TALIS BUSINESS SCHOOL 2019-2020

PHP & MySQL

Christopher Legrand - www.christopherlegrand.fr

Qu'est-ce que PHP ?

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.

Ce qui distingue PHP des langages de script comme le Javascript, est que le code est exécuté sur le serveur, générant ainsi le HTML, qui sera ensuite envoyé au client. Le client ne reçoit que le résultat du script, sans aucun moyen d'avoir accès au code qui a produit ce résultat.

Le grand avantage de PHP est qu'il est extrêmement simple pour les néophytes, mais offre des fonctionnalités avancées pour les experts.

Le langage PHP a été conçu pour créer des sites vivants dits "dynamiques"

Christopher Legrand - Talis Business School

Site Statique VS Site Dynamique

Il existe deux types de sites web : les sites statiques et les sites dynamiques.

Site Statique

Ce sont des sites réalisés uniquement à l'aide des langages HTML et CSS. Ils fonctionnent très bien mais leur contenu ne peut pas être mis à jour automatiquement.

Les sites statiques sont donc adaptés pour réaliser des sites «vitrine», pour présenter par exemple son entreprise.

Site Dynamique

Plus complexes, ils utilisent d'autres langages en plus de HTML et CSS, tels que PHP et MySQL. Le contenu de ces sites web est dit dynamique parce qu'il peut changer sans l'intervention du webmaster !

La plupart des sites web que vous visitez aujourd'hui sont des sites dynamiques.

Christopher Legrand - Talis Business School

Comment fonctionne un site web ?

Lorsque vous voulez visiter un site web, vous tapez son adresse dans votre navigateur web. Mais comment fait la page web pour arriver jusqu'à vous ?

Il faut savoir qu'Internet est un réseau composé d'ordinateurs. Ceux-ci peuvent être classés en deux catégories.

Les clients : ce sont les ordinateurs des utilisateurs. Votre ordinateur fait donc partie de la catégorie des clients. Chaque client représente un visiteur d'un site web.

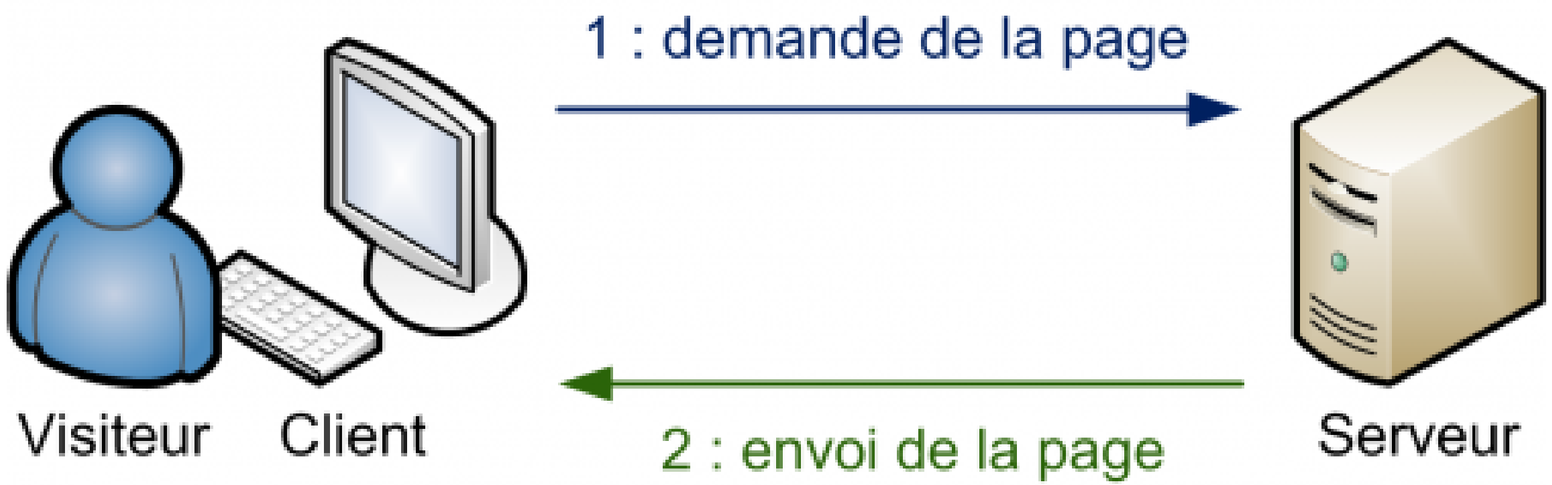
Les serveurs : Ce sont des ordinateurs puissants qui stockent et délivrent des sites web aux utilisateurs. Les serveurs sont indispensables au bon fonctionnement du Web.

Site statique

Lorsque le site est statique, le schéma est très simple.

