

Les bases de PHP

I) Variable : nom + type + valeur

```
$a = 5;           // nom: $a, type: integer , valeur: 5
$b = 'coucou';    // nom: $b , type: string, valeur: 'coucou'
$c = true;        // nom: $c, type: bool, valeur: true
```

Que peut-on faire avec ces variables ?

1) On peut afficher les variables

```
echo $b;
echo '<br>';
echo $a;
echo '<br>';
→      coucou
       5
```

2) On peut faire des opérations sur les variables

Des opérations mathématique sur les integer :

```
$addition = $a + 2;
$soustraction = $a - 3;
echo $addition;      → 7
echo '<br>';
echo $soustraction;  → 2
echo '<br>';
```

Des opérations de concaténations sur les string :

```
$phrase = $b . ' tout le monde !';
echo $phrase;      → coucou tout le monde !
echo '<br>';
```

3) on peut déboguer les variables avec var_dump

On peut utiliser **var_dump** pour connaître le type et la valeur d'une variable.

```
var_dump($b);      → string(6) "coucou"
```

II) Les conditions

```
$condition = false;
```

```
if ($condition) {
    // si c'est vrai on fait ce truc dans cette accolade
    echo 'Ceci est vrai !';
} else {
    // si c'est faux on fait ce truc dans cette accolade
    echo 'Ceci est faux !';
}
```

→ Ceci est faux !

Le condition ternaire

```
$condition = false;
if ($condition) {
    $var = "Valeur si vrai";
} else {
    $var = "Valeur si faux";
}
```

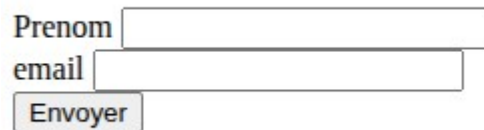
L'expression ternaire ci-dessous est équivalente au if/else ci-dessus.

```
$var = $condition ? "Valeur si vrai" : "valeur si faux";
```

III) Les formulaires : transmission des données avec \$ POST

Formulaire HTML

```
<form action="" method="post">
  Prenom <input type="text" name="firstname"><br>
  email <input type="email" name="email"><br>
  <input type="submit" value="Envoyer">
</form>
```



La balise **form** doit avoir :

- un attribut **action** qui indique vers quelle page le formulaire est envoyé
- un attribut **method** qui indique la méthode d'envoi du formulaire **post** ou **get**

Les balises **input** doivent avoir :

- un attribut **type** qui détermine le type d'input : texte, email, date, tel, checkbox, submit, etc.
- un attribut **name** qui servira à récupérer la valeur saisie dans le formulaire

Gestion de la soumission du formulaire

Les informations saisies dans le formulaire sont récupérées dans la variable **\$_POST**.

Si on fait **var_dump(\$_POST)**, on a :

- avant soumission : **array(0){}**
- après soumission :
array(2) { ["firstname"]=> string(6) "camile" ["email"]=> string(20) "camile@hunik.academy" }

On peut récupérer les valeurs de chaque champ du formulaire individuellement :

- champs prenom : **\$_POST['firstname']**
- champs email : **\$_POST['email']**

IV) Les paramètres d'URL : transmission des données avec \$ GET

URL : **https://www.monSite.com/blog/article1?parametre1=valeur1¶metre2=valeur2**

Après **?** : les paramètres sont des données que l'on fait transiter dans l'URL

Les paramètres de l'URL peuvent être récupérés dans la variable **\$_GET**.

On peut récupérer les valeurs de chaque paramètre individuellement :

- parametre1 : **\$_GET['parametre1']** → valeur1
- parametre2 : **\$_GET['parametre2']** → valeur2

V) Les tableaux

Il existe d'autres types de variables

```
$var1 = 2.5;           // type float
$var2 = null;          // type : NULL
$array = [5, true, 3.5, "jean"]; // type : array ou tableau
```

Créer un tableau :

Un tableau est système de clef => valeur

Tableau numéroté : (les clefs sont des numéros)

```
$array = [5, true, 3.5, "jean"];
```

```
var_dump($array);
→ array(4) { [0]=> int(5) [1]=> bool(true) [2]=> float(3.5) [3]=> string(4) "jean" }
```

Tableau associatif :

```

$array = [
    'nom' => 'ghastine',
    'prenom' => 'camile',
    'age' => 45,
    'developpeur' => true
];

var_dump($array);
→ array(4) { ["nom"]=> string(8) "ghastine" ["prenom"]=> string(6) "camile" ["age"]=> int(45)
["developpeur"]=> bool(true) }

```

Récupérer une valeur du tableau :

On peut récupérer chaque valeur du à partir de la clef.

