PHP

- Il faut initialiser chaque morceau de code contenant du PHP avec des balises spécifiques.

```
<?php
    $mystring = "12";
    $myinteger = 20;
?>
```

- Vous pouvez inclure des fichiers php externes.

```
<?php
    include("monfichier.php");
?>
```

- PHP va automatiquement convertir les types des données autant que nécéssaire, mais si besoin vous pouvez spécifier un type précis pour passer au dessus du type définit par défaut avec PHP.

```
<?php
    $myString = "Vive le php";
    $myInteger = (integer)$myString;
    var_dump($myInteger)
?>
```

- isset() sert à vérifier si une variable est bien définie or not.

```
$petiteVar = 1;
if (isset($petiteVar)) {
    echo "Définie\n";
} else {
    echo "Pas définie... \n »;
}
```

- Une constante est une variable qui restera inchangée car elle ne peut être modifiée plus tard dans le code contrairement aux variables. Une constante prend deux paramètres : un nom, et une valeur.

```
<?php
    define('YEAR','2017');
    define("CURRENT_TIME", time());
?>
```

- Rappel sur les variables superglobales:

$_GLOBALS Toutes les variables globales dont les variables d'environnements
$_GET Envoyées avec une requêtes GET
$_POST Envoyées avec une requêtes POST
$_FILES Envoyées avec une requête pour l'upload de fichiers
$_COOKIE Stocke les variables de cookies
$_SESSION Stocke les variables de sessions
$_ENV Stocke les variables d'environnements

- var_dump() est un bon moyen de voir ce qu'il y a à l'intérieur d'une variable. C'est un bon réflexe à prendre en main pour debugger !

```
<?php
        $capitalcities['England'] = array("Capital"=>"London",
        "Population"=>40000000, "NationalSport"=>"Cricket");
        var_dump($capitalcities)
?>
```

- Conditions & boucles

```
if ($a > $b) {
        echo "a est plus grand que b";
} elseif ($a == $b) {
        echo "a est égal à b";
} else {
        echo "a est plus petit que b";
}

for ($i = 1; $i <= 10; $i++) {
        echo $i;
}
```

- Arrays/Tableaux associatifs : On appelle tableau associatif la relation clé - valeur. Vous pouvez spécifier à chaque fois un nom de clé spécifique (sauf de type float).

```
<?php
        $myarray = array("a"=>"Pommes", "b"=>"Oranges","c"=>"Poires");
        var_dump($myarray);
?>
```

- Trier des arrays

sort() Tri les éléments dans l'ordre alphabétique - Réécrit les clés
rsort() Tri les éléments dans l'ordre inverse - Réécrit les clés
asort() Tri les éléments dans l'ordre alphabétique - Garde les clés
arsort() Tri les éléments dans l'ordre inverse - Garde les clés
ksort() Tri les clés dans l'ordre alphabétique
krsort() Tri les clés dans l'ordre inverse
Arrays - Tableaux multidimensionnels

```
<?php
        $capitalcities['England'] = array("Capital"=>"London",
        "Population"=>40000000, "NationalSport"=>"Cricket");
        $capitalcities['France'] = array("Capital"=>"Paris",
        "Population"=>2440000, "NationalSport"=>"Football");
?>
```

- Sauvegarder un array

```
$array["a"] = "Oh";
$array["b"] = "Ha";
$array["c"] = "Hey";
// serialize un array, permet son enregistrement dans un fichier par exemple
```

$str = serialize($array);
// encode pour le convertir en une version safe pour le passer sur le web
$strenc = urlencode($str);Arrays
// decode
$arr = unserialize(urldecode($strenc));
var_dump($arr);

- Afficher les erreurs en local

// Afficher les erreurs à l'écran
ini_set('display_errors', 1);
// Enregistrer les erreurs dans un fichier de log
ini_set('log_errors', 1);
// Nom du fichier qui enregistre les logs (attention
aux droits à l'écriture)
ini_set('error_log', dirname(__file__) . '/log_error_php.txt');