Cela se passe en deux temps :

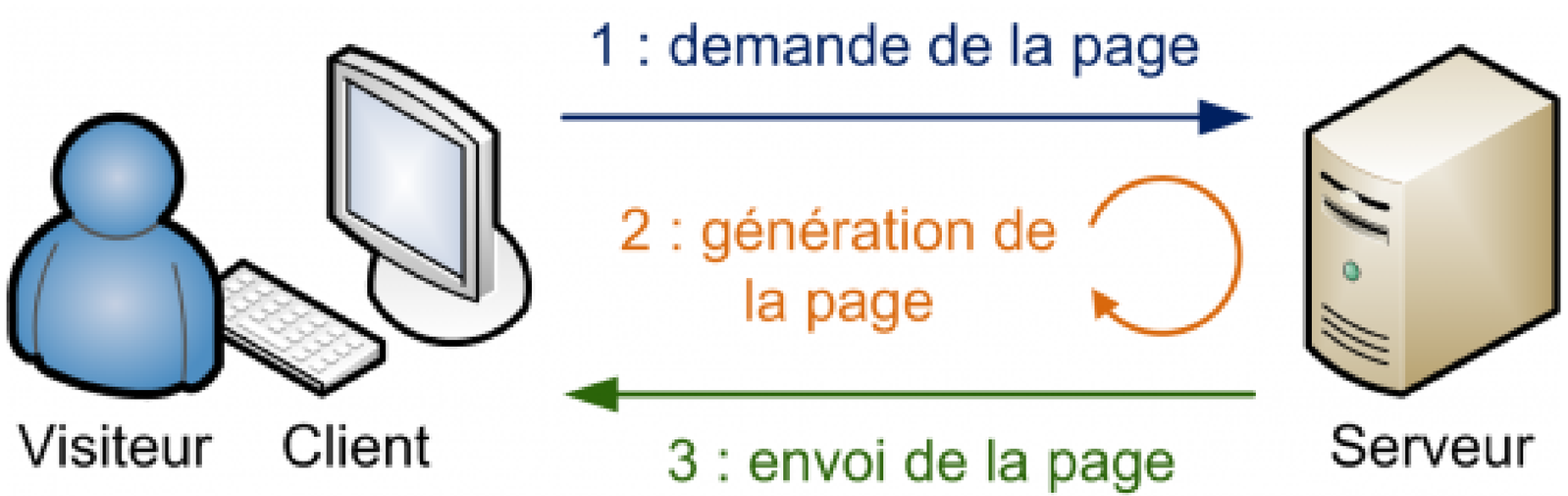
- Le client demande au serveur à voir une page web ;
- Le serveur lui répond en lui envoyant la page réclamée.



Site dynamique

Lorsque le site est dynamique, il y a une étape intermédiaire :

- Le client demande au serveur à voir une page web ;
- Le serveur prépare la page spécialement pour le client ;
- Le serveur lui envoie la page qu'il vient de générer.



PHP & MySQL

Quel que soit le site web que l'on souhaite créer, HTML et CSS sont indispensables.

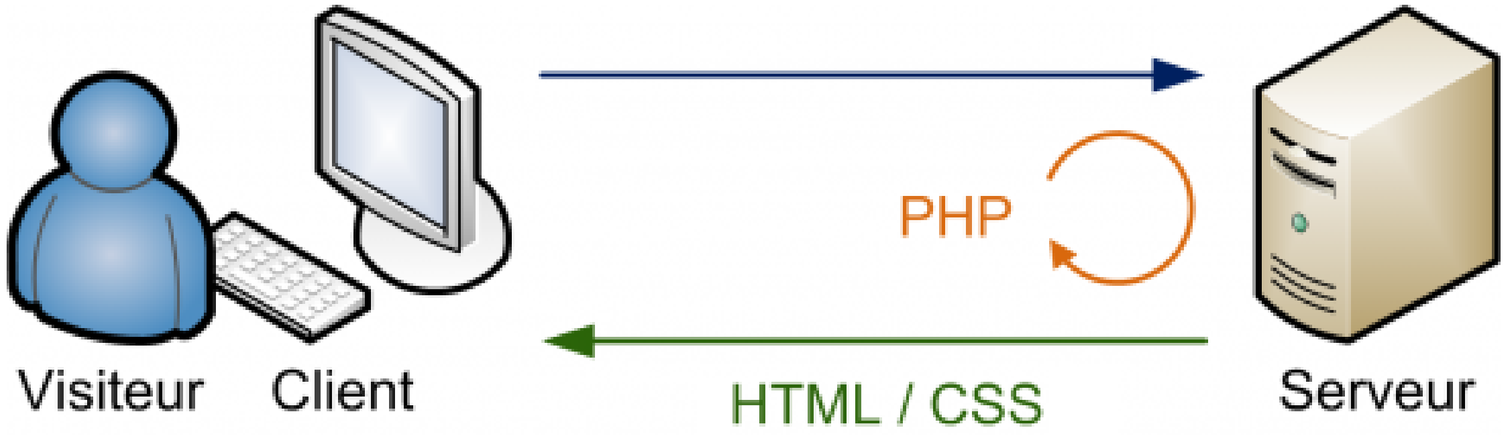
Cependant, ils ne suffisent pas pour réaliser des sites dynamiques. Il faut les compléter avec d'autres langages. Et c'est la que des langages comme PHP et MySQL interviennent.

- PHP : c'est un langage que seuls les serveurs comprennent et qui permet de rendre votre site dynamique. C'est PHP qui « génère » la page web comme on l'a vu sur un des schémas précédents.
- MySQL : c'est ce qu'on appelle un SGBD (Système de Gestion de Base de Données). Son rôle est d'enregistrer des données de manière organisée afin de vous aider à les retrouver facilement plus tard. C'est grâce à MySQL que vous pourrez enregistrer la liste des membres de votre site, les messages postés sur le forum, etc. Le langage qui permet de communiquer avec la base de données s'appelle le SQL.

