



## PHP

### Objectifs

- ✓ Maîtriser la syntaxe et les instructions de base en PHP.

01

### A quoi sert le Php ? Qu'est-ce que c'est ?



Le PHP (Hypertext Preprocessor) est un langage de programmation (fonctionnel) à l'inverse des langages Html et Css qui sont des langages de conception (structure et mise en forme de page web).

Avec PHP, nous pouvons faire tous types de site et il n'y a plus de limite : moteur de recherche (type google), réseaux sociaux (type facebook), plateforme multimédia (type YouTube), site d'informations (type Wikipédia), boutique ecommerce (type amazon), petites annonces (type leboncoin), forum, blog, espace membre, etc.

Pour mieux comprendre, prenons le cas d'un formulaire :

1 formulaire est déclaré en html,

il est mis en forme en css (html + css = conception),

mais si nous voulons récupérer et exploiter les saisies postées par l'internaute nous aurons besoin du PHP qui est un langage permettant d'effectuer des traitements (et c'est bien là le principe, si nous faisons un formulaire c'est pour récupérer des données !).

Autrement dit, sans PHP nous ne pourrons pas faire fonctionner un formulaire.

Ensuite, si nous voulions sauvegardées ces données dans une base, nous aurons besoin du langage SQL.

D'autre part, le PHP permet de créer des sites web dynamiques ! Et c'est très utile !

02

### Histoire de Php

La première version du langage PHP fut créée en 1994 par Rasmus Lerdorf et repris en 1997 par deux étudiants : Andi Gutmans et Zeev Suraski. Dans le monde, plus de 300 millions de sites reposent sur la technologie Php. Ce chiffre ne cesse d'augmenter.

Je ne m'étais pas trop intéressé à l'histoire de PHP, pour ça il y a Wikipedia : [Plus d'informations sur le langage PHP avec Wikipedia](#)

03

### Qu'est-ce qu'un site web statique ?

Un site statique est généralement réalisé avec les langages Html et Css.

La problématique c'est que ces sites ne sont pas pratiques à mettre à jour (exemple de mise à jour : ajouter une image, modifier du texte, ajouter un nouveau lien comme rubrique de menu, créer une nouvelle page).

```
/* Faces-config.xml */ package example;
/**
 * @author user
 */
public class carBean {
    private java.lang.String carName;
    public carBean() {
    }
    public java.lang.String getCarName() {
    }
}
```

Vous me direz peut-être que c'est simple et rapide pour vous d'adapter votre code-source puisque vous êtes à l'aise en Html et Css mais ce n'est pas le cas de tous et encore moins des clients qui achèteront les sites web (ils sont généralement peu techniques et veulent pouvoir effectuer des réglages rapidement et efficacement).

Le principal inconvénient d'un site statique c'est qu'il faut quelques connaissances techniques



```
    return carName;
}
public void setCarName(java.lang.String carName) {
    this.carName = carName;
}
```

car nous sommes obligés de repasser par le code-source de chaque fichier pour effectuer des modifications.

Gardez bien à l'esprit qu'un site est modifié en permanence ! Même quand vous pensez qu'il est "terminé" il ne l'est pas vraiment. Il faut donc trouver des solutions pour faciliter les modifications et les mises à jour : c'est là que le site dynamique (propulsé par PHP) rentre en jeu!

04

## Qu'est-ce qu'un site web dynamique ?

Un site dynamique est un site dont les informations proviennent d'une base de données (pour cela il faudra mélanger le langage PHP et le langage SQL ensemble).



### ➤ Attention

Il ne faut pas confondre un site dynamique avec un site de mouvements ou d'animations et couleurs clinquantes dans la page web.

Pour obtenir un site dynamique, nous utiliserons le langage PHP, qui lui-même accueillera le langage SQL afin d'obtenir des informations contenues dans la base de données.

Si nous arrivons à récupérer des données (provenant d'une base de données) sur notre site, nous pourrions aussi récupérer ces mêmes données pour un 2e site.



### ➤ Pourquoi 2 sites ? quel en serait l'intérêt ?

- Le premier site sera destiné au grand public pour la consultation des informations par les internautes.  
un Front (aussi appelé, front-office ou front-end)
- Le deuxième site sera protégé par authentification et destiné à la mise à jour et modification de contenu par l'administrateur (propriétaire du site).  
un Back (aussi appelé, back-office, back-end ou interface de gestion).

Nous appelons donc « un site dynamique » un site dont les informations sont contenues dans une base de données et qui possède 2 interfaces :

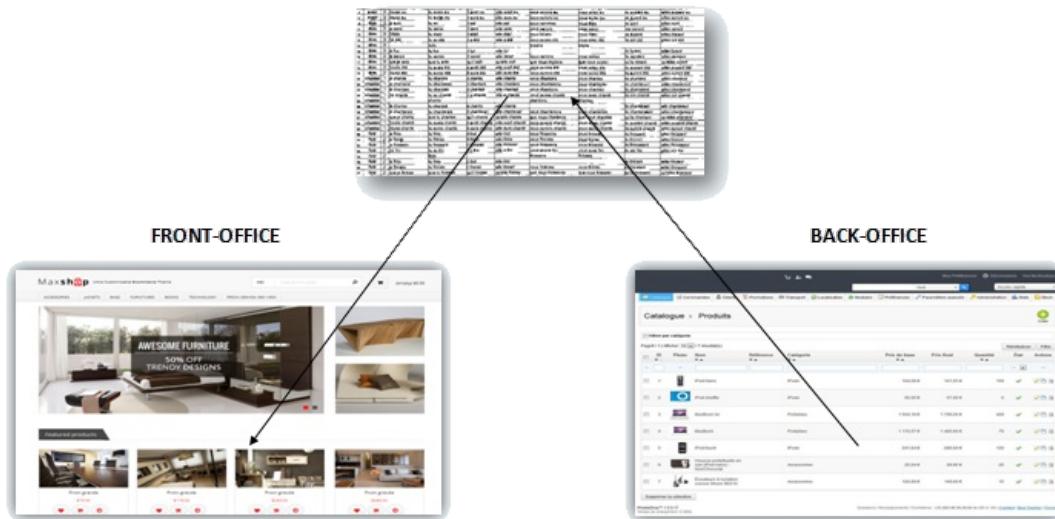
- un premier site FRONT (pour l'affichage du site normal),
- un deuxième site BACK (pour la gestion et assurer les réglages du premier site).

L'avantage d'un site dynamique c'est que les modifications seront beaucoup plus faciles et ne devront pas obligatoirement être réalisées dans le code-source d'un fichier par une personne connaissant le code et ayant des compétences techniques.

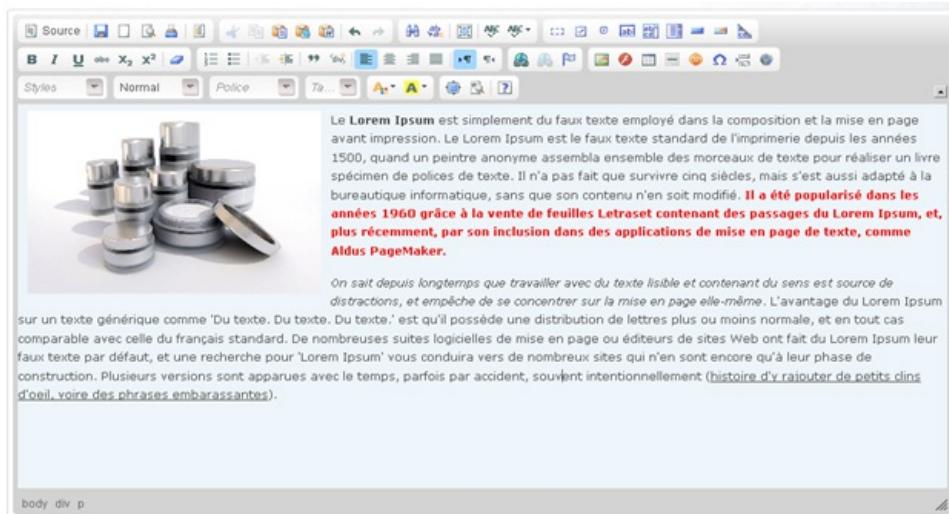
Les modifications pourront être effectuées dans une interface de gestion backOffice simple et intuitive, et par tous !

Voici un schéma illustrant cette situation :

## Base de données



Dans une interface de gestion (BackOffice), nous retrouvons la plupart du temps un éditeur avec des icônes permettant de gérer les contenus :



Par exemple l'icone permettant de créer un lien ajoutera automatiquement les balises `<a> </a>` dans le code. Sans que le développeur n'ait besoin de l'écrire.

Dans la plupart des cas, il n'y a pas qu'un seul site web mais 2 sites web (1 front et 1 back). C'est ce qu'on appelle un site web dynamique.

**i**

➤ Qu'est-ce qu'un site dynamique

Un site dynamique est un site modifiable sans avoir besoin de retourner dans le code-source obligatoirement.

Les sites web dynamiques sont majoritairement élaborés avec l'aide de php (d'autres langages de programmation peuvent être utilisés)

Le FRONT représente la partie que nous consultons en tant qu'internaute (pour l'affichage et la consultation des informations).

Le BACK réservé aux administrateurs du site (gérants) permettant les mises à jour des informations.

Si nous voulions faire la liste des sites dynamiques incluant du php et du sql, la liste serait longue voire interminable, puisque tous les sites (ou presque) sont concernés.

Prenons la chose à l'envers, quels sont les sites statiques (et donc non dynamiques) et n'incluant ni de Php ni de Sql ?

Des sites n'étant jamais mis à jour (assez rare), certains sites développés par les développeurs eux-mêmes pour se présenter (et encore ce n'est pas une majorité).

Contrairement aux idées reçues, la plupart du temps même un site vitrine est un site dynamique !

Les clients demandent un site vitrine car ils veulent se présenter mais aussi pouvoir effectuer des modifications via une interface de gestion BackOffice !

Sinon, sans connaissance technique, comment feraient-ils pour modifier les textes, les images et créer de nouvelles pages ?  
Retenez le bien : La plupart du temps un site vitrine est un site dynamique !

Maintenant vous le savez, PHP est incontournable !

PHP est actuellement utilisé en version 7 sous forme de code procédural ou orienté objet.



➤ En résumé

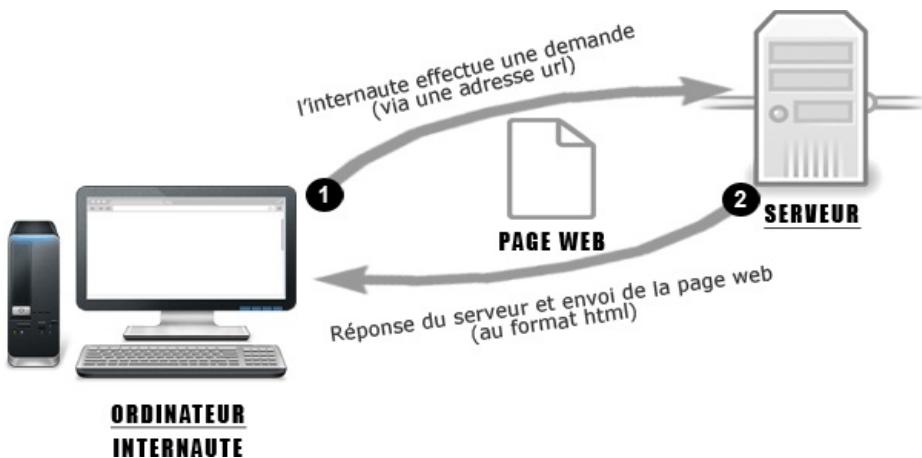
Pour construire votre page web vous aurez besoin de Html et Css (partie structure et mise en forme).

Pour la rendre fonctionnelle et dynamique vous aurez besoin de Php et Sql (pour effectuer des sauvegardes et mener des opérations sur la base de données).

JavaScript permettra davantage d'animer la page web et viendra compléter les autres langages pour la partie programmation.

05

## Architecture Client / Serveur avec un site statique



1. - L'internaute effectue une demande (via une adresse URL).

2. - Le serveur reçoit la demande et trouve le fichier (page web) intégrant uniquement du code Html et Css, il renvoie le contenu demandé à l'internaute.

C'est aussi simple que ça. Les sites statiques n'utilisent que du code html et css.

06

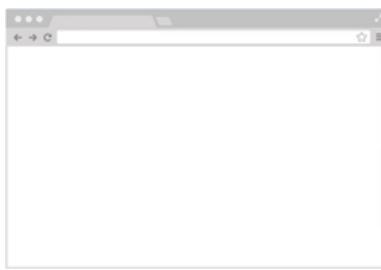
## Architecture Client / Serveur avec un site dynamique

### Comment fonctionne le Php ?

Lorsqu'une page web s'affiche sur votre écran, elle respecte un processus particulier décrit ci-dessous :

- 1- Un internaute souhaite accéder à une page web, il utilise un navigateur par lequel il sollicite une adresse url.
- 2- Cette adresse url pointe vers une ressource (fichiers contenant texte, image, code-source, etc.) hébergée par un serveur (emplacement de stockage applicatif).
- 3- Il arrive parfois que le serveur ait besoin de consulter une base de données pour intégrer des informations dans le contenu de la page web (requête SQL).

1 - navigateur de l'internaute :



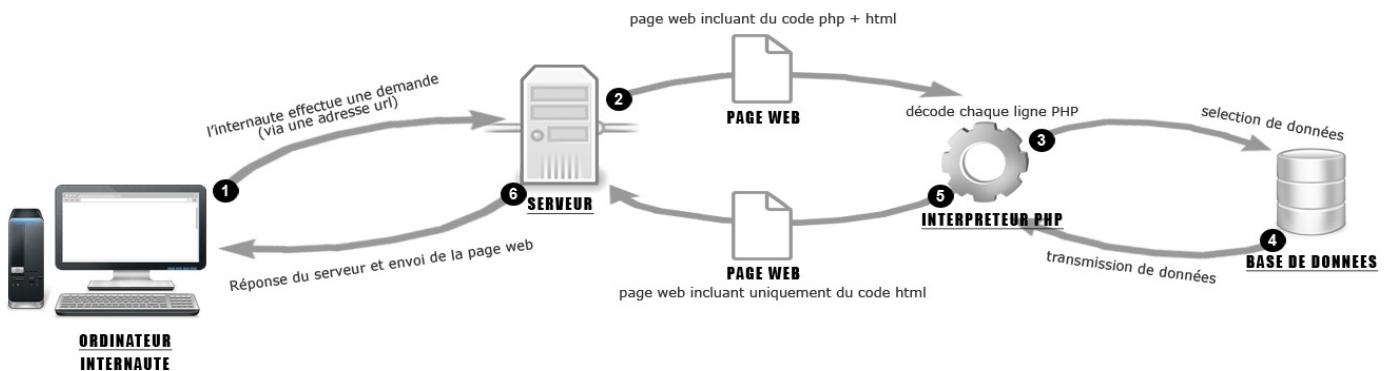
2 - serveur stockant la ressource :



3 - données de la base :

s	avoir	3	l'avais eu	tu avais eu	il avait eu	elle avait eu	nous avions eu	vous aviez eu	ils avaient eu	elles avaient eu
9	avoir	3	l'aurai eu	tu auras eu	il aura eu	elle aura eu	nous aurons eu	vous aurez eu	ils auront eu	elles auront eu
16	être	3	je suis	tu es	il est	elle est	nous sommes	vous êtes	ils sont	elles sont
11	être	3	je serai	tu seras	il sera	elle sera	nous serons	vous sarez	ils seront	elles seront
12	être	3	je étais	tu étais	il était	elle était	nous étions	vous étiez	ils étaient	elles étaient
13	être	3	je ai été	tu as été	il a été	elle a été	nous avons été	vous avez été	ils ont été	elles ont été
14	être	3	suis				soyons	soyez		
16	être	3	je fus	tu fus	il fut	elle fut			ils furent	elles furent
11	être	3	je serais	tu serais	il serait	elle serait	nous serions	vous seriez	ils seraient	elles seraient
17	être	3	que je sois	que tu sois	qu'il soit	qu'elle soit	que nous soyions	que vous soyiez	qu'ils soient	qu'elles soient
18	être	3	l'avais été	tu avais été	il avait été	elle avait été	nous avions été	vous aviez été	ils avaient été	elles avaient été
14	être	3	l'aurai été	tu auras été	il aura été	elle aura été	nous aurons été	vous aurez été	ils auront été	elles auront été
21	chanter	1	je chante	tu chantes	il chante	elle chante	nous chantons	vous chantez	ils chantent	elles chantent
21	chanter	1	je chanterais	tu chanteras	il chantera	elle chantera	nous chanterions	vous chanterez	ils chanteront	elles chanteront
22	chanter	1	je chantais	tu chantais	il chantait	elle chantait	nous chantions	vous chantiez	ils chantient	elles chantient
23	chanter	1	je chanté	tu chanté	il chanté	elle chanté	nous chanté	vous avez chanté	ils ont chanté	elles ont chanté
21	chanter	1	chante				chantons	chantez		
26	chanter	1	je chantai	tu chantas	il chanta	elle chantá			ils chantèrent	elles chantèrent
24	chanter	1	je chanterais	tu chanterás	il chanteráit	elle chanteráit	nous chantériam	vous chanterez	ils chantaríam	elles chantaríent
27	chanter	1	que je chante	que tu chantes	qu'il chante	qu'elle chante	que nous chantons	que vous chantiez	qu'ils chantent	qu'elles chantent
28	chanter	1	l'avais chanté	tu avais chanté	il avait chanté	elle avait chanté	nous avions chanté	vous aviez chanté	ils avaient chanté	elles avaient chanté
22	chanter	1	l'aurai chanté	tu auras chanté	il aura chanté	elle aura chanté	nous aurons chanté	vous aurez chanté	ils auront chanté	elles auront chanté
50	finir	2	je finis	tu finis	il finit	elle finit	nous finissons	vous finissez	ils finissent	elles finissent
51	finir	2	je finirai	tu finiras	il finira	elle finira	nous finirons	vous finirez	ils finiront	elles finiront
52	finir	2	je finissais	tu finissais	il finissait	elle finissait	nous finissions	vous finissiez	ils finissaient	elles finissaient
53	finir	2	je fini	tu as fini	il a fini	elle a fini	nous avons fini	vous avez fini	ils ont fini	elles ont fini
54	finir	2		finis			finissons	finissez		
56	finir	2	je finis	tu finis	il finit	elle finit			ils finirent	elles finirent
51	finir	2	je finirai	tu finiras	il finira	elle finira			ils finiraien	elles finiraien
57	finir	2	que je finisse	que tu finisses	qu'il finisse	qu'elle finisse	que nous finissions	que vous finissiez	qu'ils finissent	qu'elles finissent

Si l'on résume, et pour faire un schéma, techniquement voici comment se déroule la consultation de page web :



1. - L'internaute effectue une demande (via une adresse URL).
2. - Le serveur reçoit la demande et trouve le fichier (page web) intégrant code HTML + PHP, du coup il passe par l'interpréteur pour décoder les lignes de PHP.
3. - L'interpréteur commence à décoder les instructions PHP et voit des requêtes SQL, il communique donc avec la base de données.
4. - La base de données (SGBD) retourne les informations demandées.
5. - L'interpréteur intègre les données envoyées par la base et termine de décoder les instructions PHP et le fichier (page web) est renvoyé au serveur au format 100% HTML (puisque l'interpréteur a transformé toutes les instructions PHP en résultat HTML).
6. - Le serveur renvoie la page HTML au navigateur, sur l'ordinateur de l'internaute.

