<$y	Inférieur	
$x>$y	supérieur	
$x<=$y	Inférieur ou égal	
$x>=$y	Supérieur ou égal	

Les opérateurs logiques

- and : **ET** logique
- or : **OU** logique
- ! : **NOT** (non)
- && : **ET** logique
- || : **OU** logique

L'opérateur ternaire

- L'opérateur ternaire c'est l'opérateur (? :)

(expression1) ? expression2: expression3;

- Si l'expression logique expresion1 est vrai, c'est l'expression2 qui va être exécutée. Si expression1 est fausse, c'est l'expression3 qui est exécutée.
- ***Exemple***:

```
<?php
$a=$b=1;
$c=($a==$b)?10:20;              //ici $c contiendra la valeur 10
?>
```

Les structures conditionnelles (**If….elseif….else**)

- ***Syntaxe***:

```
if (condition){
bloc d'instruction;
}
else if(condition){
bloc d'instructions;
}
else {bloc d'instructions;}
```

Les structures conditionnelles (**If….elseif….else**)

- Exemple

```
<?php

$a=1;
$b=2;
if ($a == $b) { echo "$a et $b sont égaux";}
else {echo "$a et $b sont différents";}

?>
```

L'instruction switch

- Cette instruction s'appelle le choix multiple. Elle est utilisée lorsqu'on veut évaluer plusieurs valeurs pour la même variable.

- Syntaxe:

```
switch (variable){

        case valeur1:instructions; break;
        case valeur2: instructions; break;
        .....
        case valeurn: instructions; break;
        default: instructions; break;
}
```

La boucle while

- ***Syntaxe:***

```
while (condition){
Instructions;
}
```

- Tant que la condition est vrai, faire les instructions demandées

La boucle do…while

- ***Syntaxe***:

```
do {
instructions;
} while(condition);
```

- Faire les instructions tant que la condition est vrai

La boucle for

- ***Syntaxe:***

```
for (expression1; expression2; expression3){
instructions;
}
```

Instruction 1 c'est l'initialisation

Instruction 2 c'est la condition d'exécution de la boucle

Instruction 3 est l'incrémentation ou décrémentation

La boucle foreach

- Il existe la boucle **foreach** qui peut être utilisée avec les tableaux. On va voir sa syntaxe ultérieurement.

Fonctions de chaines de caractère

- **strlen($str):** retourne la longueur de la chaine $str
- **strrev($str):** inverse la chaine $str
- **strpos(chaine1,chaine2)**:cherche la position de chaine 2 dans chaine1. si elle n'est pas trouvée, la fonction retourne **false**
- **str_replace(chaine1,chaine2,chaine3)**:chaine2 remplace chaine1 dans chaine3
- **str_shuffle($str)**: mélange aléatoirement les caractères de la chaine $str

Fonctions utiles

- **strtolower**(chaine):convertir chaine en minuscule
- **strtoupper**(**chaine**):convertir chaine en majuscule
- **substr**(**chaine, indice**):extrait un sous chaine de la chaine à partir de l'indice indiqué
- **substr**(**chaine,indice,longueur**):extrait un sous chaine de la chaine à partir de l'indice indiqué et la longueur de la chaine finale est longueur
- **ctype_upper**(**chaine**)**:** vérifie si chaine est majuscule
- **ctype_lower**(**chaine**)**:**vérifie si chaine est minuscule

Fonctions mathématiques

- **min($x, $y,...):** retourne la plus petite valeur parmi les valeurs en paramètre
- **max($x, $y,...) :** retourne la plus grande valeur parmi les valeurs en paramètre
- **ceil($x):** retourne l'arrondi supérieur de $x. Il s'agit du nombre entier immédiatement supérieur ou égal $ x.
- **floor($x):** retourne l'arrondi inférieur de $x. Il s'agit du nombre entier immédiatement inférieur ou égal $ x.
- **round($x,$i):** retourne l'arrondi le plus proche de $x avec la précision $i.
- **pow($x,$y):** retourne $x à la puissance $y.
- **sqrt($x):** retourne la racine carrée de $x
- **rand($x, $y):** retourne un nombre aléatoire entre $x et $y