Comment fonctionne PHP ?

Les navigateurs sont incapables de comprendre le code PHP : ils ne connaissent que le HTML et le CSS. Seul le serveur est capable de lire du PHP.

Le rôle de PHP est justement de générer du code HTML, qui sera ensuite envoyé au client de la même manière qu'un site statique :



Préparer son environnement de travail

Pour que votre ordinateur puisse lire du PHP, il faut qu'il se comporte comme un serveur.

Il suffit simplement d'installer les mêmes programmes que ceux que l'on trouve sur les serveurs qui délivrent les sites web aux internautes

- **Apache** : c'est ce qu'on appelle un serveur web. Il s'agit du plus important de tous les programmes, car c'est lui qui est chargé de délivrer les pages web aux visiteurs. Cependant, Apache ne gère que les sites web statiques (il ne peut traiter que des pages HTML). Il faut donc le compléter avec d'autres programmes.

- **PHP** : c'est un plug-in pour Apache qui le rend capable de traiter des pages web dynamiques en PHP. En clair, en combinant Apache et PHP, notre ordinateur sera capable de lire des pages web en PHP.

- **MySQL** : c'est le logiciel de gestion de bases de données. Il permet d'enregistrer des données de manière organisée (comme la liste des membres d'un site). Nous n'en aurons pas besoin immédiatement, mais autant l'installer de suite.

Préparer son environnement de travail

Il existe plusieurs paquetages tout prêts pour Windows. Je vous propose d'utiliser WAMP Server qui a l'avantage d'être régulièrement mis à jour et disponible en français.

Téléchargement du package : http://www.wampserver.com/

Lorsque vous lancez WAMP, une icône doit apparaître en bas à droite de la barre des tâches.

Vous pouvez alors lancer la page d'accueil de WAMP. Faites un clic gauche sur l'icône de WAMP, puis cliquez sur Localhost.

La page web que vous voyez à l'écran vous a été envoyée par votre propre serveur Apache que vous avez installé en même temps que WAMP. Vous êtes en train de simuler le fonctionnement d'un serveur web sur votre propre machine.

Pour le moment, vous êtes le seul internaute à pouvoir y accéder.
On dit que l'on travaille « en local ».

Les balises PHP

Vous savez que le code source d'une page HTML est constitué de balises. Par exemple, est une balise.

Le code PHP vient s'insérer au milieu du code HTML. On va progressivement placer dans nos pages web des morceaux de code PHP à l'intérieur du HTML.

Ces bouts de code PHP seront les parties dynamiques de la page, c'est-à-dire les parties qui peuvent changer toutes seules (c'est pour cela qu'on dit qu'elles sont dynamiques).

Pour utiliser du PHP, on va devoir introduire une nouvelle balise... et celle-ci est un peu spéciale. Elle commence par <?php et se termine par ?>. C'est à l'intérieur que l'on mettra du code PHP

L'instruction "echo"

Le PHP est un langage de programmation, ce qui n'était pas le cas du HTML (on parle plutôt de langage de description, car il permet de décrire une page web).

Tout langage de programmation contient ce qu'on appelle des instructions. On en écrit une par ligne en général, et elles se terminent toutes par un point-virgule. Une instruction commande à l'ordinateur d'effectuer une action précise.

La première instruction que nous allons découvrir permet d'insérer du texte dans la page web. Il s'agit de l'instruction "echo", la plus simple et la plus basique de toutes les instructions que vous devez connaître.

L'instruction "echo" demande à PHP d'insérer à un endroit précis le texte que vous demandez.

```php
<?php echo "Ceci est du texte écrit grâce à PHP "?>
```

Configurer PHP pour afficher les erreurs

Par défaut, PHP n'affiche pas les erreurs pour éviter de donner trop d'indications aux utilisateurs pour des raisons de sécurité (un mantra à vous répéter : "moins l'utilisateur en sait sur mon application, mieux mon application se portera !").
La configuration de PHP se fait dans un fichier appelé "php.ini".

Pour connaitre l'ensemble des informations relatives au PHP utilisé par le serveur web, il existe une fonction PHP phpinfo(). Nous allons l'utiliser pour localiser le fichier de configuration pour que nous puissions le modifier.

Nous allons créer un fichier info.php, y inscrire la fonction phpinfo(), puis localiser la ligne concernant le fichier de configuration pour PHP (php.ini). Une fois localisé, il faudra l'ouvrir et le modifier:

Il faut s'assurer que les clés de configuration *error_reporting* et *display_errors* ont respectivement les valeurs *E_ALL* & *on*.

Dorénavant, en cas d'erreur, PHP nous affichera quel est notre erreur et la ligne concernée.

Les variables

Les variables sont un élément indispensable dans tout langage de programmation, et en PHP on n'y échappe pas. C'est au contraire quelque chose qui va nous simplifier la vie. Sans les variables, vous n'irez pas bien loin.

Les variables nous permettent de retenir temporairement des informations en mémoire. Avec elles, nous allons pouvoir par exemple retenir le pseudonyme du visiteur, effectuer des calculs et bien d'autres choses !