Comprenez-vous pourquoi vous ne verrez jamais 1 ligne de code PHP dans le code-source d'une page web ?

En effet, même si la page web contient du code PHP, ce code est décodé par l'interpréteur de manière à ne renvoyer que du HTML au navigateur de l'internaute.

De toute façon, le navigateur ne pourrait pas comprendre des lignes PHP, ce n'est pas son rôle, il doit simplement afficher la page web et pour ça il a uniquement besoin de garder le code HTML et CSS.

Si du code PHP est tout de même passé au navigateur, il le considérera comme du texte et ne l'exécutera pas, du coup votre programmation ne fonctionnera pas.

C'est aussi la raison pour laquelle vous ne pourrez jamais créer un fichier PHP sur votre bureau, ou le lancer directement et manuellement dans le navigateur, sans aucun serveur web ou interpréteur.

En effet, il est impératif que votre fichier contenant le code PHP respecte tout ce cycle (schéma au dessus : interprétation et exécution, échange avec la base de données éventuellement, etc.).

Afin de faire fonctionner notre code PHP nous utilisons le serveur web WAMP (qui inclut tout ce dont nous avons besoin) :

Windows - système d'exploitation.

Apache - serveur web (HTTP).

Mysql - SGBD (Système de Gestion de Base de Données).

PHP - Interpréteur PHP.



#### ➤ Bon à savoir

- PHP est un langage exécuté côté serveur, ce ne sera donc pas une exécution par l'ordinateur ou le navigateur de l'internaute.
- PHP est un langage interprété, ses instructions sont traitées séquentiellement par le serveur (pas de compilation)
- PHP est open source, tout le monde peut l'utiliser et même vendre une de ces créations à l'aide du langage PHP.

Les sites dynamiques utilisent plusieurs langages : html, css, php, sql, javascript.

07

## Les langages de programmation

Y'a t'il d'autres langages de programmation différents de Php pour construire un site web dynamique ?

Oui, d'autres langages exécutés côté serveur existent pour créer un site web dynamique : ASP (Active Server Page, développé par Microsoft), JSP (Java Server Pages, développé par Oracle), CGI (Common Gateway Interface) , etc.

Nous nous concentrerons d'abord sur Php dans le cadre de ce cours.

08

## Qu'est-ce qu'un serveur ?

1 serveur est généralement un ordinateur qui héberge un site web, mis à disposition par un hébergeur (hébergeur = une société de service informatique).

Cet ordinateur reste toujours allumé, c'est ce qui permet une consultation 24h/24 et 7j/7.

09

## Php procédural ou Php Orienté Objet ?

Deux techniques de programmation sont souvent utilisées :

- L'Approche Procédurale (le code est écrit séquentiellement).
- L'Approche Orientée Objet (le code est encapsulé dans des méthodes de classes et fonctionne via l'interaction d'objets).

Avant de s'intéresser aux avantages et inconvénients de ces deux techniques, sachez que certaines notions de bases sont forcément apprises en approche procédurale.

L'approche orientée objet est souvent privilégiée par les entreprises car cela permet d'encourager le travail collaboratif et simplifie grandement les mises à jour. (surtout pour les projets de grande envergure, développés sur plusieurs mois et par plusieurs personnes).

Techniquement, nous ne pouvons pas aller plus loin en terme de finalité et de rendu fonctionnel avec l'orienté objet par rapport au procédural. C'est avant tout une méthodologie de travail (et non pas une évolution pour créer davantage).

10

## Développement : from scratch ? cms ? framework ?

A ce jour, il y a 3 grands moyens de créer un site web :

- À la main (from scratch, en partant de 0),
- avec l'appui d'un CMS,
- avec l'aide d'un FRAMEWORK,

Lorsqu'on utilise un CMS ou un FRAMEWORK, le code peut être procédural ou orienté objet (et parfois le code reprend un peu des deux techniques).

Avant de s'intéresser à chacun de ces moyens, il est indispensable de bien connaître et maîtriser le PHP (dans sa forme basique).

Notre plan de cours reprend les chapitres indispensables de la programmation et du PHP.

Toutefois, si vous êtes attirés par des boutiques ecommerce, des lightbox qui pop, des diaporamas qui slides, des sites dynamiques avec backOffice, des réseaux sociaux avec connexions membres et espace de profil accompagné d'avatar, etc. et le tout en responsive web design... ce ne sera pas pour tout de suite ! il y a quelques étapes à respecter et pour cela il va falloir s'amuser de patience.



L'apprentissage prend du temps mais ce n'est pas une mauvaise chose car si tous ces sujets s'apprenaient en 10 minutes, ce serait payé au SMIC et ce ne serait pas un métier mais plutôt "un boulot". A l'issue de votre apprentissage vous aurez un vrai savoir-faire que vous pourrez valoriser financièrement sur le marché de l'emploi.

Avant de créer un site web, quel qu'il soit (du plus grand portail au plus simple site du type blog), vous devrez impérativement connaître les sujets suivants :

1. Intro : découverte et explications autour de PHP
2. Instructions d'affichage
3. Variable (et Constante) : Type / Déclaration / Affectation
4. Syntaxe et concaténation
5. Condition et Opérateurs
6. Fonction prédéfinie
7. Fonction utilisateur
8. Boucle
9. Inclusions de fichiers
10. Array
11. Classes et objets
12. Les Superglobale
13. Lien GET et Formulaire POST
14. Cookie
15. Session
16. Requête SQL (Mysqli, Pdo)
17. Réalisation d'un espace de dialogue (Cas Concret)
18. Approche Sécurité : Espace membre
19. Réalisation d'un site web (Cas concret : CRUD - tendance ecommerce)
20. Evaluation

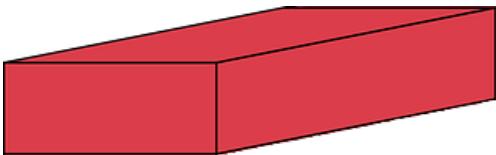
*Ce sera notre plan de cours et notre défi : apprendre le php en 20 chapitres !*

Avant de démarrer, je préfère prévenir surtout tout ceux qui ont un fort besoin de visibilité et de compréhension immédiate du contexte. La question « A quoi ça sert ? » est à bannir. Il faut d'abord comprendre « Comment ça fonctionne ? ». le contexte viendra plus tard.

On essaiera de donner un maximum de contexte mais, de votre côté, il est impératif de privilégier la question « Comment ça fonctionne ? » plutôt que « A quoi ça sert ? », dites vous par défaut que tout sert à quelque chose, sinon cela n'existerait pas et nous n'en parlerions pas !

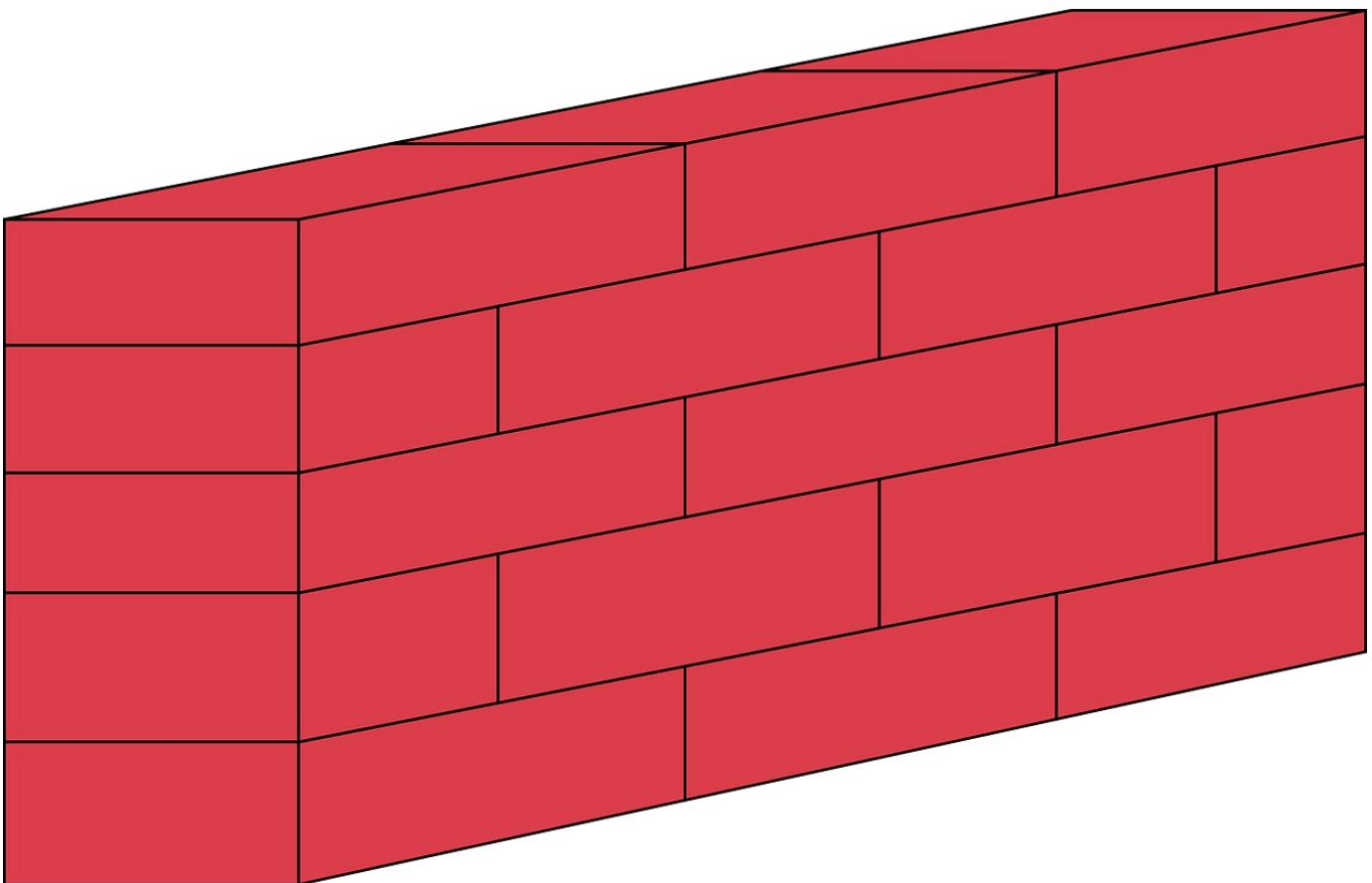
Nous pouvons expliquer à quoi sert ces différents chapitres les uns après les autres mais cela ne sera pas représentatif car cela dépend du contexte dans lequel ils sont utilisés.

C'est comme si je vous demandais : à quoi sert une brique ?



A première vue, comme ça... pas à grand chose !

Maintenant ça sert à quoi plusieurs briques ?



Plusieurs briques permettent de construire un mur. C'est concret !

Pour PHP, c'est pareil, 1 sujet peut paraître abstrait mais l'ensemble des sujets que nous allons aborder vous permettront de construire des sites web dynamiques.

Pour cela ne soyez pas trop impatient (et surtout n'essayez pas d'aller plus vite que votre compréhension, au risque de voir l'effet inverse se produire : retourner en arrière et être ralenti).

Etudions maintenant les mécanismes de PHP !

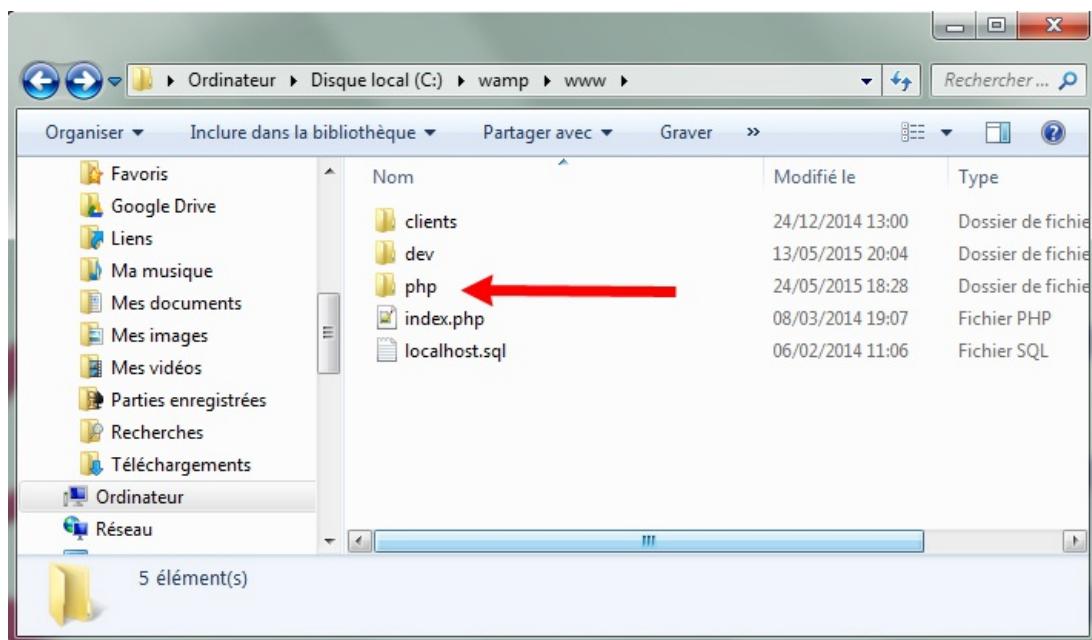
Avant de construire un site web et écrire nos premières lignes en PHP, vous allez devoir préparer le terrain avec au minimum :

- Un serveur web : WAMP (d'autres serveurs web existent : easyPHP, xampp, wamp est aussi décliné sous mac avec mamp et sous linux avec lamp)
- Un éditeur de code : NOTEPAD (d'autres éditeurs type SublimeText existent, ou éventuellement un IDE (plus puissant mais pas nécessaire pour débuter) : Eclipse, PhpStorm, Zend, etc.)
- Un navigateur (type Firefox ou Google Chrome) permettant la consultation de page web.

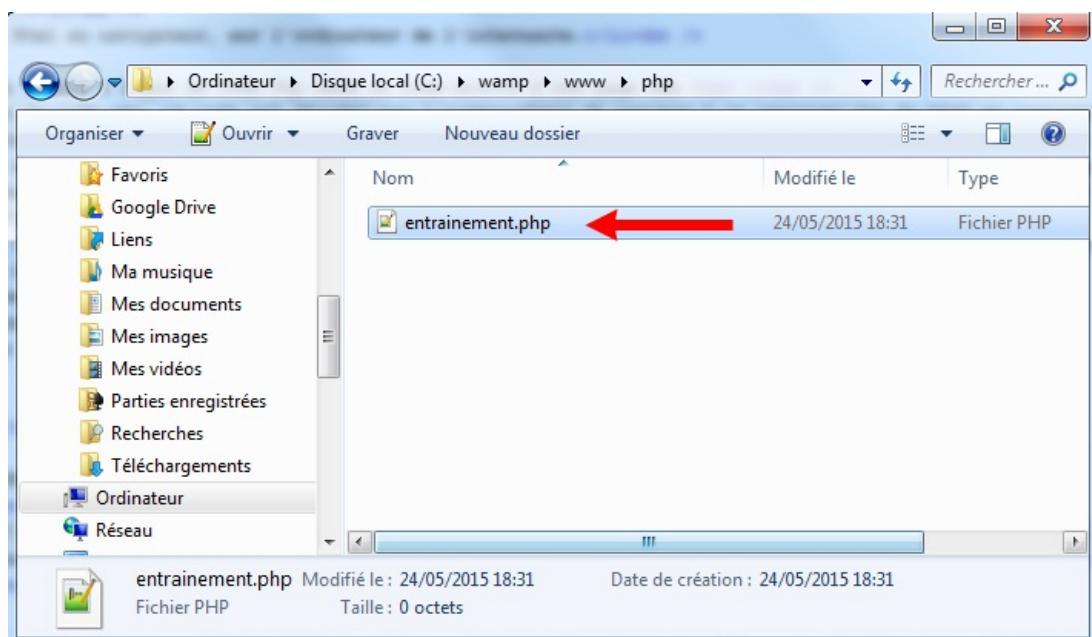
Avant de créer votre premier fichier, vous devez impérativement avoir installer [WAMP](#) et pouvoir accéder à <http://localhost/>.

Pour créer notre premier fichier PHP, nous allons nous rejoindre dans le dossier nommé /www/

Créons un premier dossier nommé /php/ :



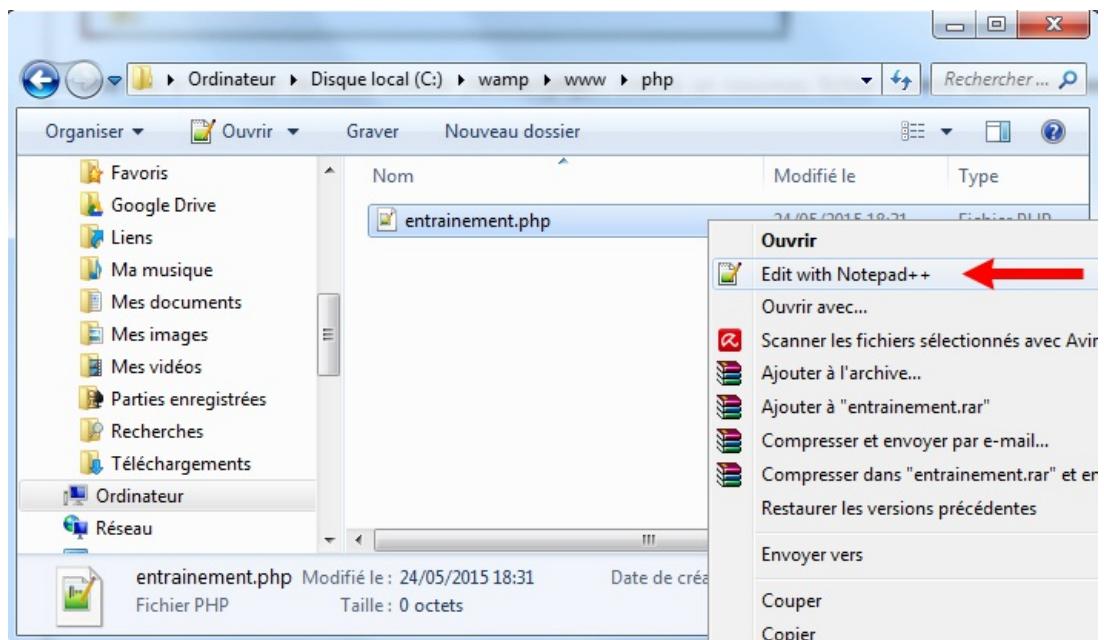
Une fois à l'intérieur de votre dossier /www/php/, créons un nouveau fichier nommé entrainement.php.