Tableau numéroté

```

$array = [5, true, 3.5, "jean"];
echo $array[3] ;      → jean

```

Tableau associatif

```

$array = [
    'nom' => 'ghastine',
    'prenom' => 'camile',
    'age' => 45,
    'developpeur' => true
];
echo $array['nom'] ;   → ghastine

```

Ajouter une valeurs à un tableau :

Tableau numéroté

```

$array = [5, true, 3.5, "jean"];
Ajouter à la suite du tableau
$array[] = ["tic"] ;
var_dump($array);
→ array(5) { [0]=> int(5) [1]=> bool(true) [2]=> float(3.5) [3]=> string(4) "jean", [4]=> string(3)
"tic" }
Ajouter à la position souhaitée
$array[10] = "rouge";
var_dump($array);
→ array(5) { [0]=> int(5) [1]=> bool(true) [2]=> float(3.5) [3]=> string(4) "jean", [4]=> string(3)
"tic" [10]=> string(5) "rouge" }

```

Tableau associatif

```

$array = [
    'nom' => 'ghastine',
    'prenom' => 'camile',
    'age' => 45,
    'developpeur' => true
];
$array['email'] = 'camile@free.fr' ;
var_dump($array);

```

```
→ array(3) { ["nom"]=> string(8) "Ghastine" ["prenom"]=> string(6) "Camile" ["age"]=> string(2) "49" ["email"]=> string(14) "camile@free.fr" }
```

Modifier les valeur d'un tableau :

Tableau numéroté

```
$array = [5, true, 3.5, "jean"];  
$array[3] = "Julie" ;  
var_dump($array);  
→ array(4) { [0]=> int(5) [1]=> bool(true) [2]=> float(3.5) [3]=> string(4) "Julie" }
```

Tableau associatif

```
$array = [  
    'nom' => 'ghastine',  
    'prenom' => 'camile',  
    'age' => 45,  
    'developpeur' => true  
];  
$array['prenom'] = 'joe' ;  
var_dump($array);  
→ array(4) { ["nom"]=> string(8) "ghastine" ["prenom"]=> string(3) "joe" ["age"]=> int(45)  
["developpeur"]=> bool(true) }
```

VI) Les boucles

Boucle while : « tant que »

```
$i = 0;           → On initialise la variable  
while ($i < 5) {  → On vérifie la condition (si la condition n'est pas réalisée on sort de  
    echo 'hello-' . $i;           la boucle)  
    echo '<br>';  
    $i++;           → On incrément la variable ($i++ <=> ajouter +1 à $i)  
}
```

Boucle for : « pour »

```
for ($i=0; $i < 5 ; $i++) {  → l'initialisation, la condition et l'incrémentation se font dans le for  
    echo 'hello-' . $i;  
    echo '<br>';  
}
```

Boucle for each: « pour chaque » - pour boucler sur les tableaux

```
$user =[  
    'nom' => 'Ghastine',           → ligne 1  
    'prenom' => 'Camile',         → ligne 2  
    'email' => 'camile@free.fr'   → ligne 3  
];  
  
foreach ($user as $key => $value) {  
    echo $key . ' : ' . $value . '<br>';  
}
```

La boucle foreach :

- commence par la première ligne et récupère :
 - sa clef (nom) et la stocke dans la variable \$key
 - sa valeur (Ghastine) et la stocke dans la variable \$value
- Puis, il passe à la ligne suivante et fait la même chose.
- La boucle s'arrête après la dernière ligne du tableau.

VI) Les fonctions

Lorsqu'on a besoin d'utiliser un code plusieurs fois, on le range dans une fonction.

```
function direBonjour()
{
    echo 'Bonjour';
}
```

On peut ensuite appeler la fonction dans le code

```
<?php
direBonjour() ;      → Bonjour
```

On peut écrire et appeler une fonction avec des paramètres :

```
function calculTVA(float $prix, float $tva)
{
    $valeurTVA = $prix * $tva / 100 ;

    return $valeurTVA;
}
```

On peut ensuite appeler la fonction dans le code

```
<?php
$prix = 100 ;
$tva = 20 ;
$valeurTVA1 = calculTVA($prix, $tva);
echo $valeurTVA1. ' €';           → 20 €
echo '<br>';
echo calculTVA(100, 5) . ' €';   → 5 €
```