C'est à vous de créer des variables. Vous en créez quand vous en avez besoin pour retenir des informations.

Une variable est toujours constituée de deux éléments :

- Son nom : pour pouvoir la reconnaître, vous devez donner un nom à votre variable. Par exemple: age_du_visiteur;
- Sa valeur : c'est l'information qu'elle contient, et qui peut changer. Par exemple : 17.

Les variables

Les variables sont capables de stocker différents types d'informations. On parle de types de données. Voici les principaux types à connaître :

- *Les chaînes de caractères (string)* : les chaînes de caractères sont le nom informatique qu'on donne au texte. Tout texte est appelé chaîne de caractères. En PHP, ce type de données a un nom :string. Exemple : "Je suis un texte".
- *Les nombres entiers (int) :* ce sont les nombres du type 1, 2, 3, 4, etc. On compte aussi parmi eux les entiers relatifs : -1, -2, -3...Exemple : 42.
- *Les nombres décimaux (float)* : ce sont les nombres à virgule, comme 14,738. On peut stocker de nombreux chiffres après la virgule, ce qui devrait convenir pour la plupart des usages que vous en ferez. Attention, les nombres doivent être écrits avec un point au lieu de la virgule (c'est la notation anglaise).Exemple : 14.738.
- *Les booléens (bool)* : c'est un type très important qui permet de stocker soit vrai soit faux. Cela permet de retenir si une information est vraie ou fausse. On les utilise très fréquemment. On écrit true pour vrai, et false pour faux.
- *Rien (NULL)* : aussi bizarre que cela puisse paraître, on a parfois besoin de dire qu'une variable ne contient rien. Rien du tout. On indique donc qu'elle vaut NULL. Ce n'est pas vraiment un type de données, mais plutôt l'absence de type.

Les variables

Voici les différentes étapes à respecter pour l'écriture d'une variable :

- D'abord, on écrit le symbole « dollar » ($), il précède toujours le nom d'une variable.
- Ensuite, il y a le signe « égal » (=) : c'est pour dire que $age_du_visiteur est égal à...
- À la suite, il y a la valeur de la variable, ici 17.
- Enfin, il y a l'incontournable point-virgule (;) qui permet de terminer l'instruction.

```php
$ageDuVisiteur = 17; // La variable est créée et vaut 17
$ageDuVisiteur = 23; // La variable est modifiée et vaut 23
$ageDuVisiteur = 55; // La variable est modifiée et vaut 55
```

```php
$nomDuVisiteur = "TalisBS"; // La variable est créée et vaut TalisBS
```

```php
$poidsDuVisiteur = 57.3;
```

```php
$japprendLePHP = true;
```

```php
$pasDeValeur = NULL;
```

Les variables

Nous avons appris à créer des variables et à stocker des informations à l'intérieur. Mais pour le moment, aucun de nos codes source n'affiche quoi que ce soit.
On peut se servir de l'instruction "echo" pour afficher la valeur d'une variable !

```php
$ageDuVisiteur = 17;

echo $ageDuVisiteur;
```

Ecrire 17 comme ça n'est pas très glamour, on aimerait écrire du texte autour pour dire : « Le visiteur de ce site a 17 ans ». La **concaténation** est justement un moyen d'assembler du texte et des variables.

C'est cette méthode qu'utilisent la plupart des programmeurs expérimentés en PHP. En effet, le code est plus lisible, on repère bien la variable.

```php
$ageDuVisiteur = 17;
echo 'Le visiteur de ce site a ' . $ageDuVisiteur . ' ans';
```

Les variables

On va maintenant faire travailler votre ordinateur, et vous allez voir qu'il encaisse les calculs sans broncher. Eh oui, PHP sait aussi faire des calculs !
On ne va travailler que sur des variables qui contiennent des nombres.

Les signes à connaître pour faire les quatre opérations de base sont les suivants :

- + => Addition;
- - => Soustraction
- * => Multiplication
- / => Division

```php
$addition = 1 + 4; // $addition prend la valeur 5
$soustraction = 4 - 1; // $soustraction prend la valeur 3
$multiplication = 2 * 5; // $multiplication prend la valeur 10
$division = 20 / 2; // $division prend la valeur 10
```

Les conditions

On a souvent besoin d'afficher des choses différentes en fonction de certaines données. Par exemple, si c'est le matin, vous voudrez dire « Bonjour » à votre visiteur ; si c'est le soir, il vaudra mieux dire « Bonsoir ».

C'est là qu'interviennent les conditions. Elles permettent de donner des ordres différents à PHP selon le cas. Pour notre exemple, on lui dirait : Si c'est le matin, affiche « Bonjour ». Sinon, si c'est le soir, affiche « Bonsoir ».

Les conditions constituent vraiment la base pour rendre votre site dynamique.

Une condition peut être écrite en PHP sous différentes formes. On parle de structures conditionnelles.

Les symboles à connaitre

- == Est égal à
- > Est supérieur à
- < Est inférieur à
- >= Est supérieur ou égal à
- <= Est inférieur ou égal à
- != Est différent de

Ici, le double égal sert à tester l'égalité, à dire « Si c'est égal à... ».Dans les conditions, on utilisera toujours le double égal (==).