Le dossier /www/ se nomme **htdocs** sous MAMP (pour mac).

Pour inscrire et introduire du code dans votre fichier entrainement.php, vous pouvez effectuer un clic droit sur le fichier avec votre souris et éditer le code

avec l'aide de notepad :



Il est important que ce fichier possède l'extension « .php ».

## Instruction d'affichage

14

## Instruction d'affichage

Démarrons avec notre fichier entraînement.php

```
entraînement.php
?>
<?php
echo 'Bonjour';
print 'Nous sommes vendredi et il fait beau !';
```

Ce code PHP permet d'afficher du texte.

`echo` et `print` peuvent être traduits par « affiche moi »s.

Chaque instruction se termine par un point-virgule « ; ».

Le texte peut être mis entre apostrophes (quotes) ou guillemets, nous reviendrons sur ces deux possibilités dans le chapitre 5 dédié à ce sujet.

15

## Déférence entre echo & print

Pour une première initiation vous pouvez retenir qu'il n'y a pas de différence entre echo et print, et qu'il vaut mieux utiliser le echo.

Pour les plus connaisseurs et ceux qui aiment bien creuser un sujet jusqu'au bout, sachez que le echo ressort comme étant légèrement plus rapide (centième de seconde) car il ne fait pas de return. print peut se retrouver à droite d'une expression pour être évalué. (Fermez les yeux sur cette partie si cette phrase n'est pas claire, elle ne vous empêchera pas de suivre le déroulé du cours).

D'autres instructions permettent d'effectuer des affichages pour se débugger :

```
print_r  
var_dump
```

Durant la création d'un site web, ces instructions seront très utiles pour se répéter.

Nous reviendrons sur leur utilisation un peu plus tard afin de faire un tour complet sur leur utilisation.

Pour introduire des commentaires dans une page web, nous pouvons le faire de plusieurs manières différentes :

```
entrainement.php
```

```
<?php  
echo 'Bonjour';  
echo 'Nous sommes vendredi et il fait beau !';  
// commentaire sur 1 ligne  
/* commentaire  
sur plusieurs  
lignes */  
# commentaire sur 1 ligne
```

**Ctrl+Q** vous permettra d'utiliser le raccourci clavier afin d'appliquer des commentaires PHP sous notepad.

Pour introduire du code Html dans un fichier Php (et non pas du php dans du html !) nous avons deux solutions :

#### 1- Html dans du PHP via l'instruction echo

```
entrainement.php
```

```
<?php  
echo '<h1>Bonjour</h1>';  
echo 'Nous sommes <strong>vendredi</strong> et il fait beau !<br>';  
echo '<div class="mazone">  
    Je crée une zone html dans du php  
</div>';
```

Vous pouvez remarquer la présence des balises **<h1>**, **<br>** et **<strong>** directement dans le code PHP.

Le code Html peut donc facilement être introduit à l'intérieur du Php.

Voici un autre moyen de mélanger les deux langages :

## 2- Entrée et sortie répétitives de PHP

### entrainement.php

```
<h1><?php echo 'Bonjour'; ?></h1>
<?php echo 'Nous sommes'; ?> <strong>vendredi</strong> <?php echo ' et il fait beau !'; ?> <br>
<div class="mazone"><?php echo 'Je crée une zone html dans du php'; ?> </div>
```



#### ➤ Attention

Ce code est moins propre (ce qui veut dire "moins professionnel" dans le jargon du développement) car il fait sortir et rentrer dans le code php à plusieurs reprises.

D'autre part il n'est pas aisément à mettre en œuvre sur plusieurs lignes d'affilée.

Ce code n'est donc pas montré dans le but de vous le faire retenir (car il ne s'agit pas d'une bonne pratique) mais plutôt vous le faire découvrir afin que vous ne soyez pas surpris si vous le croisez dans le code d'un fichier écrit par un débutant, ou certains fichiers templates (notamment avec l'utilisation de CMS).

Il peut arriver que ce type de pratique (entrée et sortie répétitive de PHP) ait lieu dans le cadre de l'écriture de fichier template mélangeant html et php (surtout dans l'univers des cms) car les développeurs pensent que c'est plus facile pour des intégrateurs (qui ne connaissent que HTML) de comprendre une majorité de code HTML avec simplement quelques morceaux de PHP « posés » à l'intérieur.

Par défaut, même si l'extension du fichier est ".php", nous sommes par défaut en langage HTML avant d'ouvrir les balises php <?php ... ?>.



#### ➤ Bon à savoir

Nous pouvons écrire du HTML dans un fichier ayant l'extension PHP, l'inverse n'est pas possible !

19

## Exécuter son code PHP pour tester

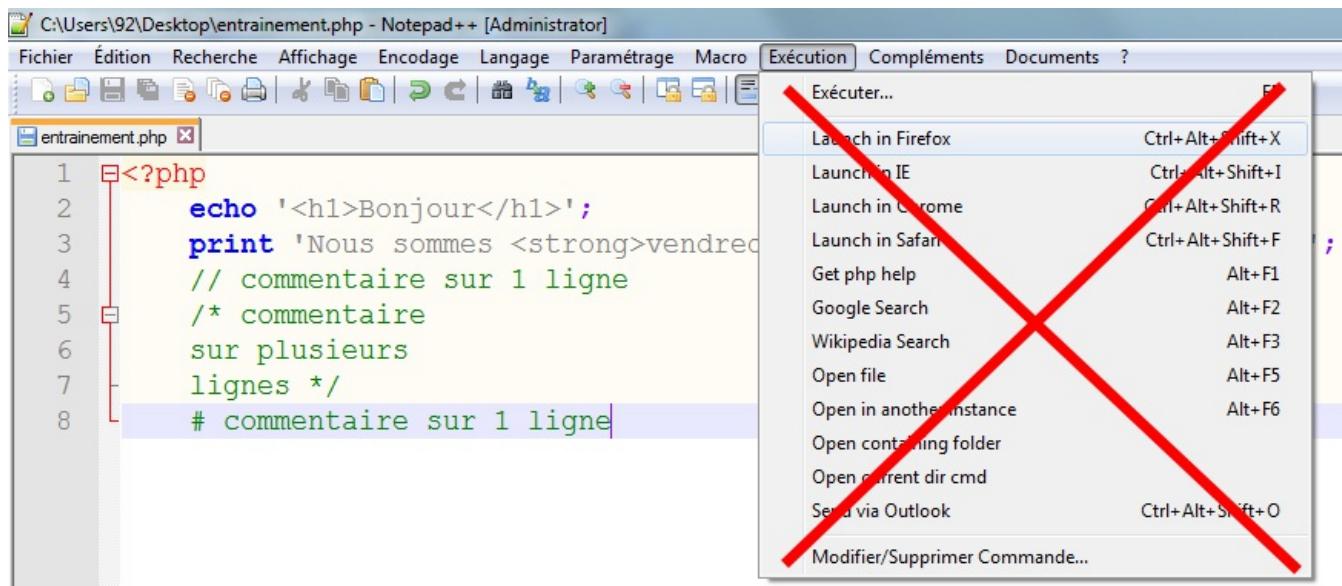
Voici le code que nous gardons pour les tests :

### entrainement.php

```
<?php
echo '<h1>Bonjour</h1>';
echo 'Nous sommes <strong>vendredi</strong> et il fait beau !<br>';
// commentaire sur 1 ligne
/* commentaire
sur plusieurs
lignes */
# commentaire sur 1 ligne
```

Pour exécuter votre code php, il va falloir passer par l'adresse web du serveur.

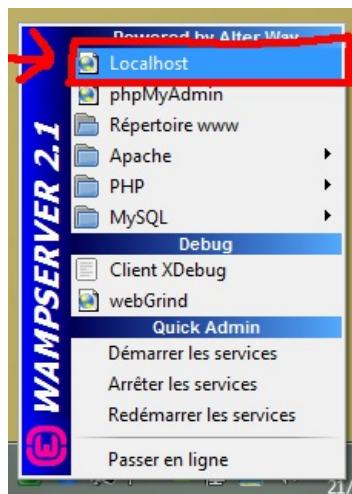
Si vous aviez l'habitude de lancer votre fichier avec notepad (comme pour le html), il va falloir oublier :



Si vous ne comprenez pas pourquoi nous ne pouvons plus lancer l'exécution du fichier à partir de notepad, c'est parce que nous ne pouvons pas envoyer du PHP au navigateur pour qu'il l'évalue, ce n'est pas son travail (le navigateur doit juste présenter la page et pour cela il a besoin de Html et Css).

Si ce n'est toujours pas clair, je vous invite à relire autant de fois que nécessaire [ce schéma explicatif](#).

L'adresse du serveur web qui vous permettra d'exécuter votre code PHP est accessible à partir du menu de wamp :



The screenshot shows the WampServer configuration page. At the top, it displays the WampServer logo and version information: Version 2.1 English Version. Below this, the Configuration Serveur section provides details about Apache (Version 2.2.17), PHP (Version 5.3.5), and MySQL (Version 5.5.8). A list of loaded extensions is shown in four columns:

Extensions Chargées:	Core	bcmath	calendar	com_dotnet	ctype
	date	ereg	filter	ftp	hash
	iconv	json	mcrypt	SPL	odbc
	pcre	Reflection	session	standard	mysqlnd
	tokenizer	zip	zlib	libxml	dom
	PDO	Phar	SimpleXML	wddx	xml
	xmlreader	xmlwriter	apache2handler	gd	mbstring
	mysql	mysqli	pdo_mysql	pdo_sqlite	mhash
	xdebug				

A red arrow points to the 'php' project folder under the 'Vos Projets' section.

Et ensuite vous devrez cliquer sur le fichier nommé `entrainement.php`

The screenshot shows a browser window with the URL `localhost/php/entrainement.php`. The page content is as follows:

**Bonjour**

Nous sommes **vendredi** et il fait beau !

résultat

Pensez à regarder le code-source (**CTRL+U** ou clic droit > code source de la page), vous verrez qu'on n'y voit uniquement le code html et non pas le php qui lui a été décodé, conformément à notre [schéma de départ](#).

Si vous voyez une erreur apparaitre, vérifier bien votre code et aussi si vous avez bien fermé vos quotes (apostrophe) et mis vos points-virgules en fin de ligne.

## Les variables et constantes

## Définition d'une variable

Une variable est un espace nommé permettant de conserver une valeur.

Une variable porte un nom (que l'on définit nous même) et elle permet de conserver une valeur pour la transporter dans notre script.

Il arrive régulièrement que nous ayons connaissance d'une information ligne 27 (par exemple) et que nous voulions la ré-utiliser ligne 232 (par exemple) .

Pour retenir une information en mémoire : nous utilisons une variable !



### ➤ A quoi ça sert une variable ?

une variable permet de garder une informations en mémoire (durant l'exécution du script), nous pouvons la consulter ou l'utiliser plus tard dans l'exécution du script.

Pourquoi utiliser des variables ?

*Pour 1000 choses différentes, si nous évoquions toutes les situations cela risquerait de vous semer car vous n'avez pas encore le niveau pour comprendre les réponses qui font appel à d'autres aspects techniques.*

*Alors tant que vous êtes encore jeune dév apprenti, n'oubliez pas de vous concentrer sur la question « comment ça marche? » plutôt que vous distraire sur la question « à quoi ça sert? ».*

*J'insiste mais c'est important, avec mon expérience de formateur j'ai vu trop de gens se braquer et abandonner sous prétexte qu'ils ne voyaient pas où cet apprentissage allait les emmener.*

Voici un exemple pratique avec un prenom :

```
1 <?php  
2 echo "<p>Bonjour Fred</p>";  
3  
4 echo "<p>Comment va-tu Fred ?</p>";  
5  
6 echo "<p>Voici tes informations de profil Fred ...</p>";
```

Vous voyez la répétition ? si nous avons besoin d'appeler le prénom Fred à plusieurs endroits différents, cela risque d'être embêtant.

Pour cela, nous pourrions prévoir une variable qui gardera en mémoire le prénom de l'internaute.

Exemple :

entraînement.php

```
1 <?php  
2 $prenom = "Fred";  
3  
4 echo "<p>Bonjour $prenom</p>";  
5  
6 echo "<p>Comment va-tu $prenom ?</p>";  
7  
8 echo "<p>Voici tes informations de profil $prenom ...</p>";
```



### ➤ Informations

pour que cet exemple fonctionne parfaitement, il faudra prévoir l'utilisation de guillemets plutôt que des quotes (apostrophes)

Avec cet exemple, vous verrez que nous déclarons (ligne 2) la variable prenom contenant la valeur "Fred", cela permet de garder l'information en mémoire pour ensuite nous la ré-utiliser facilement à plusieurs reprises sur les lignes 4, 6 et 8. (nous transportons de lignes en lignes le prénom grâce à notre variable !)

A vous de faire le test dans votre navigateur !

résultat

Bonjour Fred

Comment va-tu Fred ?

Voici tes informations de profil Fred ...

#### Explications et décomposition du code

Une variable est toujours déclarée avec le symbole \$ (dollar),

Ensuite nous précisons le nom de la variable, ici nous avons choisi prenom, ce qui donne \$prenom.

Le signe égal " = " est dans ce contexte un opérateur d'affectation.

"Fred" est la valeur entrante dans la variable pour être gardée en mémoire tout le temps d'exécution du script.

L'instruction echo permet quant à elle uniquement de faire des affichages.

Voici un peu de vocabulaire (utile si vous vous retrouvez à travailler à plusieurs, ou en entreprise) :

On dit que l'on déclare la variable \$prenom et qu'on lui affecte la valeur Fred.

Autre avantage, si nous devons changer le prenom, nous le ferons qu'à un seul endroit et il sera répercuté partout :

```
entraînement.php
?
1 <?php
2 $prenom = "Nicolas";
3
4 echo "<p>Bonjour $prenom</p>";
5
6 echo "<p>Comment va-tu $prenom ?</p>";
7
8 echo "<p>Voici tes informations de profil $prenom ...</p>";
```

Imaginez qu'il y ait 1 000 lignes qui fassent appel à la variable \$prenom, nous ferons qu'une seule modification ligne n°2 et elle sera répercutée partout par la suite.

résultat

Bonjour Nicolas

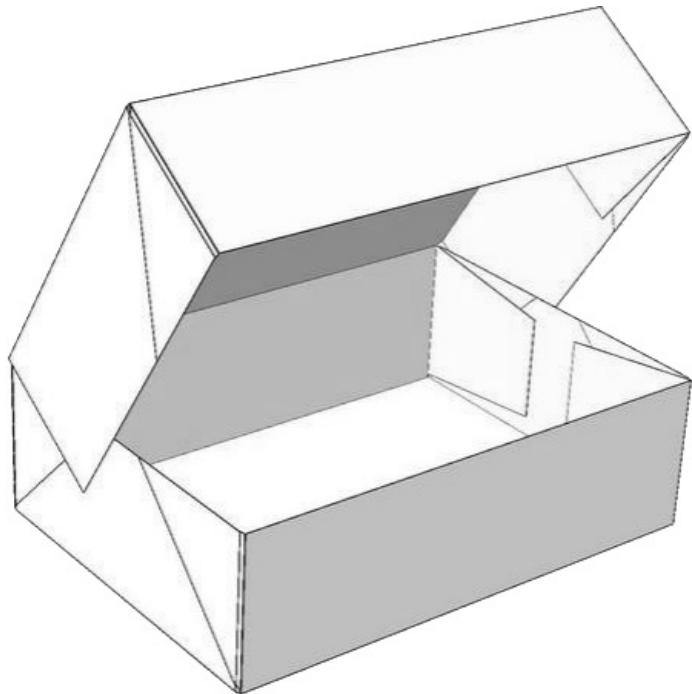
Comment va-tu Nicolas ?

Voici tes informations de profil Nicolas ...

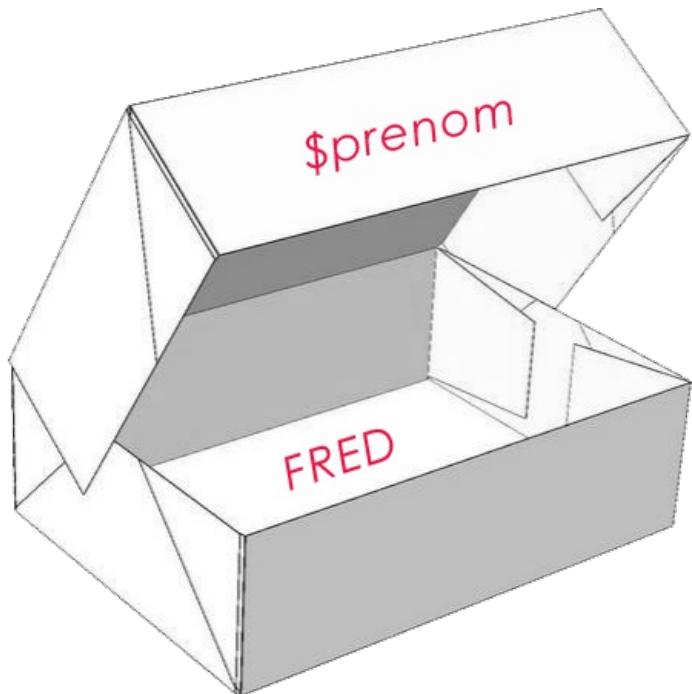
Pour mieux se représenter ce qu'est une variable, voici une métaphore :

Une variable est comme une petite boite dans laquelle nous rangeons quelque chose, plus tard nous rouvrons notre boite et retrouvons ce que nous avions introduit.

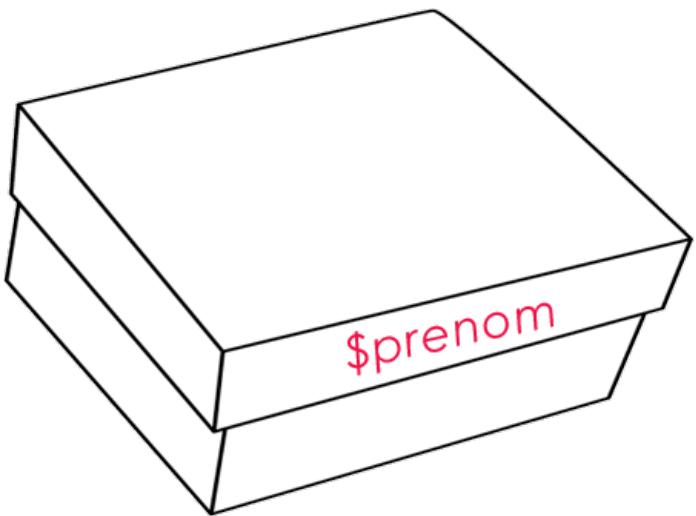
A l'origine la variable n'existe pas :



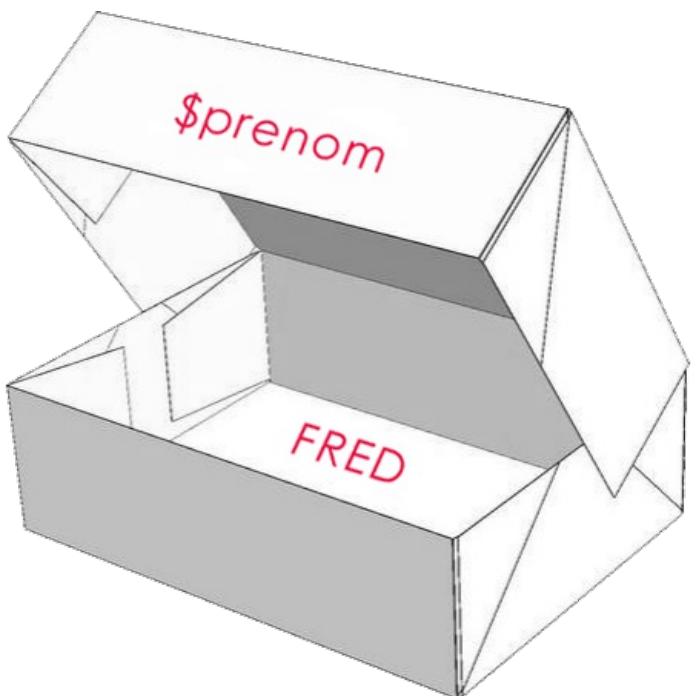
Nous déclarons la variable \$prenom avec l'information "Fred" à l'intérieur :



Cette information est conservée à l'intérieur de la variable :



Plus tard, nous avons besoin de connaître l'information contenue dans notre variable (nous rouvrons notre boîte) :



Comme son nom l'indique, une variable ça peut varier !

Le contenu d'une variable n'est pas "figé", il peut être modifié à tout moment, voici un exemple :

```
entraînement.php
```

```
1 | <?php
2 | $prenom = "Fred";
3 |
4 | echo "<p>Bonjour $prenom</p>";
5 |
6 | echo "<p>Comment va-tu $prenom ?</p>";
7 |
8 | $prenom = "Marie";
9 |
10| echo "<p>Voici tes informations de profil $prenom ...</p>";
```

N'oubliez pas d'enregistrer votre fichier et de faire des tests dans le navigateur !

résultat

Bonjour Fred

Comment va-tu Fred ?

Voici tes informations de profil Marie ...

Cela n'a pas de sens dans le cas présent de modifier subitement le prénom d'une personne mais ce code vous démontre qu'une variable est interchangeable et ce autant de fois qu'on le souhaite (même en cours de route) ! Ceci sera très pratique pour nos futures programmation.

21

## Nom de variable

Le nom d'une variable doit être explicite, le nom peut être défini en fonction de l'information qu'elle contient.

*Si ma variable est destinée à contenir "rouge", je pourrais la nommer \$couleur.*

**Le nom d'une variable doit respecter certaines règles**

Par exemple, elle ne doit pas commencer par 1 chiffre mais peut en contenir :

*Exemple : \$voiture2 mais pas \$2voiture.*

Nous pouvons mettre toutes les lettres de l'alphabet (A à Z) dans nos variables (en majuscule ou minuscule).

Nous ne pouvons pas mettre de caractères spéciaux, pas d'espace, et bien entendu on évite les accents sur des noms de variable : \$prenom et non pas \$préénom.

**Respecter une convention de nommage**

Il faut absolument éviter (pour vous et pour les autres développeurs qui rejoindront ou récupéreront votre projet) de nommer des variables avec des noms de convention différentes.

**Exemple :** \$mon-prenom, \$mon\_prenom, \$monPrenom, \$MonPréNom, \$pren, \$info, \$\_monprenom.

Plusieurs conventions de nommage existent, l'essentiel c'est d'en choisir une et de la garder du début à la fin de votre programmation, je vous suggère la syntaxe suivante : \$monPrenom.

Cela consiste à inscrire une majuscule lorsqu'un espace doit avoir lieu. (la première lettre est une minuscule).

N'oubliez pas d'enregistrer votre fichier et de faire des tests dans le navigateur !

Les variables sont sensibles à la casse, par conséquent, si vous déclarez une variable contenant une majuscule, il ne faudra pas se tromper au moment de lui demander de s'afficher.

**Exemple :**

```
1 <?php
2 $monPrenom = "Fred";
3
4 echo $monPrenom; // affiche : Fred
5
6 echo $monprenom; // affiche une erreur "undefined variable..." soit variable indéfinie (inconnue)
```

En effet, sur les lignes 2 et 4 les variables sont nommées de la même manière, il n'y a donc aucun problème.

Sur la ligne 6, le "p" de prenom ainsi que toutes les autres lettres sont en minuscule, ce qui ne correspond pas à la déclaration de notre variable ligne 2.

22

## Les types de variables

Reprendons notre fichier entrainement.php pour les tests :

```
entraînement.php
```

```
1 | <?php
2 | $variable1 = 127;
3 | echo gettype($variable1) . "<br>";
```

Sur la ligne n°2, nous faisons une affectation de la valeur 127 dans la variable nommée "\$variable1".

Sur la ligne n°3, nous utilisons gettype() qui est une fonction prédefinie (= prévue par le langage Php) nous permettant de voir le type d'une variable. Dans notre cas, il s'agit d'un integer, ce qui correspond à un entier (un nombre).

Le point nous permet seulement de faire suivre des informations (avec l'utilisation du <br> pour passer à la ligne). Il s'agit de la concaténation dont nous parlerons en détail au chapitre 5.

### entraînement.php

```
1 | <?php
2 | $variable2 = 1.5;
3 | echo gettype($variable2) . "<br>";
```

Cette variable est de type double correspondant à un nombre à virgule (déclaré avec un point).

### entraînement.php

```
1 | <?php
2 | $variable3 = "une chaîne";
3 | echo gettype($variable3) . "<br>";
```

Là non plus, nous ne regardons pas le contenu d'une variable mais son type : string

### entraînement.php

```
1 | <?php
2 | $variable4 = '127';
3 | echo gettype($variable4) . "<br>";
```

C'est toujours le type string, même s'il s'agit d'un nombre, l'utilisation des quotes fait que le php le considère comme une chaîne de caractères

### entraînement.php

```
1 | <?php
2 | $variable5 = TRUE;
3 | echo gettype($variable5) . "<br>";
```

Type Boolean : TRUE (VRAI, assimilé à 1) ou FALSE (FAUX, assimilé à 0) uniquement.

### entraînement.php

```
1 | <?php
2 | $variable6 = FALSE;
3 | echo gettype($variable6) . "<br>"; // boolean
```

Type Boolean : TRUE (VRAI, assimilé à 1) ou FALSE (FAUX, assimilé à 0) uniquement.

résultat

integer

double

string

string

boolean

boolean

Pour débuter, voici les principaux types de variables :

Type	Description
integer	chiffre ou nombre
string	chaîne de caractères (texte)
boolean	vrai ou faux (TRUE or FALSE)

*// existe d'autres types de variable.*

23

## Assignation par référence

Voici une notion importante à comprendre, l'assignation par référence :

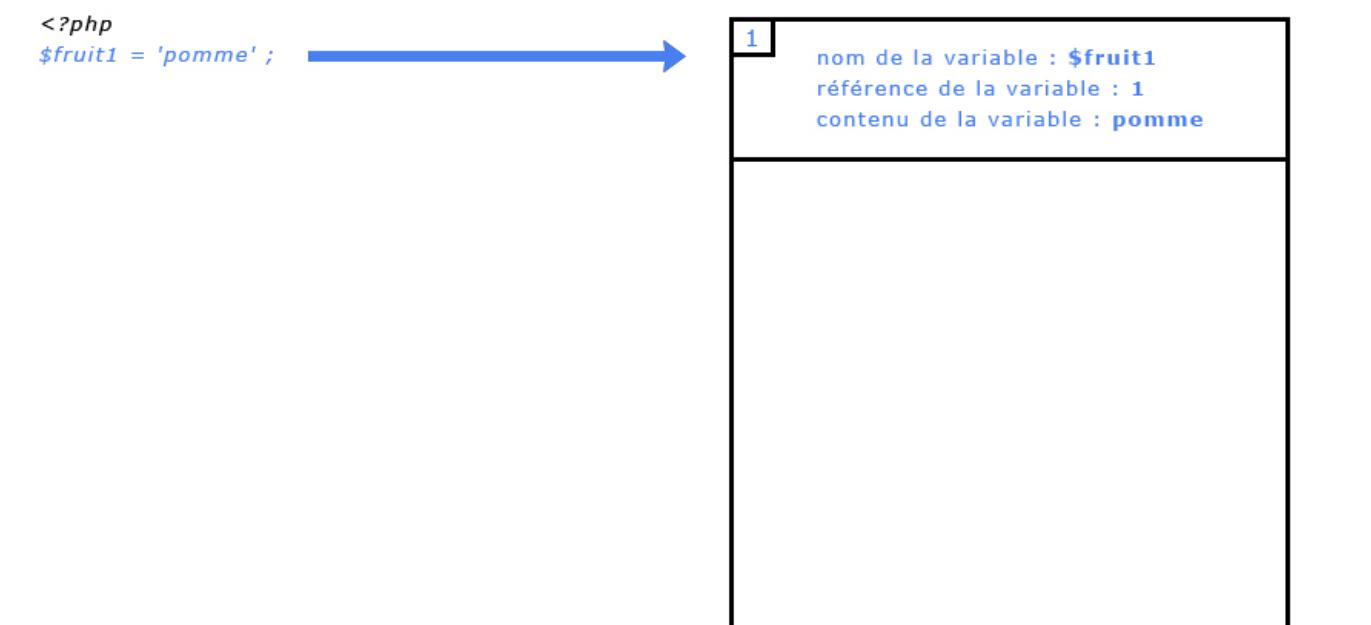
```
entraînement.php
```

```
1 | <?php
2 | $fruit1 = 'pomme'; // affecte la valeur 'pomme' à $fruit1
3 | $fruit2 = 'banane'; // affecte la valeur 'banane' à $fruit2
4 | echo "fruit1 : $fruit1 <br>"; // affiche : pomme
5 | echo "fruit2 : $fruit2 <br>"; // affiche : banane
6 | $fruit2 = &$fruit1; // passage de référence (assigne la référence de $fruit1 à $fruit2)
7 | echo "fruit1 : $fruit1 <br>"; // affiche : pomme (référence 1)
8 | echo "fruit2 : $fruit2 <br>"; // affiche : pomme (référence 1)
9 | $fruit2 = 'fraise'; // modifie la valeur de $fruit2 (et de $fruit1 par répercussion, même espace mémoire)
10 | echo "fruit1 : $fruit1 <br>"; // affiche : fraise
11 | echo "fruit2 : $fruit2 <br>"; // affiche : fraise
12 | // Si nous changeons $fruit2, $fruit1 change en conséquence (et vice versa), leur destin sont liés puisqu'il représente la même référence
   | qui pointe vers 1 seul et même espace mémoire
```

Ligne 1 : Ouverture de php (pas difficile ça ! :p)

Ligne 2 : Nous affectons la valeur 'pomme' à la variable \$fruit1.

Voici ce qu'il se passe réellement :



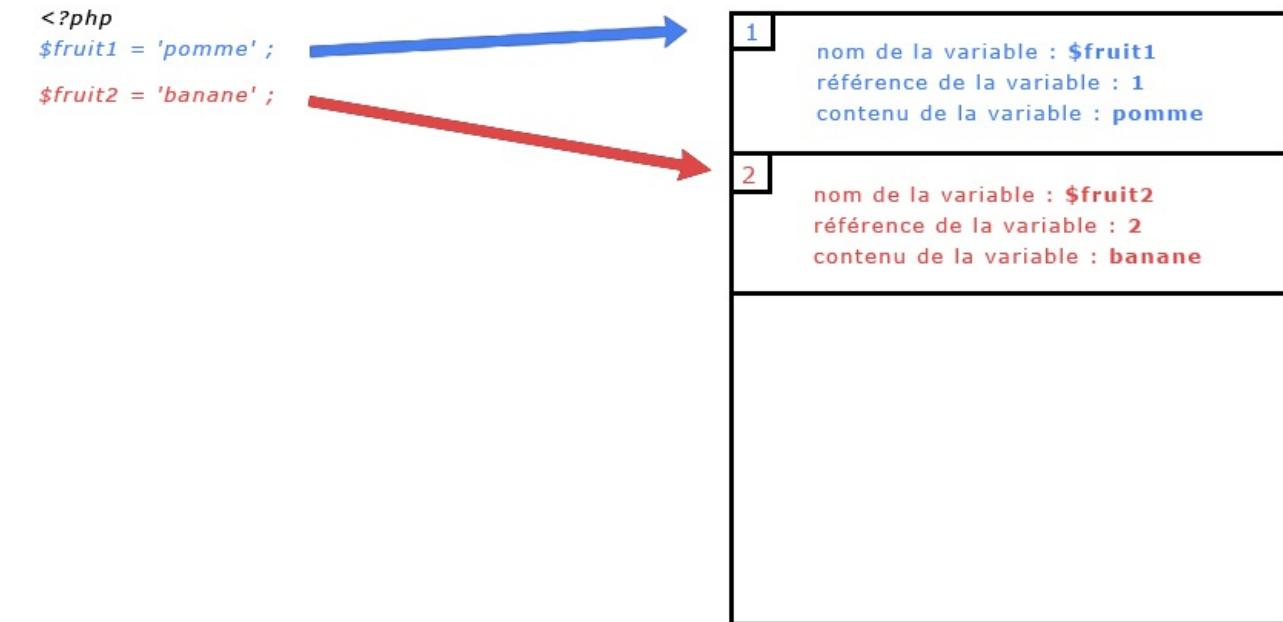
Par habitude, on dit que la variable \$fruit1 contient la valeur "pomme" pour gagner du temps et aller plus vite lors d'un dialogue.

En réalité, la variable \$fruit1 ne contient pas directement de valeur, elle possède la référence 1 qui représente un espace mémoire (avec la même référence, ce qui permet d'assurer la liaison). C'est dans l'espace mémoire 1 qu'est contenu la valeur "pomme".

C'est à dire que la variable n'est pas contenue directement dans la variable au sens propre du mot, la variable est un espace mémoire identifiable par une référence (un numéro, ici le 1) qui mène vers la valeur "pomme".

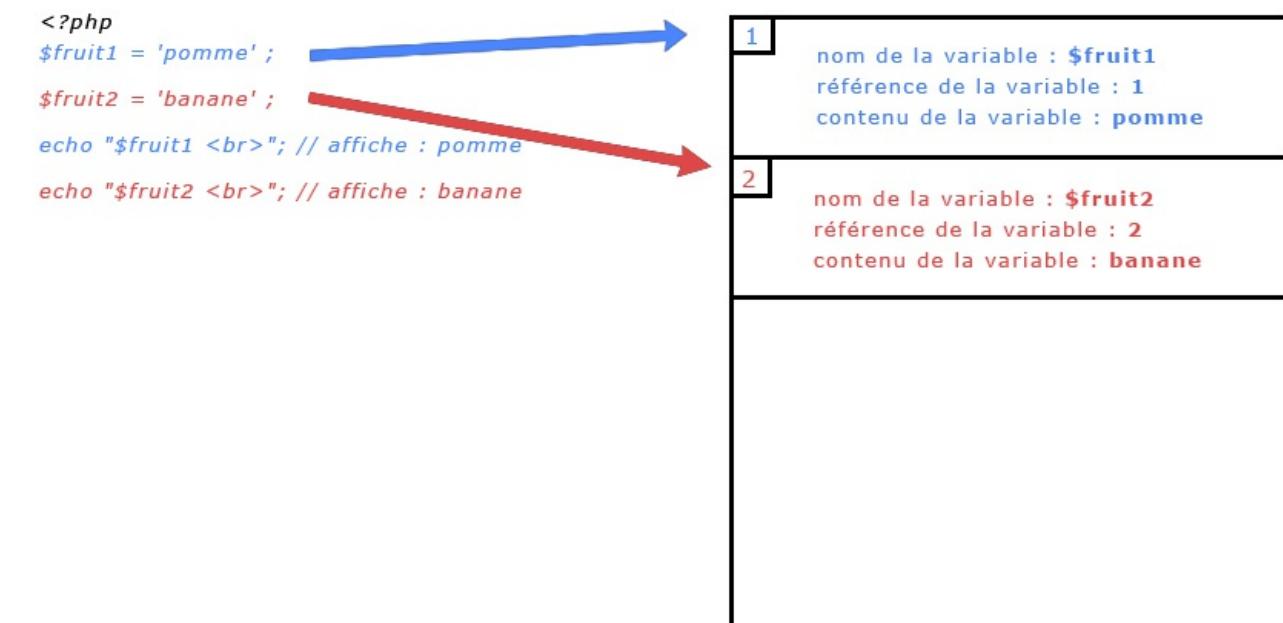
Question : Pourquoi la variable \$fruit1 possède la référence 1 et pas 56 (par exemple) ? Réponse : car c'est la 1ère variable qui a été déclarée.

Ligne 3 : Nous affectons la valeur 'banane' à la variable \$fruit2.



Même principe, la variable \$fruit2 possède la référence 2 qui mène vers la valeur 'banane'.

Ensuite ligne 4 et 5, nous demandons aux variables de nous donner les valeurs qu'elles contiennent dans leur espace mémoire (la liaison se fait grâce à leur référence), voici l'affichage :

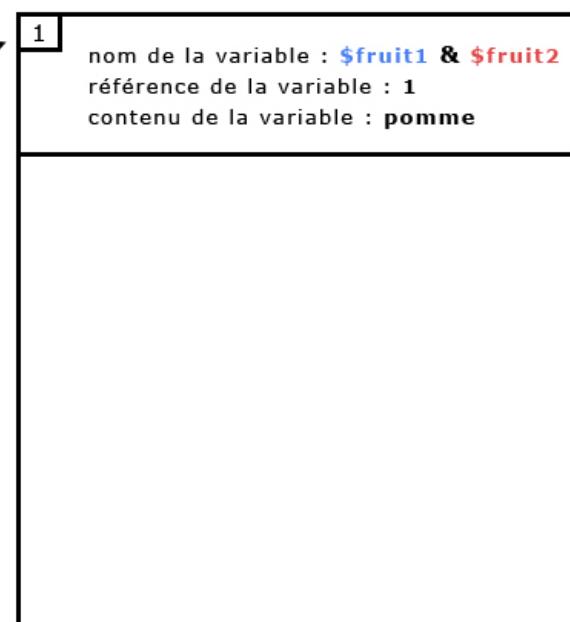


Ligne 6 :

par l'utilisation du symbole &, nous assignons la référence 1 de \$fruit1 à \$fruit2.