A ne surtout pas confondre avec le simple égal (=) que l'on a vu dans le chapitre sur les variables et qui permet d'attribuer une valeur.

if... else

Pour écrire une condition, nous devons respecter les étapes suivantes :

- Pour introduire une condition, on utilise le mot "if", qui en anglais signifie « si ».
- On ajoute à la suite entre parenthèses la condition.
- Puis on ouvre des accolades à l'intérieur desquelles on placera les instructions à exécuter si la condition est remplie.

```php
$age = 16;

if ($age <= 17)
{
    echo "Trop jeune pour visiter ce site ! ";
}
```

Dans notre exemple, on travaille sur la variable $age. Ce qui compte ici, c'est qu'il y a deux possibilités :

- Soit la condition est remplie (l'âge est inférieur ou égal à 12 ans) et alors on affiche des instructions
- Soit on saute les instructions entre accolades, on ne fait rien.

if... else

Améliorons notre exemple. On va afficher un autre message si l'âge est supérieur ou égal à 17 ans. En clair, on demande : Si l'âge est inférieur ou égal à 17 ans, fais ceci, sinon fais cela.

```php
$age = 16;

if ($age <= 17) // SI AGE INFERIEUR OU EGAL A 17, ALORS ....
{
    echo "Trop jeune pour visiter ce site ! ";
}
else // SINON
{
    echo "Vous avez l'age requis pour accéder à ce site ! ";
}
```

if... else : Les booléens

Les booléens sont particulièrement utiles avec les conditions !

```php
$autorisationEntrer = true;

if ($autorisationEntrer == true)
{
    echo "Bienvenue sur le site ! )";
}
elseif ($autorisationEntrer == false)
{
    echo "Vous ne pouvez pas accèder au site !";
}
```

```php
$autorisationEntrer = true;

if ($autorisationEntrer)
{
    echo "Bienvenue sur le site ! )";
}
else
{
    echo "Vous ne pouvez pas accèder au site !";
}
```

if... else : Les conditions multiples

Il est également possible de vérifier plusieurs conditions à la fois. Pour cela, on aura besoin de nouveaux mots-clés. Voici les principaux à connaître :

- AND qui signifie **Et** et peut s'écrire **&&**
- OR qui signifie **Ou** et peut s'écrire **||**

```php
$age = 8;
$langue = "anglais";


if ($age <= 12 AND $langue == "français")
{
    echo "Bienvenue sur mon site !";
}
elseif ($age <= 12 AND $langue == "anglais")
{
    echo "Welcome to my website!";
}
```

```php
$pays = "France";

if ($pays == "France" OR $pays == "Belgique")
{
    echo "Bienvenue sur notre site !";
}
else
{
    echo "Service non accessible !";
}
?>
```

Les conditions : Switch

Il existe une autre structure qui va nous permettre de simplifier des traitements

```php
<?php
$note = 7;

if ($note == 0)
{
    echo "SHAME ! ";
}

elseif ($note == 5)
{
    echo "Ouh le nul ! ";
}

elseif ($note == 7)
{
    echo "Un peu moins nul que celui qui à eu 5";
}
elseif ($note == 15)
{
    echo "Très bon travail !";
}
// etc...
else
{
    echo "Désolé, je n'ai pas de message à afficher p
}
?>
```

```php
$note = 7;

switch ($note) // on indique sur quelle variable on travail
{
    case 0: // dans le cas où $note vaut 0
        echo "SHAME ! ";
        break;

    case 5: // dans le cas où $note vaut 5
        echo "Ouh le nul ! ";
        break;

    case 7: // dans le cas où $note vaut 7
        echo "Un peu moins nul que celui qui à eu 5";
        break;

    case 15:
        echo "Très bon travail !";
        break;

    default:
        echo "Désolé, je n'ai pas de message à afficher pou
}
?>
```

Christopher Legrand - Talis Business School

Exercices

Les boucles

Une boucle et une structure qui fonctionne sur le même principe que les conditions. Il y a d'ailleurs beaucoup de similitudes avec les conditions. Concrètement, une boucle permet de répéter des instructions plusieurs fois. C'est un gain de temps, c'est très pratique, et bien souvent indispensable.

Imaginez que vous êtes en train de créer un forum. Sur une page, on affiche par exemple une trentaine de messages. Il serait bien trop long et répétitif de dire « Affiche le message 1 et le nom de son auteur », « Affiche le message 2 et le nom de son auteur », « Affiche le message 3 et le nom de son auteur », etc.

Pour éviter d'avoir à faire cela, on peut utiliser un système de boucle qui nous permettra de dire une seule fois : « Affiche 30 messages et le nom de leur auteur à chaque fois ».

La boucle while()

Voici comment faire avec une boucle simple : while().

Quel que soit le type de boucle (while ou for), il faut indiquer une condition. Tant que la condition est remplie, les instructions sont réexécutées. Dès que la condition n'est plus remplie, on sort de la boucle.

while peut se traduire par « tant que ».
Ici, on demande à PHP : TANT QUE $nombreDeLignes est inférieur ou égal à 50, exécute ces instructions.

```php
$nombreDeLignes = 1;

while ($nombreDeLignes <= 50)
{
    echo 'J\'écouterai mon prof en cours de PHP. <br>';
    $nombreDeLignes++; // $nombreDeLignes = $nombreDeLignes + 1
}
```

La boucle for()

for et while donnent le même résultat et servent à la même chose : répéter des instructions en boucle. L'une peut paraître plus adaptée que l'autre dans certains cas, cela dépend aussi des goûts.