Par conséquent, les deux variables \$fruit1 et \$fruit2 pointent vers la même référence (et donc le même espace mémoire), soit 2 variables qui pointent vers la référence 1

```
<?php
$fruit1 = 'pomme';
$fruit2 = 'banane';
echo "$fruit1 <br>"; // affiche : pomme
echo "$fruit2 <br>"; // affiche : banane
$fruit2 = &$fruit1; // passage de référence
                  (assigne la référence
                   de $fruit1 à $fruit2)
echo "$fruit1 <br>"; // affiche : pomme (référence 1)
echo "$fruit2 <br>"; // affiche : pomme (référence 1)
```



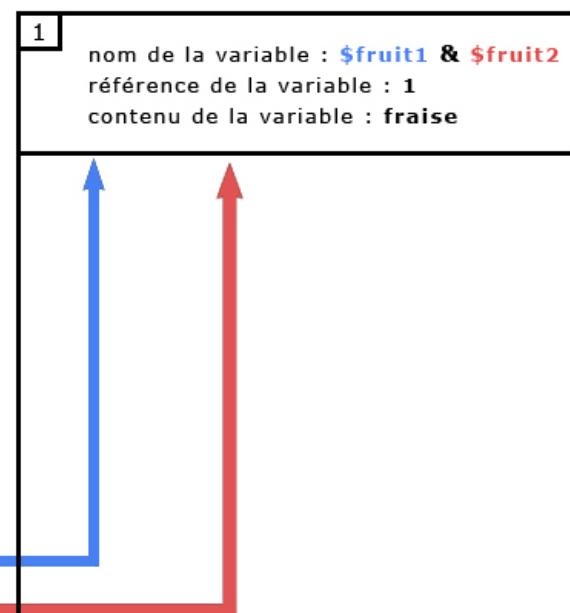
Ligne 7 et 8 : nous demandons des affichages ce qui confirme le schéma d'au dessus.

Ligne 9 : Dorénavant, le destin de \$fruit1 et \$fruit2 sont liés.

Si je change la variable \$fruit2, \$fruit1 changera en conséquence puisqu'il pointe vers le même espace mémoire (même référence)

N'oubliez pas, nous ne changeons pas la variable en elle-même mais le contenu qu'elle représente (dans l'espace mémoire) et la liaison se fait avec la référence.

```
<?php
$fruit1 = 'pomme';
$fruit2 = 'banane';
echo "$fruit1 <br>"; // affiche : pomme
echo "$fruit2 <br>"; // affiche : banane
$fruit2 = &$fruit1; // passage de référence
                  (assigne la référence
                   de $fruit1 à $fruit2)
echo "$fruit1 <br>"; // affiche : pomme (référence 1)
echo "$fruit2 <br>"; // affiche : pomme (référence 1)
$fruit2 = 'fraise'; // modifie la valeur de $fruit2
                   (et de $fruit1 par répercussion,
                    même espace mémoire)
echo "$fruit1 <br>"; // affiche : fraise
echo "$fruit2 <br>"; // affiche : fraise
```



Si nous changeons \$fruit2, \$fruit1 change en conséquence (et vice versa), leur destin sont liés puisqu'ils représentent la même référence qui pointe vers 1 seul et même espace mémoire

C'est donc ce qu'on appelle un passage de référence.

résultat

```
fruit1 : pomme
fruit2 : banane
fruit1 : pomme
fruit2 : pomme
fruit1 : fraise
fruit2 : fraise
```

24

## Assignation de valeur

Different du passage de référence, voici le passage de valeur :

```
entraînement.php
?
1 | <?php
2 | $fruit1 = 'pomme'; // affecte la valeur 'pomme' à $fruit1
3 | $fruit2 = 'banane'; // affecte la valeur 'banane' à $fruit2
4 | echo "fruit1 : $fruit1 <br>"; // affiche : pomme
5 | echo "fruit2 : $fruit2 <br>"; // affiche : banane
6 | $fruit2 = $fruit1; // passage de valeur ($fruit2 contient maintenant "pomme")
7 | echo "fruit1 : $fruit1 <br>"; // affiche : pomme (référence 1)
8 | echo "fruit2 : $fruit2 <br>"; // affiche : pomme (référence 1)
9 | $fruit2 = 'fraise'; // modifie la valeur de $fruit2 uniquement
10 | echo "fruit1 : $fruit1 <br>"; // affiche : pomme
11 | echo "fruit2 : $fruit2 <br>"; // affiche : fraise
12 | // $fruit2 et $fruit1 ne sont pas liés, ils possèdent des références différentes et pointent vers des espaces mémoire différents.
```

Ligne 2 : Nous affectons la valeur 'pomme' à la variable \$fruit1.

Ligne 3 : Nous affectons la valeur 'banane' à la variable \$fruit2.

Ligne 4 : Nous affichons la valeur de la variable \$fruit2, soit : pomme.

Ligne 5 : Nous affichons la valeur de la variable \$fruit1, soit : banane.

Ligne 6 : Nous passons la valeur de la variable \$fruit1 à la variable \$fruit2 (uniquement la valeur, pas de passage de référence sans le symbole &).

Ligne 7 : Nous affichons la valeur de la variable \$fruit1, soit : pomme.

Ligne 8 : Nous affichons la valeur de la variable \$fruit2, soit : pomme.

Ligne 9 : Nous modifions la valeur de la variable \$fruit2, soit : fraise.

Ligne 10 : Nous affichons la valeur de la variable \$fruit1, soit : pomme.

Ligne 11 : Nous affichons la valeur de la variable \$fruit2, soit : fraise.

Comme vous pouvez le constater, \$fruit2 et \$fruit1 ne sont pas liés, ils possèdent des références différentes et pointent vers des espaces mémoire différent.

Voici le schéma final de nos variables :

```

<?php
$fruit1 = 'pomme';
$fruit2 = 'banane';
echo "$fruit1 <br>"; // affiche : pomme
echo "$fruit2 <br>"; // affiche : banane
$fruit2 = $fruit1; // passage de valeur
                ($fruit1 transmet sa valeur
                 à $fruit2)

echo "$fruit1 <br>"; // affiche : pomme (référence 1)
echo "$fruit2 <br>"; // affiche : pomme (référence 1)
$fruit2 = 'fraise'; // modifie la valeur de $fruit2
echo "$fruit1 <br>"; // affiche : pomme
echo "$fruit2 <br>"; // affiche : fraise

```

1	nom de la variable : <b>\$fruit1</b> référence de la variable : <b>1</b> contenu de la variable : <b>pomme</b>
2	nom de la variable : <b>\$fruit2</b> référence de la variable : <b>2</b> contenu de la variable : <b>fraise</b>

## résultat

fruit1 : pomme  
fruit2 : banane  
fruit1 : pomme  
fruit2 : pomme  
fruit1 : pomme  
fruit2 : fraise



### ➤ Ne l'oubliez pas

Une variable est une zone en mémoire du système, qui sert à conserver une valeur.

25

## Les constantes

A quelques détails près une constante c'est comme une variable, son rôle est de conserver une valeur.

Sauf que dans son cas, la valeur ne pourra pas changer durant l'exécution du script, en effet comme son nom l'indique, une constante c'est constant ! Alors qu'une variable ça peut varier.

Au niveau de la convention de nommage, une constante s'écrit toujours en majuscule.

Au niveau de la syntaxe, la déclaration est différente :

	entrainement.php	?
1	<?php	
2	define("CAPITALE","Paris");	
3	echo CAPITALE . " ";	
4		

`define` est une fonction prédéfinie (prévue par le langage PHP) permettant de définir une constante en annonçant d'abord son nom, puis ensuite sa valeur.

Par convention, une constante se déclare toujours en majuscule.

Nous demandons ensuite l'affichage du contenu de la constante nommée CAPITALE : Paris.

résultat

Paris

*Nous ne pourrons pas déclarer d'autres valeurs à l'intérieur de la constante nommée CAPITALE*

A quoi peut bien servir une constante ? devons nous utiliser plutôt une variable ou une constante ?

Dans la majorité des cas nous utilisons des variables, cela dépend surtout de la nature de l'information à garder.

Comme nous l'avons évoqué, le contenu d'une constante ne pourra pas être redéfinie durant l'exécution du script, il peut donc être utile de garder certaines informations de manière sûre comme les informations de connexion à une base de données, le chemin fixe vers un dossier du site, etc.

26

## Les constantes magiques

D'autres constantes existent dans le langage PHP, on les appelle des constantes magiques, exemple :

entrainement.php

```
1 <?php
2 echo __FILE__ . "<br>"; // Affiche le chemin complet vers le fichier actuel
3 echo __LINE__ . "<br>"; // Affiche le numéro de la ligne
4
```

D'autres constantes magiques existent, [voici le lien vers la documentation officielle de PHP](#).

## Syntaxe et concaténation

27

## Concaténation lors d'un affichage

La concaténation en PHP permet de faire suivre des informations sur la même ligne d'instruction (avec un `echo` par exemple).

entrainement.php

```
1 <?php
2 $x = "bonjour "; // affectation de la valeur "bonjour" dans la variable $x.
3 $y = "tout le monde"; // affectation de la valeur "tout le monde" dans la variable $y.
4 echo $x . " " . $y . "<br>"; // point de concaténation que l'on peut traduire par "suivi de".
5 echo "$x $y <br>"; // même chose que la ligne d'en dessus, sans concaténation.
6 echo "Hey ! " . $x . " " . $y; // concaténation de texte et de variables.
7 echo "<br>", "Hello ", $y, "<br>"; // concaténation avec l'utilisation d'une virgule (la virgule et le point de concaténation sont similaires).
```

### Explications

Ligne 2 : affectation de la valeur "bonjour" dans la variable \$x.

Ligne 3 : affectation de la valeur "tout le monde" dans la variable \$y.

Ligne 4 : affichage de la variable \$x, suivi d'un espace, suivi de la variable \$y et d'un passage à la ligne (balise br).

Ligne 5 : Même chose que la ligne 4, inscrit différemment.

Ligne 6 : Affichage du texte "Hey ! ", suivi de l'affichage de la variable \$x, un espace et \$y.

Ligne 7 : Affichage d'un passage à la ligne, du texte "Hello ", suivi de l'affichage de la variable \$y.

La concaténation peut se faire avec l'utilisation d'un point ou d'une virgule (pas de virgule dans le cas de l'instruction `print`).

La virgule et le point de concaténation sont donc similaires lorsque nous utilisons `echo`.

résultat

bonjour tout le monde

bonjour tout le monde

Hey ! bonjour tout le monde

Hello tout le monde

28

## Concaténation lors d'une affectation

entraînement.php

```
1 <?php
2 $prenom1 = "Bruno"; // Affectation de la valeur "Bruno" sur la variable : $prenom1
3 $prenom1 = "Claire"; // Affectation de la valeur "Claire" sur la variable : $prenom1. cela remplace Bruno par Claire.
4 echo $prenom1 . "<br>"; // Affiche : Claire
5 $prenom2 = "Nicolas"; // Affectation de la valeur "Nicolas" sur la variable : $prenom2
6 $prenom2 .= " Marie"; // Affectation de la valeur " Marie" sur la variable : $prenom2. Ajout SANS remplacement de la valeur précédente
7 grâce à l'opérateur =
8 echo $prenom2. "<br>"; // Affiche : Nicolas Marie
```

### Explications

Ligne 2 : affectation de la valeur "Bruno" dans la variable \$prenom1.

Ligne 3 : affectation de la valeur "Claire" dans la variable \$prenom1 (cela remplace Bruno par Claire).

Ligne 4 : affichage de la variable \$prenom1, soit : Claire.

Ligne 5 : affectation de la valeur "Nicolas" dans la variable \$prenom2.

Ligne 6 : affectation de la valeur "Marie" dans la variable \$prenom2, avec l'utilisation de l'opérateur de concaténation lors de l'affectation ".=", cela ajoute Marie dans la variable sans pour autant retirer ou remplacer la précédente valeur "Nicolas".

Ligne 7 : Affichage de la variable \$prenom2, soit Nicolas Marie.

résultat

Claire

Nicolas Marie

Nous retiendrons que pour inscrire une nouvelle valeur dans une variable (sans pour autant remplacer la valeur précédente), nous utiliserons l'opérateur .= permettant de faire une concaténation dans le contenu de la variable.

Pour autant, cela ne sera pas pratique lors de l'affichage, en effet, nous n'aurons pas la possibilité d'afficher soit Marie, soit Nicolas, les deux valeurs s'afficheront systématiquement.

Nous voyons donc que nous n'utiliserons pas une variable pour conserver plusieurs valeurs (pour ça, nous verrons les array au chapitre 12), retenons qu'il est préférable de réfléchir de la sorte :

1 variable = 1 valeur

29

## Syntaxe : Différence entre l'utilisation des guillemets et des quotes

Voyons le code suivant :

entraînement.php

```

1  <?php
2  $jour = 'aujourd\'hui'; // utilisation des quotes avec l'anti-slash pour éviter une erreur sur l'apostrophe du mot : aujourd'hui.
3  $jour = "aujourd'hui"; // utilisation des guillemets
4  $texte = 'bonjour'; // utilisation des quotes
5  echo $texte . " tout le monde<br>"; // Concaténation d'une variable et de texte
6  echo "$texte tout le monde<br>"; // Affichage dans des guillemets, la variable est évaluée. Pas de concaténation.
7  echo '$texte tout le monde<br>'; // Affichage dans des quotes, la variable n'est pas évaluée !

```

### Explications

Ligne 2 : affectation de la valeur *aujourd'hui* dans la variable \$jour, nous prévoyons un anti-slash "\\" pour éviter que l'apostrophe du mot *aujourd'hui* soit considéré par PHP comme la fin de l'affectation, ce qui n'est pas le cas et ferait une erreur php à coup sur.

Avez-vous compris à quoi servait l'anti-slash "\\" ?

Ligne 3 : affectation de la valeur *aujourd'hui* dans la variable \$jour en utilisant des guillemets, pas d'anti-slash à prévoir car pas de conflit possible sur ce coup-là.

Ligne 4 : affectation de la valeur *bonjour* dans la variable \$texte en utilisant des quotes (cela aurait pu aussi être fait avec des guillemets).

Ligne 5 : Affichage du texte "Bonjour tout le monde", avec un point de concaténation entre la variable et le texte.

Ligne 6 : Affichage du texte "Bonjour tout le monde", sans concaténation entre la variable et le texte.

Nous pouvons donc dire qu'entre guillemets la variable est évaluée et que la concaténation n'est pas indispensable

Ligne 7 : Affichage du texte '\$texte tout le monde', sans concaténation entre la variable et le texte. Cela nous montre la différence entre les guillemets et les quotes lors d'un affichage de variable.

### Déférence entre les guillemets et les quotes

Lors d'une affectation, il n'y a pas vraiment de différence (les développeurs utilisent plus couramment les quotes).

Lors d'un affichage en php, avec des guillemets la variable est évaluée (c'est son contenu qui s'affiche), tandis qu'en utilisant les quotes lors de ce même affichage, c'est le nom de la variable qui est affiché (variable non évaluée)

résultat

bonjour tout le monde

bonjour tout le monde

\$texte tout le monde

## Conditions et opérateurs

30

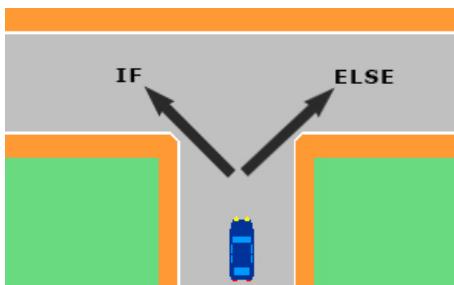
### Les Conditions

Les conditions est l'un des chapitres clé de notre apprentissage, c'est aussi un sujet présent dans tous les autres langages de programmation.

Par exemple, lorsque vous créez un site web et que l'un de vos internautes voudra se connecter à son espace de profil, il rentrera son pseudo ainsi que son mot de passe (normal, comme partout).

Il faudra exprimer la phrase suivante en PHP : "si (if) tu as un bon pseudo et un bon mot de passe, je te laisse rentrer dans ton espace de profil, sinon (else) tu resteras sur la page de connexion avec un message d'erreur".

en clair, il faut voir une condition comme une intersection dans la ville pour une voiture, plusieurs routes existent et en fonction de la situation, l'une des deux routes sera utilisée :



Avant de voir cela plus en détail par le code, voyons les conditions mises à notre disposition :

Condition	Signification
if	si
else	sinon
elseif	sinon si

31

## Les Opérateurs

Voici les opérateurs mis à notre disposition :

Opérateur	Signification
=	est égal
==	comparaison de la valeur
===	comparaison de la valeur et du type
!=	différent de
!	n'est pas
>	supérieur à ...
<	inférieur à ...
>=	supérieur ou égal à ...
<=	inférieur ou égal à ...
&& AND	et
 OR	ou
XOR	ou exclusif

32

## La condition IF/ELSE

entrainement.php

```

1 | <?php
2 | $a = 10;
3 | $b = 5;
4 | $c = 2;
5 |

```

```

6 | if($a > $b) // Si A est supérieur à B
7 | {
8 |   echo "A est bien supérieur à B <br>";
9 | }
10| else // Sinon ... A n'est pas supérieur à B
11| {
12|   echo "non, c'est B qui est supérieur à A <br>";
13| }

```

Nous déclarons 3 variables avec :

- A égal à 10
- B égal à 5
- C égal à 2

Ensuite, nous demandons si la variable A est bien supérieure à B, en clair : est-ce que 10 est supérieur à 5 ?

Oui, 10 est plus grand que 5 ! alors la condition renvoie TRUE (vrai) et nous rentrons dans les accolades du IF.

résultat

A est bien supérieur à B

Si nous rentrons dans le IF, nous ne pouvons pas rentrer dans le ELSE. C'est soit l'un, soit l'autre.

L'instruction ELSE fonctionne après un IF et exécute les instructions correspondantes au cas où la condition du IF est fausse.



#### ➤ Informations

Le ELSE n'a jamais de parenthèse ni de condition particulière, il s'agit du cas par défaut.

33

## La condition avec AND (&&)

Voici un exemple avec une double condition devant être parfaitement valable :

entraînement.php

```

1 | <?php
2 | if($a > $b && $b > $c) // Si A est supérieur à B et que dans le même temps B est supérieur à C
3 | {
4 |   echo "A est supérieur à B, ET, B est supérieur à C <br>";
5 | }

```

*Nous pouvons mettre plusieurs lignes de traitement dans les accolades si nous en avons la nécessité*

Cette fois ci, nous demandons si A est supérieur à B, donc : est-ce que 10 est plus grand que 5 ?

Et aussi, si B est supérieur à C, ce qui donne : est-ce que 5 est supérieur à 2 ?

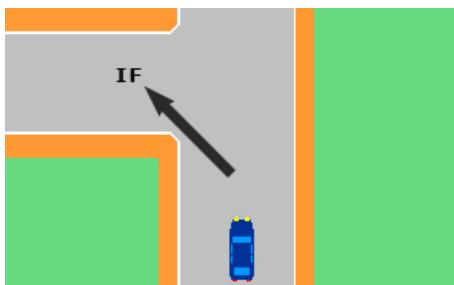
Oui A est plus grand que B, et dans le même temps, B est supérieur à C ! Donc la condition renvoie TRUE (vrai) et nous rentrons dans les accolades du IF.

résultat

A est supérieur à B, ET, B est supérieur à C

Notez qu'il n'y a pas de ELSE, il est possible de faire un IF sans ELSE.

Ca ressemblerait plutôt à ça au niveau de notre route :



en clair, nous devons continuer tout droit, mais toutefois si la condition renvoie vrai, nous serions alors déviés de notre chemin pour aller sur une autre route.

Avec le signe `&&` (prononcé AND, et commercial, esperluette), il faut absolument que les 2 conditions soient bonnes pour que l'évaluation du code retourne TRUE et que l'on puisse rentrer dans les accolades du IF.

34

## La condition avec OR (||)

Voici un exemple avec une double condition (où l'une des deux doit au minimum être valable) :

entraînement.php

```

1 | <?php
2 | if($a == 9 || $b > $c) // Si A contient 9 ou que B est supérieur à C
3 | {
4 |   echo "Au moins l'une des 2 conditions est bonne (ou peut-être les 2), sinon vous ne me verriez pas m'afficher ! <br>";
5 |

```

Cette fois ci, nous demandons si A est égal à 9, donc : est-ce qu'il y a 9 dans A ?

Et aussi, si B est supérieur à C, ce qui donne : est-ce que 5 est supérieur à 2 ?

Non A n'est pas égal à 9 (mais à 10), par contre B est bien supérieur à C ! Donc le fait qu'au moins l'une des deux conditions renvoie TRUE (vrai) cela nous permet de rentrer dans les accolades du IF.

résultat

Au moins l'une des 2 conditions est bonne (ou peut-être les 2), sinon vous ne me verriez pas m'afficher !

Si nous ne rentrions pas dans le IF, l'interpréteur continuerait d'évaluer la suite du script en ignorant cette partie de notre code.

Pour rappel, voici les variables que nous avions déclarées précédemment :

```

1 | <?php
2 | $a = 10;
3 | $b = 5;
4 | $c = 2;

```

Notez qu'il n'y a pas de ELSE, il est possible de faire un IF sans ELSE.

Avec le signe `||` (prononcé OR, ou), il faut qu'au moins l'une des 2 conditions soit bonne pour que l'évaluation du code retourne TRUE et que l'on puisse rentrer dans les accolades du IF.

Si les 2 conditions étaient bonnes en même temps, nous serions aussi rentrés dans le IF, exemple :

entraînement.php

```

1 | <?php
2 | if($a == 10 || $b > $c) // Si A contient 10 ou que B est supérieur à C
3 | {
4 |   echo "Au moins l'une des 2 conditions est bonne (ou peut-être les 2), sinon vous ne me verriez pas m'afficher ! <br>";
5 |

```

A est bien égal à 10.

B est bien supérieur à C.

Comme tout à l'heure, nous rentrons dans le IF.

35

## Différence entre AND (&&) et OR (||)

Dans le cas du AND, il faut absolument que les 2 conditions soient bonnes pour rentrer dans le IF.

Dans le cas du OR, les 2 conditions (ou seulement l'une des 2 conditions) doivent être bonnes pour rentrer dans le IF.

36

## La condition XOR

Voici un exemple avec une double condition (où l'une des deux doit obligatoirement être valable) :

entraînement.php

```
1 <?php
2 if($a == 10 XOR $b == 6) // XOR : seulement l'une des 2 conditions doit être valide.
3 {
4     echo 'Une seule des 2 conditions est bonne !<br>';
5 }
```

Nous demandons si A est égal à 10, donc : est-ce qu'il y a 10 dans A ?

Ensuite nous demandons si B est égal à 6, donc : y'a t'il 6 dans B ?

Oui A est égal à 10 mais B n'est pas égal à 6.

L'une des 2 conditions est bonne, l'autre pas. C'est parfait ! c'est précisément ce dont nous avons besoin pour rentrer dans le IF.

Si les 2 conditions sont bonnes, nous ne rentrons pas dans le IF.

Si les 2 conditions sont mauvaises, nous ne rentrons pas dans le IF.

Avec XOR, il faut précisément qu'une seule des 2 conditions soit bonne pour rentrer dans le IF.

résultat

Une seule des 2 conditions est bonne !

Son utilisation est un peu plus rare mais très utile dans certains cas très spécifiques !

37

## Les conditions avec IF / ELSEIF / ELSE

Voici un exemple avec une condition comprenant plusieurs cas :

entraînement.php

```
1 <?php
2 if($a == 8) // le double égal (==) permet de vérifier une information à l'intérieur d'une variable
3 {
4     echo "1 - A est égal à 8";
```

```

5 }
6 elseif($a != 10) // Sinon Si A est différent de 10
7 {
8     echo "2 - A est différent de 10";
9 }
10 else // Sinon, c'est que je ne suis ni tombé dans le if, ni dans le elseif, je me trouve donc ici dans le else.
11 {
12     echo "3 - tout le monde a faux A n'est pas égal à 8 et n'est pas différent de 10. Du coup c'est moi le else qui gagne !<br>";
13 }

```

IF : Nous demandons si A est égal à 8, donc : est-ce qu'il y a 8 dans A ?

ELSEIF : Sinon si, est-ce que A est différent de 10 ?

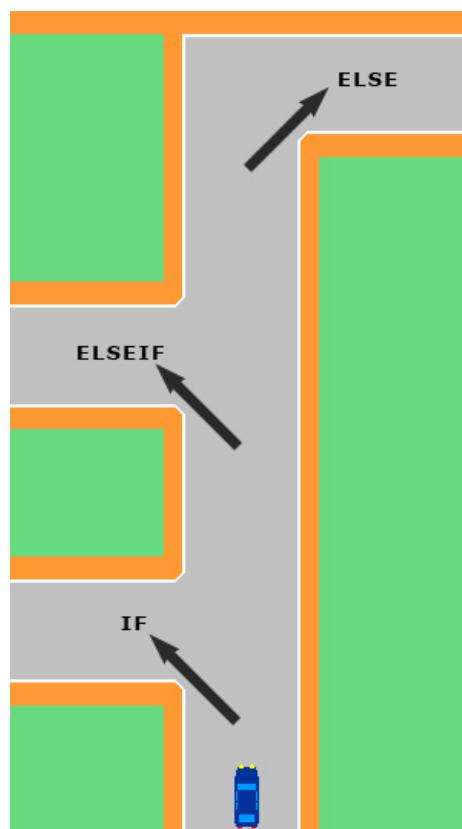
ELSE : Sinon, si les 2 conditions d'au dessus ne sont pas bonnes, nous tomberons dans le else (cas par défaut). Non A n'est pas égal à 8 (mais à 10), A n'est pas non plus différent de 10 ! Du coup nous tomberons forcément dans le cas par défaut, le ELSE.

résultat

3 - tout le monde a faux A n'est pas égal à 8 et n'est pas différent de 10. Du coup c'est moi le else qui gagne !

Nous aurions pu mettre plusieurs elseif d'affilée, le nombre n'est pas limité.

Pour imaginer le code, voici ce que cela donne au niveau de notre route :



Si nous ne rentrons pas dans le IF, ni dans le ELSEIF, nous rentrons forcément dans le ELSE, c'est le cas par défaut (la route restante, puisque nous ne pouvons pas revenir en arrière).

```

1 | <?php
2 | echo ($a == 10) ? "A est égal à 10<br>" : "A n'est pas égal à 10<br>";

```

Le ? (point d'intérogation) remplace le IF, le : (double point) remplace le else.

Nous ne garderons pas forcément cette forme d'écriture durant le cours mais il est utile d'en avoir connaissance si vous récupérez du code écrit par d'autres développeurs (utilisant les formes contractées) qu'il faudra comprendre.

Si A est égal à 10, nous rentrons dans le IF (point d'intérogation "?").

Sinon, A n'est pas égal à 10, nous rentrons dans le ELSE (double point ":").

résultat

A est égal à 10

39

## Condition de comparaison

Voici un exemple avec une condition de comparaison :

```

1 | <?php
2 | $vara = 1;
3 | $varb = "1";
4 | if($vara == $varb)
5 | {
6 |   echo "il s'agit de la même valeur";
7 |

```

Nous demandons si la \$vara contient la même valeur que la \$varb.

Oui c'est bien, le cas, dans les deux variables se trouve la valeur 1.

résultat

il s'agit de la même valeur

```

1 | <?php
2 | $vara = 1;
3 | $varb = "1";
4 | if($vara === $varb)
5 | {
6 |   echo "il s'agit de la même valeur et du même type";
7 |

```

résultat

Le résultat est vide dans la mesure où la condition renvoie FALSE (faux), nous ne rentrons pas dans le IF, par conséquent le echo ne se déclenchera pas.

Nous demandons si la \$vara contient la même valeur et le même type que la \$varb.

Non ce n'est pas le cas, la \$vara contient le chiffre 1 et est de type integer, la \$varb contient le chiffre 1 et est de type string.

Comme vous avez pu le constater, dans notre code la variable \$vara contient un 1 qui a été affecté normalement comme un chiffre (sans quotes ni guillemets).

La variable \$varb contient un 1 qui a été affecté comme un string (une chaîne de caractères), avec des guillemets.

L'interpréteur ne fait pas de différenciation de type avec le double égal, mais avec le triple égal il voit une différence et n'est pas d'accord pour dire que

le type est le même.

Différence entre 1 égal, 2 égals et 3 égaux en PHP ?

Opérateur	Signification
=	1 égal permet de faire une affectation de valeur à une variable
==	2 égaux permettent de faire une comparaison de la valeur et poser une question, par exemple : y'a t'il 1 dans la variable ?
===	3 égaux permettent de faire une comparaison de la valeur et du type afin de poser une question, par exemple : y'a t'il 1 dans la variable et est-elle de type integer ?

40

## Condition Switch

Lorsque vous avez un grand nombre de conditions à tester (supérieur à 3), nous utilisons généralement la structure conditionnelle SWITCH.

Voici ce que le code donnerait avec les structures que nous connaissons IF, ELSEIF, ELSE :

```
entraînement.php
?
1 <?php
2 $monPays = 'France';
3
4 if($monPays == 'Italie')
5 {
6     echo 'vous êtes Italien!<br>';
7 }
8 elseif($monPays == 'Espagne')
9 {
10    echo 'vous êtes Espagnol!<br>';
11 }
12 elseif($monPays == 'Angleterre')
13 {
14    echo 'vous êtes Anglais!<br>';
15 }
16 elseif($monPays == 'France')
17 {
18    echo 'vous êtes Français!<br>';
19 }
20 elseif($monPays == 'Suisse')
21 {
22    echo 'vous êtes Suisse!<br>';
23 }
24 else
25 {
26    echo 'vous n\'avez pas une nationalité connue dans notre liste de possibilités<br>';
27 }
```

résultat

vous êtes Français!

S'il y a beaucoup de tests à réaliser, l'utilisation de IF, ELSEIF, ELSE va être assez contraignante et peu optimisée pour les performances (temps d'exécution du serveur plus faible une fois le code mis en ligne avec des multi connexions).

Par conséquent, retenez que dans le cas d'un grand nombre de conditions à tester, il sera préférable d'utiliser la structure conditionnelle switch :

```
entraînement.php
?
1 <?php
2 $monPays = 'France';
3
4 switch($monPays)
5 {
6     case 'Italie':
7         echo 'vous êtes Italien!<br>';
8         break;
9     case 'Espagne':
10        echo 'vous êtes Espagnol!<br>';
11        break;
12     case 'Angleterre':
13         echo 'vous êtes Anglais!<br>';
14 }
```

```

14 break;
15 case 'France':
16     echo 'vous êtes Français!<br>';
17 break;
18 case 'Suisse':
19     echo 'vous êtes Suisse!<br>';
20 break;
21 default:
22     echo 'vous n\'avez pas une nationalité connue dans notre liste de possibilités<br>';
23 break;
24 }

```

Je vous recommande d'être particulièrement vigilant(e) à l'écriture du code dans la mesure où nous alternons des quotes, des point-virgule, des doubles-point, des accolades, des parenthèses, etc.

résultat

vous êtes Français!

### Explications

Nous inscrivons la variable à tester dans les parenthèses de la structure switch.

Nous listons tous les cas à tester avec le mot-clé `case` et nous délimitons un cas avec le mot-clé `break` (en remplacement des accolades habituelles).

Le cas par défaut (équivalent du `else`) est mentionné par le mot-clé `default`.

41

## Définition entre IF et ELSEIF

Beaucoup diront "c'est bon, pas besoin d'expliquer pendant 2 jours ce que veut dire IF et ELSE, on a capté !" mais il est très important de ne pas confondre les IF et les ELSEIF, et par dessus tout, comprendre pourquoi.

La différence entre IF et ELSEIF :

Les instructions IF seront toutes testées à la chaîne, les unes après les autres.

Les instructions ELSEIF seront testées uniquement si les conditions au dessus n'ont pas été évaluées comme TRUE (vrai) et n'ont pas donné satisfaction.

Pour mieux comprendre, voici un exemple avec une série de conditions comprenant plusieurs IF et un ELSE à la fin :

entraînement.php

```

1 <?php
2 $couleur = 'bleu';
3
4 if($couleur == 'bleu')
5 {
6     echo 'votre couleur est préférée est le bleu.<br>';
7 }
8 if($couleur == 'orange')
9 {
10    echo 'votre couleur est préférée est le orange.<br>';
11 }
12 if($couleur == 'rose')
13 {
14    echo 'votre couleur est préférée est le rose.<br>';
15 }
16 else
17 {
18    echo 'nous n\'arrivons pas à déterminer votre couleur préférée !!<br>';
19 }

```

Qu'est-ce que donne ce code ? Voyons ça tout de suite...

résultat

votre couleur est préférée est le bleu.

nous n'arrivons pas à déterminer votre couleur préférée !!

### Explications

Et oui, vous ne vous trompez pas, le if de la couleur bleue est bien exécuté, mais aussi le else de fin.

Pourquoi ? Et bien c'est simple, l'interpréteur lit les instructions séquentiellement.

Ligne 4 : il voit que la couleur est bleu et rentre dans le IF.

Ligne 8 : la couleur n'est pas orange et ne rentre pas dans le IF.

Ligne 12 : la couleur n'est pas rose et ne rentre pas dans le IF.

Ligne 16 : Puisque la couleur n'était pas rose (cas juste au dessus), il rentre dans le ELSE (cas par défaut) car chaque else se réfère au IF qui le précède !

C'est donc un problème ! voyons plutôt ce code :

```
entrainement.php  
1 | <?php  
2 | $couleur = 'bleu';  
3 |  
4 | if($couleur == 'bleu')  
5 | {  
6 |     echo 'votre couleur est préférée est le bleu.<br>';  
7 | }  
8 | elseif($couleur == 'orange')  
9 | {  
10|     echo 'votre couleur est préférée est le orange.<br>';  
11| }  
12| elseif($couleur == 'rose')  
13| {  
14|     echo 'votre couleur est préférée est le rose.<br>';  
15| }  
16| else  
17| {  
18|     echo 'nous n\'arrivons pas à déterminer votre couleur préférée !!<br>';  
19| }
```

Cette fois ce code est mieux formulé, voyons ce que ça donne :

résultat

votre couleur préférée est le bleu

### Explications

Ligne 4 : il voit que la couleur est bleu et rentre dans le IF.

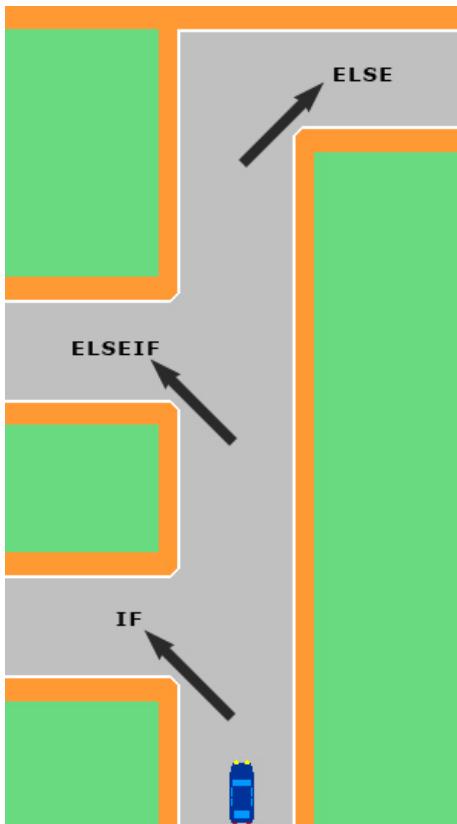
Ligne 8 : il ne cherche pas à rentrer dans un ELSEIF puisque le IF a été évalué comme étant positif (TRUE, vrai).

Ligne 12 : il ne cherche pas à rentrer dans un ELSEIF puisque le IF a été évalué comme étant positif (TRUE, vrai).

Ligne 16 : il ne cherche pas non plus à rentrer dans un ELSE puisque le IF a été évalué comme étant positif (TRUE, vrai).

En conclusion, nous pouvons dire qu'il y a une réelle et importante différence entre IF et ELSEIF.

Si votre objectif est de prévoir qu'un seul cas possible pour vos conditions, vous devrez privilégier la structure IF / ELSEIF / ELSE ou éventuellement la structure conditionnelle SWITCH.



## Les fonctions prédéfinies

42

## Les Fonctions Prédéfinies

Qu'est-ce qu'une fonction prédéfinie ?

Une fonction est un morceau de code permettant d'automatiser un traitement ou de donner un affichage particulier.

Dans la langue Française, vous avez la possibilité de consulter un dictionnaire pour voir les mots et expressions existant(e)s.

Tout comme notre langue, Php possède plusieurs mots-clés et fonctions déjà existants dans le langage, nous pourrons les retrouver dans la [documentation officielle de PHP](#).