For ressemble beaucoup au while, mais c'est la première ligne qui est un peu particulière.

```php
for ($i = 1; $i <= 100; $i++) {
    echo 'Je suis la ligne n°' . $i . '<br>';
}
```

Décrivons chacun de ces éléments :

- Le premier sert à l'initialisation. C'est la valeur que l'on donne au départ à la variable.
- Le second, c'est la condition. Tant que la condition est remplie, la boucle est réexécutée. Dès que la condition ne l'est plus, on en sort.
- Enfin, le troisième c'est l'incrémentation, qui vous permet d'ajouter +1 à la variable à chaque itération

Les tableaux

Avant toute chose, il est bon de préciser qu'un tableau PHP et un tableau HTML sont deux choses complètement différentes. Un tableaux PHP a pour fonction de stocker et manipuler des informations tandis qu'un tableau HTML sert à présenter des données sur un écran.

Nous allons voir qu'il est possible d'enregistrer de nombreuses informations dans une seule variable grâce aux tableaux. On en distingue deux types :

- les tableaux indexés;
- les tableaux associatifs.

Les tableaux : Syntaxe

Un tableau peut être créé en utilisant la structure de langage array(). Il prend un nombre illimité de paramètres, chacun séparé par une virgule, sous la forme d'une paire key => value. Depuis PHP 5.4, la structure array() peut être remplacé par []

```php
$array = array(
    key      => value,
    key2     => value2,
    key3     => value3
);
```

```php
$array = [
    key      => value,
    key2     => value2,
    key3     => value3
];
```

Christopher Legrand - Talis Business School

Les tableaux indexés

Un tableau indexé numériquement est tout simplement une liste d'éléments repérés chacun par un index numérique unique.

Le premier élément du tableau sera repéré par l'index 0, le second par l'index 1, le troisième par l'index 2 et ainsi de suite.

Pour accéder à un élément du tableau, il suffit d'y faire référence de cette manière : $tableau[0], $tableau[1], $tableau[2]...

```php
// Déclaration d'un tableau indexé
$legumes = ['carotte', 'poivron', 'aubergine' ,'chou'];
// Ajout d'un légume dans le tableau
$legumes[] = 'salade';
// Affichage du chou
echo $legumes[3];
```

Les tableaux associatif

Un tableau associatif est un tableau composé de couples : clé chainée / valeur.

Il est apparu pour pallier les faiblesses du tableau à index numériques. En effet pour ce dernier, il faut absolument connaître son emplacement pour atteindre la valeur et pour un développeur, ce n'est pas toujours le cas.

A chaque clé est référencée une valeur :

```php
$identite = [
    "nom"       => "Legrand",
    "prenom"    => "Christopher",
    "age"       => 19
];

echo 'Nom : ' . $identite['nom'] . '<br/>';
echo 'Prénom : ' . $identite['prenom'] . '<br/>';
echo 'Age : ' . $identite['age'] . ' ans<br/>';
```

Les tableaux : Parcourir un tableau

Il y a une autre syntaxe de la boucle foreach() qui permet de récupérer la valeur ET la clé !
C'est très pratique et vous l'utiliserez régulièrement.

```php
$identite = [
    "Nom"       => "Legrand",
    "Prenom"    => "Christopher",
    "Age"       => 24
];

foreach($identite as $cle => $valeur){
    echo $cle . ' vaut ' . $valeur . '<br>';
}
```

Les tableaux : Parcourir un tableau

Comme dans tout autre langage de programmation, le parcours de tableau se fait à l'aide de boucles.

PHP dispose de sa propre structure de contrôle pour parcourir le contenu d'un tableau.
Il s'agit de la structure foreach().

C'est une boucle particulière qui avance le pointeur du tableau à chaque itération. Celle-ci a été intégrée depuis la version 4 de PHP et se présente sous deux syntaxes possibles :

```php
// Déclaration d'un tableau indexé
$legumes = ['carotte', 'poivron', 'aubergine' ,'chou'];

foreach ($legumes as $legume){
    echo $legume . '<br>';

}
```

Les tableaux : Afficher le contenu d'un tableau

Lorsque l'on développe, il arrive très souvent que l'on veuille afficher le contenu d'un tableau dans le but de pouvoir débugguer un programme.

Pour cela, PHP introduit la fonction print_r() qui assure cette fonction.

Afin de respecter l'indentation à l'affichage, nous préfixons le résultat de cette fonction par les balises <pre> et </pre>.

```php
// Déclaration d'un tableau indexé
$legumes = ['carotte', 'poivron', 'aubergine' ,'chou'];

echo '<pre>';
print_r($legumes);
echo '</pre>';
```

Les tableaux : Rechercher dans un tableau

Nous allons maintenant faire des recherches dans des arrays.

Cela vous sera parfois très utile pour savoir si votre array contient ou non certaines informations.
Nous allons voir trois types de recherches, basées sur des fonctions PHP :

- array_key_exists : pour vérifier si une clé existe dans l'array ;
- in_array : pour vérifier si une valeur existe dans l'array ;
- array_search : pour récupérer la clé d'une valeur dans l'array.