43

## Afficher la date du jour avec la fonction date()

Prenons un exemple pour afficher la date du jour :

```

entraînement.php
?
1 | <?php
2 | echo 'Date : <br>';
3 | echo date("d/m/Y");

```

Lorsque vous voyez un mot-clé (qui n'est pas le vôtre), en anglais, suivi de parenthèse, c'est qu'il s'agit d'une fonction prédéfinie de PHP.

### Explications

La [fonction date](#) permet d'afficher la date avec les arguments suivants :

- d : pour day (jour)
- m : pour month (mois)
- Y : pour année

*Le fait de mettre les arguments (d/m/Y) en minuscule ou majuscule à une incidence sur le résultat. Faites le test !*

44

## Qu'est-ce qu'un argument ?

Un argument (aussi appelé paramètre) est une information entrante dans la fonction.

45

## Afficher la taille d'une chaîne avec la fonction strlen()

entraînement.php

```
1 | <?php
2 | $phrase = 'Mettez une phrase différente ici à cet endroit';
3 | echo strlen($phrase);
```

Nous transmettons l'argument (argument = paramètre) \$phrase à la fonction prédéfinie strlen().

résultat

47

[strlen\(\)](#) est une fonction prédéfinie permettant de retourner la taille (nombre de caractères) d'une chaîne.

Les valeurs de retours sont :

- En cas de succès : INT
- En cas d'échec : BOOLEAN FALSE

Argument(s) :

1. Nous devons lui fournir la chaîne à évaluer (pour connaître la taille).

Il faut lui envoyer un texte, une phrase ou un mot (par l'intermédiaire d'une variable ou non), car pour vous renvoyer "la taille de quelque chose", il lui faut ce "quelque chose", cette fonction a donc besoin d'une information de départ pour compter le nombre de caractères, on appelle cette information un argument (ou paramètre).

Contexte : cela pourra être pratique lorsque nous souhaiterons connaître le nombre de caractères dans un pseudo qui nous est transmis (lors d'une inscription).

## entraînement.php

```

1 | <?php
2 | $texte = "Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard
3 | dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has
   | survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged.";
   | echo substr($texte, 0, 20) . "...<a href=\"\"> Lire la suite </a>";

```

Nous transmettons l'argument (aussi appelé paramètre) \$texte à la fonction pré définie substr() ainsi que 2 nombres (int).

## résultat

Lorem Ipsum is simpl... Lire la suite

[substr\(\)](#) est une fonction pré définie permettant d'afficher une partie coupée d'une chaîne.

Les valeurs de retours sont :

En cas de succès : STRING

En cas d'échec : BOOLEAN FALSE

Argument(s) attendus :

1. nous devons lui fournir la chaîne que l'on souhaite couper (string)
2. nous devons lui préciser la position de début (int)
3. nous devons lui préciser la position de fin (int)

Dans le cas présent, nous lui demandons d'afficher seulement les 20 premiers caractères de notre phrase (0,20).

Vous pouvez demander les 50 premiers caractères en modifiant les arguments entrants par : 0,50.

Contexte: substr est souvent utilisé pour afficher des actualités avec une "accroche".

Sur le web, il n'est pas rare de voir le début d'un article (l'accroche) puis ...lire la suite.

Cela ne veut pas dire que c'est toujours fait avec la fonction substr, parfois le texte affiché est un résumé et non le début de l'article.

Pour ne pas couper un mot il y a d'autres fonctions pré définies qui existent.

Il est possible de se servir de fonction pré définie dans le cadre de nos conditions, voici un exemple avec ISSET :

## entraînement.php

```

1 | <?php
2 | $pseudo = "joker";
3 | if(isset($pseudo))
4 | {
5 |   echo 'la variable $pseudo existe!<br>';
6 | }
7 | else
8 | {
9 |   echo 'la variable $pseudo n'existe PAS!<br>';
10 |

```

## Résultat

la variable \$pseudo existe!

Autre cas, cette fois-ci, sans déclarer la variable \$pseudo :

```
entrainement.php
```

```
1 | <?php
2 | if(isset($pseudo))
3 | {
4 |     echo 'la variable $pseudo existe!<br>';
5 | }
6 | else
7 | {
8 |     echo 'la variable $pseudo n'existe PAS!<br>';
9 | }
```

#### Résultat

la variable \$pseudo n'existe PAS!

[isset\(\)](#) est une fonction prédéfinie permettant de dire si une variable existe ou non (et différent de null).

Les valeurs de retours sont :

En cas de succès : boolean TRUE (la variable existe)

En cas d'échec : boolean FALSE (la variable n'existe pas)

Argument(s) attendus :

1. nous devons lui fournir la variable que l'on souhaite tester

48

## Tester le contenu d'une variable avec empty()

Différent de isset, voici un exemple avec la fonction prédéfinie EMPTY :

```
entrainement.php
```

```
1 | <?php
2 | $pseudo = "joker";
3 | if(empty($pseudo))
4 | {
5 |     echo 'la variable $pseudo est vide!<br>';
6 | }
7 | else
8 | {
9 |     echo 'la variable $pseudo n'est pas vide (et est donc remplie).<br>';
10| }
```

#### Résultat

la variable \$pseudo n'est pas vide (et est donc remplie)!

Autre cas, cette fois-ci, sans remplir la variable \$pseudo :

```
entrainement.php
```

```
1 | <?php
2 | $pseudo = "";
3 | if(empty($pseudo))
4 | {
5 |     echo 'la variable $pseudo est vide!<br>';
6 | }
7 | else
8 | {
9 |     echo 'la variable $pseudo n'est pas vide (et est donc remplie)!<br>';
10| }
```

#### Résultat

la variable \$pseudo est vide!

[empty\(\)](#) est une fonction prédéfinie permettant de dire si une variable est vide ou non. (False sera considéré comme empty).

Les valeurs de retours sont :

En cas de succès : boolean TRUE (la variable est vide)

En cas d'échec : boolean FALSE (la variable n'est pas vide et est donc remplie)

Argument(s) attendus :

1. nous devons lui fournir la variable que l'on souhaite tester

Il est également possible de demander l'inverse "si ça n'est pas vide" avec l'opérateur !

```
entraînement.php
?
1 | <?php
2 | $pseudo = "joker";
3 | if(empty($pseudo))
4 | {
5 |   echo 'la variable $pseudo n\'est pas vide (et est donc remplie)!<br>';
6 | }
```

La simple présence du "!" (point d'exclamation) dans les parenthèses du IF permet de changer le sens de la condition, cette fois nous demandons : est-ce que la variable \$pseudo n'est pas vide ? (en gros, est-elle remplie ?).

#### Résultat

la variable \$pseudo n'est pas vide (et est donc remplie)!

De nombreuses autres fonctions prédéfinies existent et pour toute sorte de chose, nous y reviendrons, d'ici là vous pouvez mettre en favoris le site de la [documentation officielle de PHP](#).

*On dit souvent qu'un bon développeur n'est pas celui qui connaît le code par cœur mais celui qui sait lire et interpréter une documentation, avoir les bons mots-clés dans ses recherches Google, et être autonome dans l'exploitation et la mise en oeuvre du code.*

## Les fonctions utilisateurs

49

## Les Fonctions Utilisateur

Qu'est-ce qu'une fonction utilisateur ?

Une fonction est un morceau de code permettant d'automatiser un traitement ou de donner un affichage particulier.

Cela permet également de ré-utiliser un morceau de code à plusieurs reprises.

Imaginez la problématique suivante :

Vous souhaitez calculer votre imposition, cela vous demande plusieurs lignes de calcul, de la réflexion et vous finissez par y parvenir.

Ensuite, vous devrez reprendre ces lignes chaque année puisque l'imposition est annuelle.

Plutôt qu'effectuer un copier/coller, il serait judicieux d'enfermer votre calcul dans une fonction que vous pourrez rappeler à tout moment lorsque vous en aurez envie.

D'autre part, votre situation peut changer, vous vous mariez et avez des enfants, le calcul d'imposition n'est plus le même (car vous avez le droit à des abatements).

Plutôt que modifier le coeur de votre calcul, vous pourriez ajouter ce que l'on appelle des paramètres (ou arguments). Ce sont des informations variables qui peuvent venir modifier le comportement initialement prévu par la fonction ou bien compléter son traitement.

De cette manière votre fonction sera adaptée à tout le monde et à toutes les situations possibles (marié ou célibataire, avec ou sans enfants), pour

l'appliquer il faudra juste le préciser au moment d'exécuter la fonction.

A l'inverse des fonctions prédéfinies (qui sont déjà prévues par le langage), c'est à nous d'écrire nos propres fonctions spécifiques (lorsqu'aucune fonction de la documentation peut nous convenir).

Cela pourrait s'appeler *fonction développeur*.

50

## Créons notre première fonction pour calculer une TVA

Prenons le contexte d'un site ecommerce qui aura besoin d'ajouter la TVA sur ses produits.

Déclarons notre première fonction pour calculer la TVA sur 100 € :

```
entraînement.php
```

```
1 | <?php
2 | // déclaration d'une fonction
3 | function calculTva()
4 | {
5 |     return 100*1.2;
6 | }
7 |
8 | // exécution de notre fonction
9 | echo calculTva() . <br>;
```

### Résultat

120

### Explications

Pour déclarer une fonction, nous utilisons le mot-clé `function` et ensuite nous lui donnons un nom (sans espace, ni accent), ici elle s'appelle : `calculTva`.

Les parenthèses permettent de préciser s'il y a des arguments entrants, ce n'est pas notre cas, notre fonction est donc destinée à ne pas recevoir d'arguments.

Les accolades définissent le périmètre de la fonction. A l'intérieur des accolades nous pouvons écrire des lignes de traitement PHP (ces lignes feront partie de la fonction).

Le mot-clé `return` est utilisé pour retourner un résultat. Il est en général placé à la fin pour terminer la fonction.

Le calcul  $120 \times 1.2$  (100 multiplié par 1, puis 0.2 fait bien 120). 1.2 correspond bien à 20 % (la TVA Française la plus courante en vigueur).

51

## Créons une fonction avec un argument pour faire plus de calculs

Le site ecommerce (que nous avons pris comme sujet) ne vendra pas que des produits à 100 €.

Nous allons améliorer notre fonction afin de pouvoir effectuer tous les calculs et non pas que sur un nombre fixe de 100 €.

Pour cela nous allons avoir besoin d'un argument entrant (le prix, ici appelée `$nombre`) :

```
entraînement.php
```

```
1 | <?php
```

```
2 // déclaration d'une fonction
3 function calculTva2($nombre)
4 {
5     return $nombre*1.2;
6 }
7
8 // exécution de notre fonction
9 echo calculTva2(500) . '<br>';
10 echo calculTva2(1000) . '<br>';
```

## Résultat

600

1200

## Explications

Cette fois-ci, dans les parenthèses de notre déclaration de fonction, nous avons prévu une variable nommée `$nombre`, cette variable pourrait très bien être nommée `$x` ou `$y`. Peu importe tant qu'il y'a une variable de réception de prévue. J'ai choisi de la nommer `$nombre` car c'est plus cohérent avec le sujet.

Le calcul a également changé, ce n'est plus `100*1.2` car ce calcul est fixe et donne toujours le même résultat. Cette fois-ci nous allons calculer 20% sur le nombre que nous allons recevoir `$nombre*1.2`, et il peut donc changer à tout moment.

En effet, une fonction est déclarée une fois mais peut être exécutée autant de fois que nécessaire.

Dans notre cas nous réalisons une première exécution :

```
1 | echo calculTva2(500) . '<br>';
```

Ce qui donne comme calcul :  $500 * 1.2 = 600$  €

Et, une deuxième exécution :

```
1 | echo calculTva2(1000) . '<br>';
```

Ce qui donne comme calcul :  $1000 * 1.2 = 1200$  €

La fonction a été renommée en `calculTva2` pour éviter tout conflit avec la 1ère fonction `calculTva` si vous l'aviez gardée présente dans le même fichier. En effet, nous ne pouvons pas déclarer 2 fonctions du même nom au sein du même fichier au risque d'avoir une erreur du type : *Fatal error: Cannot redeclare ... (previously declared in ...)* on line ...

Ce message d'erreur précise l'endroit de la 1ère déclaration afin que vous puissiez faire un choix entre garder la première, ou la deuxième ou éventuellement fusionner les deux.

Au moment de la déclaration, lorsqu'une fonction est destinée à recevoir 1 argument, vous devrez obligatoirement lui envoyer 1 argument au moment de l'exécution au risque d'avoir une erreur du type : *Warning: Missing argument ... for ...()*

Pour une meilleure compréhension, voici le chemin parcouru par l'argument :

```
<?php
// déclaration d'une fonction
function calculTva2 ($nombre)
{
    return $nombre*1.2;
}
|
// exécution de notre fonction
echo calculTva2(500) . '<br>';
echo calculTva2(1000) . '<br>';
```

```

//----- Déclaration
function bonjour ($qui) 2
{
    return "Bonjour $qui <br />\n";
}
//----- Exécution
echo bonjour ("Pierre") ; 1

```

52

## Créons une fonction avec un argument pour faire tous les calculs possibles

Dans notre précédente fonction utilisateur, nous pouvions calculer la TVA de nimporte quel nombre mais pas avec nimporte quel taux de TVA.

Nous allons améliorer notre fonction afin de pouvoir effectuer tous les calculs sur tous les nombres et sur tous les taux de TVA

Nous allons avoir besoin de deux arguments entrants :

entrainement.php

```

1 | <?php
2 | // déclaration d'une fonction
3 | function calculTva3($nombre, $taux)
4 |
5 |     return $nombre*$taux;
6 |
7 |
8 | // exécution de notre fonction
9 | echo "500 € avec un taux de 5.5 % font : " . calculTva3(500, 1.055) . '<br>';
10 | echo "1000 € avec un taux de 20 % font : " . calculTva3(1000, 1.2) . '<br>';

```

### Résultat

527.5

1200

### Explications

Cette fois-ci, dans les parenthèses de notre déclaration de fonction, nous avons prévu deux variables nommées \$nombre et \$taux. Ce qui implique qu'il faudra transmettre 2 informations au moment de l'exécution.

Le calcul a également changé, c'est \$nombre\*\$taux. Le nombre et le taux peuvent changer à tout moment.

Dans notre cas nous réalisons une première exécution :

```
1 | echo calculTva3(500) . '<br>';
```

Ce qui donne comme calcul :  $500 * 1.055 = 527.5$  €

Et, une deuxième exécution :

```
1 | echo calculTva3(1000) . '<br>';
```

Ce qui donne comme calcul :  $1000 * 1.2 = 1200$  €

La fonction a été renommée en calculTva3 pour éviter tout conflit avec la 1ère fonction calculTva et calculTva2 si vous l'aviez gardée présente dans

le même fichier.

Au moment de la déclaration, lorsqu'une fonction est destinée à recevoir 2 arguments, vous devrez obligatoirement lui envoyer les 2 arguments au moment de l'exécution, ne l'oubliez pas !

Dans une fonction nous pouvons déclarer autant de réceptions d'argument que l'on souhaite à condition de les transmettre au moment de l'exécution. Si le nombre d'arguments est conséquent, il est préférable de passer par une variable array.

53

## Argument facultatif par défaut

Nous l'avons dit, si votre fonction est destinée à recevoir 1 argument c'est qu'il faut lui envoyer obligatoirement 1 argument, si elle est destinée à recevoir 2 arguments, il faut lui en envoyer 2 et ainsi de suite...

Toutefois, il est possible que votre fonction attende 2 arguments mais en déclarant l'un d'entre eux comme étant facultatif !

Pour rendre un argument facultatif, il faut absolument lui déclarer une valeur par défaut.

Reprenons notre calcul de TVA :

```
entraînement.php

1 | <?php
2 | // déclaration d'une fonction
3 | function calculTva4($nombre, $taux = 1.2)
4 | {
5 |     return $nombre*$taux;
6 | }
7 |
8 | // exécution de notre fonction
9 | echo calculTva4(1500) . '<br>'; // exécution de la fonction avec 1 seul argument
10 | echo calculTva4(800, 1.07) . '<br>'; // exécution de la fonction avec 2 arguments
```

### Résultat

1800

856

### Explications

La première fois la fonction est exécutée avec un seul argument. C'est donc le taux de TVA par défaut 1.2 qui est pris en compte.

La seconde fois la fonction est exécutée avec deux arguments. et le taux 1.07 vient remplacer le taux par défaut (1.2) qui ne sera pas appliqué en présence d'un argument choisi.

On peut donc dire que nous avons une fonction destinée à recevoir 2 arguments, mais qu'il seul peut suffire.

Si un seul argument est envoyé, le taux par défaut est appliqué soit 1.2 :

```
1 | function calculTva4($nombre, $taux = 1.2)
```

En présence du 2e argument (lors de l'exécution) il est prioritaire et le taux par défaut (1.2) n'est pas appliqué.

```
1 | calculTva4(800, 1.07)
```

54

## Opérateurs Arithmétiques

Puisque nous parlons de calcul de TVA et de multiplications, voici ci-dessous quelques opérateurs arithmétiques

Derrière ce mot impressionnant se cache simplement des additions, soustractions, multiplications et divisions que des élèves de ce2 pourraient faire :).

C'est souvent comme ça en informatique, sont utilisés "certains mots qui font peur" ou qui font croire qu'il faut bac+5 pour les comprendre et les maîtriser alors qu'avec des explications correctes il n'en est rien, du moment que c'est bien expliqué.

### entraînement.php

```
1 | <?php
2 | $a = 10 ; $b = 2 ;
3 | echo $a + $b . '<br>'; // affiche 12 (addition 10 + 2)
4 |
5 | echo $a - $b . '<br>'; // affiche 8 (soustraction 10 - 2)
6 |
7 | echo $a * $b . '<br>'; // affiche 20 (multiplication 10 * 2)
8 |
9 | echo $a / $b . '<br>'; // affiche 5 (division 10 / 2)
10|
11| echo $a % $b . '<br>'; // affiche 0 (reste de la division 10 / 2)
```

### Résultat

```
12
8
20
5
0
```

Jusque là, rien de bien difficile !

Nous pouvons aussi réaliser ces opérations en affectant une variable (dans la même exécution) :

### entraînement.php

```
1 | <?php
2 | $a += $b ; // équivaut à $a = $a + $b (ici $a vaut 12, voir ci-dessus)...10+2
3 | $a -= $b ; // équivaut à $a = $a - $b (ici $a vaut 12)...12-2
```

### Résultat

```
12
10
```

Explications :

`$a += $b` est pareil que `$a = $a + $b`

Cette notation de code peut être pratique lors d'un calcul sur un panier de site e-commerce.

55

## Exercice fonction

Voici une simple fonction permettant de formuler une phrase au sujet de la météo :

### entraînement.php

```
1 | <?php
2 | meteo("hiver", 5); // il est possible d'exécuter une fonction avant qu'elle soit déclarée dans le code.
3 | function meteo($saison, $temperature) // 2 arguments attendus
4 | {
5 |   echo "Nous sommes en $saison et il fait $temperature degrés<br>";
6 | }
```

Comme vous pouvez le constater, cette fonction attend 2 arguments, c'est la raison pour laquelle nous lui envoyons 2 arguments "hiver" et "5".