Les tableaux : array_key_exists

Voici notre problème : on a un array, mais on ne sait pas si la clé qu'on cherche s'y trouve .Pour vérifier ça, on va utiliser la fonction array_key_exists qui va parcourir le tableau pour nous et nous indiquer s'il contient cette clé.

La fonction renvoie un booléen, c'est-à-dire true, si la clé est dans l'array, et false si la clé ne s'y trouve pas. Ça nous permet de faire un test facilement avec une structure conditionnelle :

```php
$identite = [
    "Nom"       => "Legrand",
    "Prenom"    => "Christopher",
    "Age"       => 24
];

if (array_key_exists('Nom', $identite)) {
    echo "La clé Nom se trouve dans les coordonnées ! <br>";
}

if (array_key_exists('Prenom', $identite)) {
    echo "La clé Prénom se trouve dans les coordonnées !";
}
```

Les tableaux : in_array

Le principe est le même que array_key_exists mais cette fois on recherche dans les valeurs. in_array renvoie true si la valeur se trouve dans l'array,false si elle ne s'y trouve pas.

```php
$identite = [
    "Nom"       => "Legrand",
    "Prenom"    => "Christopher",
    "Age"       => 24
];

if (in_array('Legrand', $identite )) {
    echo "Le nom Legrand est bien présent dans la BDD <br>";
} else {
    echo "Le nom Legrand n'est pas présent dans la BDD <br>";
}

if (in_array('Christopher', $identite )) {
    echo "Le prenom Christopher est bien présent dans la BDD <br>";
} else {
    echo "Le prenom Christopher n'est pas présent dans la BDD <br>";
}
```

Les tableaux : array_search

array_search fonctionne comme in_array : il travaille sur les valeurs d'un array. Voici ce que renvoie la fonction :

- si elle a trouvé la valeur, array_search renvoie la clé correspondante (c'est-à-dire le numéro si c'est un array numéroté, ou le nom de la clé si c'est un array associatif);
- si elle n'a pas trouvé la valeur, array_search renvoie false.

```php
$fruits = ['Banane', 'Cerise', 'Citron', 'Pêche'];

$position = array_search('Cerise', $fruits);
echo 'Cerise se trouve en position ' . $position . '<br>';

$position = array_search('Pêche', $fruits);
echo 'Pêche se trouve en position ' . $position . '<br>';
```

Christopher Legrand - Talis Business School

Les fonctions

Une fonction est une série d'instructions qui effectue des actions et qui retourne une valeur.

En général, dès que vous avez besoin d'effectuer des opérations un peu longues dont vous aurez à nouveau besoin plus tard, il est conseillé de vérifier s'il n'existe pas déjà une fonction qui fait cela pour vous. Et si la fonction n'existe pas, vous avez la possibilité de la créer.

Imaginez que les fonctions sont des robots, vous ne savez pas ce qui se passe à l'intérieur mais vous pouvez appuyer sur un bouton pour lui demander de faire quelque chose de précis.
Avec les fonctions, c'est le même principe !

Les fonctions

Bien que PHP propose des centaines et des centaines de fonctions, parfois il n'y aura pas ce que vous cherchez et il faudra écrire vous-mêmes la fonction.
C'est une façon pratique d'étendre les possibilités offertes par PHP.

```php
function DireBonjour($nomClient) {
    echo "Bonjour " . $nomClient . ' ! :) <br>
    Je suis ravi de vous voir sur mon site !';
}

?>

<p><?php DireBonjour("Christopher"); ?></p>
```

Les fonctions : Appeler une fonction

En PHP, appeler une fonction est très simple => Par son nom ! =>

```php
<?php
calculSquare();
?>
```

Comme vous le voyez, j'ai simplement écrit le nom de la fonction, suivi de parenthèses vides, puis d'un point-virgule.

En faisant cela, j'appelle la fonction calculSquare mais je ne lui envoie aucune information, aucun paramètre. Certaines fonctions peuvent fonctionner sans paramètres, mais elles sont assez rares.

Si on veut lui envoyer un paramètre (un nombre, une chaîne de caractères, un booléen...), il faut l'écrire entre les parenthèses :

```php
<?php
calculSquare(2);
?>
```

Les fonctions : Récupérer la valeur

En PHP, appeler une fonction est très simple => Par son nom ! =>

```php
<?php
calculSquare();
?>
```

Comme vous le voyez, j'ai simplement écrit le nom de la fonction, suivi de parenthèses vides, puis d'un point-virgule.

En faisant cela, j'appelle la fonction calculSquare mais je ne lui envoie aucune information, aucun paramètre. Certaines fonctions peuvent fonctionner sans paramètres, mais elles sont assez rares.

Si on veut lui envoyer un paramètre (un nombre, une chaîne de caractères, un booléen...), il faut l'écrire entre les parenthèses :

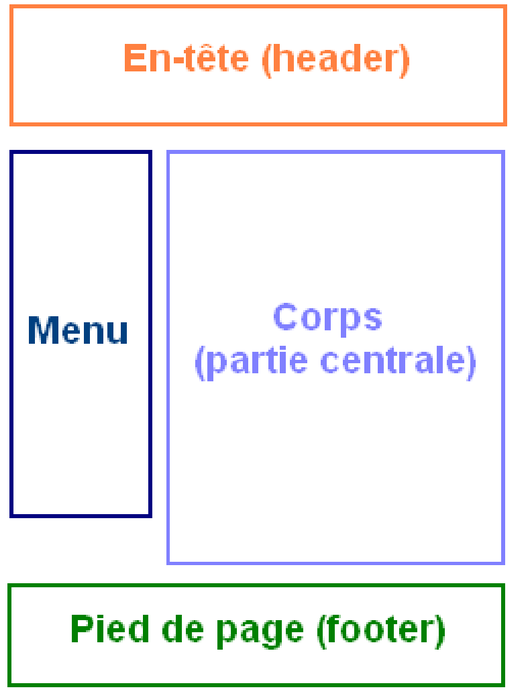
```php
function DireBonjour($nomClient) {
    echo "Bonjour " . $nomClient . ' ! :) <br>
    Je suis ravi de vous voir sur mon site !';
}

?>

<p><?php DireBonjour("Christopher"); ?></p>
```

Inclure des portions de page

Comme nous l'avons vu avec SASS, il est possible de découper proprement son code en pleins de fichiers différents avec PHP.

L'inclusion de pages est une des fonctionnalités les plus simples et les plus utiles de PHP. On peut très facilement inclure toute une page ou un bout de page à l'intérieur d'une autre.
Cela va grandement vous faciliter la tâche en vous évitant d'avoir à copier le même code HTML plusieurs fois.

La plupart des sites web sont généralement découpés selon ce schéma :



On était donc condamnés à copier-coller sur chaque page à l'identique :l'en-tête ; le menu ;le pied de page etc..

Cela donnait du code lourd et répétitif sur toutes les pages !

Inclure des portions de page

Le principe de fonctionnement des inclusions en PHP est plutôt simple à comprendre. Vous avez un site web composé de disons vingt pages.
Sur chaque page, il y a un menu, toujours le même.

Pourquoi ne pas écrire ce menu (et seulement lui) une seule fois dans une page header.php ?

En PHP, vous allez pouvoir inclure votre menu sur toutes vos pages. Lorsque vous voudrez modifier votre menu, vous n'aurez qu'à modifier header.php et l'ensemble des pages de votre site web sera automatiquement mis à jour !

Transmettez des données avec l'URL

URL signifie Uniform Resource Locator, et cela sert à représenter une adresse sur le web.
Toutes les adresses que vous voyez en haut de votre navigateur, comme https://christopherlegrand.fr sont des URL.

Nous allons maintenant apprendre à transiter des informations de pages en pages lorsque notre visiteur évolue sur notre site. Vous allez voir qu'il est possible de faire transiter des informations à travers la base d'adresse de l'utilisateur.

Avez-vous déjà remarqué dans les URL des caractères un peu spéciaux ? Des "?", des "=", etc.. ?
Tapez "cafe" dans Google et observez le résultat.

On peut voir l'url de google.fr/ puis q=cafe...&oq=cafe&aqs=chrome....&sourceid=chrome... etc..
C'est un système qu'on utilise sur le web pour faire transiter des informations d'une page à une autre

```
<p>
    <a href="bonjour.php?nom=Legrand&amp;prenom=Christopher">Dis moi bonjour</a>
</p>
```

```
<p>
    Bonjour <?php echo $_GET['nom'] . " " . $_GET['prenom'];?>
</p>
```

Transmettez des données avec l'URL

Attention, il ne faut ***JAMAIS*** faire confiance aux données qui nous sont envoyés par les visiteurs. Pourquoi ? Parce que certains visiteurs mal intentionnés pourraient modifier les informations qu'ils nous envoient dans le but de faire planter votre site voir même d'entre prendre le contrôle.

Il va donc falloir faire très attention à ce que nous allons faire transiter dans la barre d'adresse car il est très facile d'en modifier le contenu. Un petit test, enlevez la partie du prénom dans l'URL :)

Pour résoudre ce problème, on peut faire appel à une fonction un peu spéciale : isset().
Cette fonction teste si une variable existe. Nous allons nous en servir pour afficher un message spécifique si le nom ou le prénom sont absents.

```php
if (isset($_GET['nom']) AND isset($_GET['prenom'])) {
    echo "Bonjour " . $_GET['nom'] . ' ' . $_GET['prenom'];
}
else {
   echo "Il manque des informations pour afficher cette page";
}
```

Transmettez des données avec l'URL

De plus, dans le cas ou on demande un integer pour l'age par exemple, on peut imaginer que notre gentil visiteur veuille s'amuser et mettre un mot à la place de son age. Ce n'est pas ce que nous voulons, et nous voulons nous assurer qu'il mette bien un nombre.

Pour convertir explicitement une valeur en un entier, on peut utiliser le mot-clé (int) :

```php
$age = (int)$_GET['age'];
```

```php
if (isset($_GET['nom']) AND isset($_GET['prenom']) AND isset($_GET['age'])) {
    echo "Bonjour " . $_GET['nom'] . ' ' . $_GET['prenom'] . " " . $age . " ans";
}
else {
   echo "Il manque des informations pour afficher cette page";
}
```