## Résultat

Nous sommes en hiver et il fait 5 degrés

Imaginons que nous envoyons le chiffre 1 en 2e argument, cela afficherait "1 degrés" avec une faute de Français pour le "s", du coup notre exercice va être de gérer le pluriel et le singulier en fonction du nombre envoyé.

Exercice : pourriez-vous gérer le cas de la lettre S sur le mot degré en fonction du nombre envoyé ?

Pour 1 degré, cela doit s'écrire : 1 degré et non pas 1-degrés

Pour 2 degrés, cela doit s'écrire : 2 degrés et non pas 2-degré

*Ne regardez pas la correction si vous souhaitez jouer le jeu*

### entraînement.php

```
1 | <?php
2 | meteo("hiver", 1); // il est possible d'exécuter une fonction avant qu'elle soit déclarée dans le code.
3 | function meteo($saison, $temperature) // 2 arguments attendus
4 |
5 |     echo "Nous sommes en $saison et il fait $temperature";
6 |     if($temperature > 1 || $temperature < -1) echo " degrés<br>";
7 |     else echo " degré<br>";
8 | }
```

## Résultat

Nous sommes en hiver et il fait 1 degré

*Nous pouvons remarquer le résultat sans la présence du S au mot degré*

Vous pouvez constater que lorsque nous écrivons un IF / ELSE avec une seule instruction à l'intérieur nous pouvons le mettre sur la même ligne sans accolades.

Cependant dans la majorité des cas, il est préférable de garder les accolades.

Attention, comme vous le savez le nombre d'arguments qu'attend une fonction est important, mais l'ordre des arguments l'est tout autant !

Voici un mauvais exemple avec l'inversion de l'ordre des arguments :

### entraînement.php

```
1 | <?php
2 | meteo(30, "été"); // il est possible d'exécuter une fonction avant qu'elle soit déclarée dans le code.
3 | function meteo($saison, $temperature) // 2 arguments attendus
4 |
5 |     echo "Nous sommes en $saison et il fait $temperature";
6 |     if($temperature > 1 || $temperature < -1) echo " degrés<br>";
7 |     else echo " degré<br>";
8 | }
```

Cet exemple affichera :

## Résultat

Nous sommes en 1 et il fait été degré

Nous pouvons donc constater que le nombre et l'ordre des arguments sont vraiment importants !

Ce chapitre risque d'être abstrait sur le coup mais très important et indispensable lors de la création d'un site web.  
je vous recommande donc de suivre les explications. Concentrez-vous :)

### Qu'est-ce qu'un espace local ?

Un espace local est le code déclaré à l'intérieur d'une fonction.

Exemple :

```
entraînement.php
```

```
1 | <?php
2 | function affichageVille()
3 | {
4 |     $ville = "Lyon"; // variable locale.
5 | }
```

On dit que la variable `$ville` est déclarée dans un espace local.

### Qu'est-ce qu'un espace global ?

Un espace global est le code déclaré à l'extérieur d'une fonction (partout ailleurs).

Exemple :

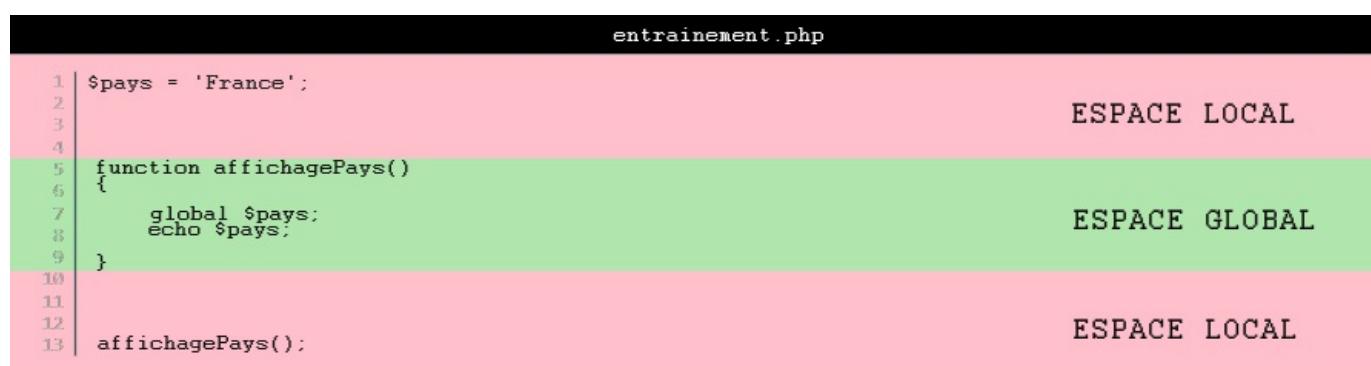
```
entraînement.php
```

```
1 | <?php
2 | function affichageVille()
3 | {
4 |     // ...
5 | }
6 |
7 | $ville2 = "Marseille"
```

On dit que la variable `$ville2` est déclarée dans l'espace global (espace par défaut).

### Quelle est la différence entre 1 espace local et 1 espace global ?

Mieux qu'une succession d'explications, voici une image qui représente parfaitement la chose :



```
echo "bonjour 1<br />";
```

ESPACE GLOBAL

```
function joursemaine()
{
    $jour = "lundi";
    return $jour;
    echo 'ALLO';
}
```

ESPACE LOCAL

```
echo $jour; // /!\$recup = joursemaine();echo $recup . '<br />';
```

ESPACE GLOBAL

Autre Exemple :

```
$pays = 'France';
```

```
function affichage_pays()
{
    global $pays;
    echo $pays; // /!\"}
```

ESPACE LOCAL

```
affichage_pays();
```

ESPACE GLOBAL

### Problématique avec l'espace local

entraînement.php

```
1 <?php
2 function affichageVille()
3 {
4     $ville = "Lyon"; // variable locale.
5 }
6 affichageVille();
7 echo $ville; // /!\ ne fonctionne pas car cette variable n'est connu qu'à l'intérieur de la fonction
```

Ce code ne fonctionnera pas, voici ce qu'il affichera :

#### Résultat

Notice: Undefined variable: ville in C:\wamp\www\php\entraînement.php on line ...

#### Quelques Explications :

Ligne 1 : Nous déclarons une fonction affichageVille().

Ligne 3 : Nous déclarons une variable locale. variable locale = variable déclarée à l'intérieur d'une fonction.

Ligne 5 : Exécution de la fonction... (dans le vide)

Ligne 6 : Tentative d'affichage de la variable locale \$ville déclarée à l'intérieur de la fonction.

Nous obtenons une erreur car en dehors de la fonction (au dessus et en dessous) nous sommes dans un espace global (c'est l'espace par défaut), alors qu'à l'intérieur de la fonction nous sommes dans un espace local.

Les informations de l'espace local ne sont pas connues dans l'espace global et vice-versa.

### Comment véhiculer des informations de l'espace local vers l'espace global ?

Pour faire passer des informations déclarées dans l'espace local vers l'espace global, voici le code que nous pourrions mettre en oeuvre :

```
entraînement.php
```

```
1 | <?php
2 | function affichageVille()
3 | {
4 |     $ville = "Lyon"; // variable locale.
5 |     return $ville; // une fonction peut retourner qqchose (à ce moment là on quitte la fonction)
6 |     echo 'test'; // !\ cette ligne ne s'affichera pas car il y a un return avant.
7 |
8 |     $villeReçu = affichageVille();
9 |     echo $villeReçu;
```

#### Résultat

Lyon

#### Quelques Explications :

Ligne 1 : Nous déclarons une fonction `affichageVille()`.

Ligne 3 : Nous déclarons une variable locale. **variable locale = variable déclarée à l'intérieur d'une fonction.**

Ligne 4 : `Return` permettra lors de l'exécution de la fonction, de retourner le contenu de la variable `$ville` (pour en avoir connaissance à l'extérieur de la fonction).

Ligne 5 : Ajout d'un `echo` qui n'a aucun rapport, juste pour montrer qu'après un `return` nous quittons la fonction et la suite du code n'est pas exécutée. Nous ne verrons pas le mot "test" s'afficher dans le résultat.

Ligne 7 : Nous récupérons l'exécution de la fonction dans une variable, la variable `$villeReçu` contiendra le contenu du `return` de la fonction `affichageVile()` c'est à dire "Lyon".

Ligne 8 : Nous affichons le contenu de la variable `$villeReçu` soit "Lyon".

En conclusion, nous pouvons donc dire que l'utilisation du mot-clé `return` permettra de faire passer des informations déclarées en local dans l'espace global.

Lorsque vous créez une fonction perso, pensez à l'utilisation du mot-clé `return` pour préciser ce que votre fonction devra renvoyer !

### Comment véhiculer des informations de l'espace global vers l'espace local ?

De la même manière, si les informations déclarées en local ne sont pas connues en global, nous retrouverons la même problématique dans l'autre sens (déclaration de variable en global et utilisation souhaitée en local).

#### Problématique avec l'espace global

```
entraînement.php
```

```
1 | <?php
2 | $pays = 'France';
3 | function affichagePays()
4 | {
5 |     echo $pays; // générera une erreur PHP.
6 | }
7 | affichagePays();
```

#### Résultat

Notice: Undefined variable: pays in C:\wamp\www\php\entraînement.php on line ...

#### Quelques Explications :

Ligne 1 : Nous déclarons une variable `$pays` dans l'espace global (et oui nous sommes bien dans l'espace global puisque nous sommes à l'extérieur d'une fonction).

Ligne 2 : Déclaration d'une fonction utilisateur.

Ligne 4 : Affichage de la variable \$pays.

Ligne 6 : Exécution de notre fonction affichagePays().

Suite à l'exécution de notre fonction affichagePays(), nous aurons une erreur sur la variable \$pays qui apparaîtra pour l'interpréteur comme étant indéfinie (undefined).

Cette erreur est normale dans la mesure où la variable est déclarée en global et on tente de l'afficher en local.

Pour faire passer des informations déclarées dans l'espace global vers un espace local (une fonction), voici le code que nous pourrions mettre en oeuvre :

```
entraînement.php
```

```
1 | <?php
2 | $pays = 'France';
3 | function affichagePays()
4 | {
5 |     global $pays; // importe la variable pays de l'espace global
6 |     echo $pays;
7 | }
8 | affichagePays();
```

## Résultat

France

### Quelques Explications :

Ligne 4 : Le mot-clé `global` permet d'importer une variable de l'espace global vers l'espace local, en l'occurrence la variable `$pays`.

Nous aurions également pu envoyer `$pays` en argument à notre fonction.

Retenez bien que vous aurez besoin du mot-clé `global` lorsque vous aurez besoin de connaître une variable dans une fonction.

Exemple : la connexion vers une base de données peut être représentée par une variable globale, et les requêtes SQL (permettant de récupérer des données) pourraient se trouver dans des fonctions locales, du coup il faudrait connaître la connexion globale pour effectuer des requêtes en local.

En conclusion, `global` permet d'importer des informations globales dans un espace local. `return` permet de sortir des informations locales dans l'espace global.

## Les boucles

57

### Boucle (structures itératives)

Les boucles (structures itératives) est un moyen de faire répéter une portion de code plusieurs fois.

Nous appelons structure itérative (ou récursive), la structure qui permet d'exécuter plusieurs fois les mêmes instructions. Elle permet d'exécuter "en boucle" un bloc d'instructions.

En 1 mot : Une boucle c'est une répétition !

58

### Comprendre le principe des boucles

Là aussi, il s'agit d'un des chapitres les plus importants de notre apprentissage puisque tous les sites dynamiques du monde incluent forcément un système de boucle à un moment donné.

Contexte : cela sera très utile au moment d'afficher une liste de produits sur un site ecommerce.

Imaginez que le code html suivant permet d'afficher l'un de vos produits :

```
entrainement.php  
1 <?php  
2 <div class="produit">  
3   <h2>Tshirt Gris</h2>  
4     
5   <p>30 €</p>  
6   <form method="post" action=""><input type="submit" name="ajout_panier" value="ajout au panier"></form>  
7 </div>
```

PS : Ce code représente un exemple, il n'est pas à la lettre pour construire votre page produit ;).



Résultat (exemple)

Si vous aviez besoin d'afficher 50 produits par page, pensez-vous que vous copier/collerez votre code html 50 fois ?

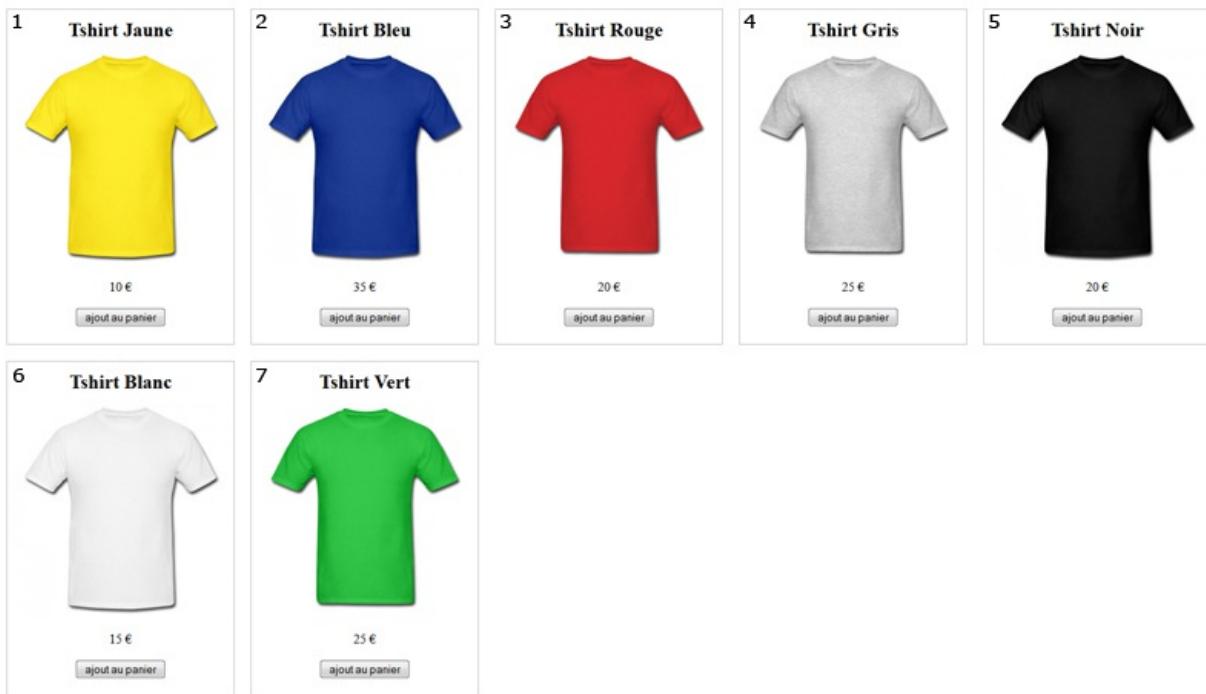
NON ! bien entendu ce n'est pas comme cela que ça fonctionne pour plusieurs raisons.

Avec les boucles, nous allons déclarer 1 seule fois le code et le répéter 50 fois automatiquement (50 fois ou autant de fois que l'on souhaite).

Résultat

(exemple)

## Catalogue



Bien entendu, dans la version finale les produits seront différents les uns des autres mais le principe est là.

Vous comprenez le principe et l'intérêt des boucles ?

Nous allons donc voir en pratique avec le code comment nous pourrions enclencher ce phénomène de répétition.

Soyons clair, les boucles c'est un sujet capital, dire "je ne comprends pas" ou "ça ne m'intéresse pas, j'attends le concret" revient à dire "je ne programmerais pas de site web dans ma vie".

4 techniques de boucles (répétition) : while, do-while, for, foreach.

59

## La boucle While

entraînement.php

```
1 <?php
2 $i = 0; // valeur de départ
3 while ($i<5) // tant que $i est inférieur à 5
4 {
5   echo "$i "; // affichage
6   $i++; // Ceci est une forme contractée de : $i = $i+1; l'incrémentation du "compteur" est effectuée à chaque tour de boucle.
7 }
```

Quelques Explications :

### Tour de boucle 0

Ligne 2 : nous déclarons une variable \$i et lui affectons la valeur 0

Nous utilisons en général une variable que l'on nomme \$i pour INT (integer) mais nous aurions pu la nommer complètement différemment, il n'y a pas d'obligations, juste des usages.

Ligne 3 : While (notre boucle) peut être traduit par "Tant que", ce qui donne tant que \$i est inférieur à 5, nous rentrons dans la boucle.

Est-ce que \$i est bien inférieur à 5 ? Oui puisque \$i est égal à 0. (c'est ce que nous avons déclaré sur la ligne d'en haut).

Ligne 5 : Nous affichons la valeur contenue dans la variable \$i.

Quelle est cette valeur ? 0 ! et oui puisque nous avons mis \$i = 0; il n'y a pas de raison que cette variable nous donne une autre valeur que 0.

Cela affichera donc "0 /", nous avons mis un slash de séparation mais nous aurions pu choisir un autre caractère.

Ensuite, si nous devons enclencher ce fameux phénomène de répétition (la boucle) il nous faut un système qui puisse compter et progresser dans le nombre de tours que l'on effectuera dans notre boucle.

D'accord pour répéter une portion de code mais combien de fois ? pas à l'infini quand même !

Ligne 6 : Nous utilisons \$i++; (forme contractée de \$i = \$i + 1) afin de rendre cette variable évolutive.

\$i est la variable qui permettra d'entrer dans la boucle afin d'effectuer des tours.

C'est à dire que lorsque cette ligne s'exécutera, \$i passera de 0 à 1. puis ensuite de 1 à 2. et ainsi de suite...

Au moment où la ligne 6 nous permet d'incrémenter notre variable \$i, l'interpréteur remonte automatiquement (dans le cas d'une boucle comme ici) sur la ligne 3 pour tenter d'exécuter à nouveau son contenu.

Ca veut dire que nous remontons dans le code et revenons sur la ligne n°3 !

#### Tour de boucle 1

Ligne 3 : Par conséquent, l'interpréteur test à nouveau la condition d'entrée dans la boucle : est-ce que \$i est bien inférieur à 5 ?

Oui, encore 1 fois ! puisque cette fois-ci \$i est passé à 1 (grâce au \$i++; vu précédemment).

1 est bien inférieur à 5, nous rentrons à nouveau dans la boucle while.

Ligne 5 : Nous affichons à nouveau la valeur contenue dans la variable \$i.

Cela affichera "1 /".

Ligne 6 : Nous repassons sur la ligne \$i++; (forme contractée de \$i = \$i + 1) afin de passer \$i à 2.

L'interpréteur remonte à nouveau dans le code (ligne 3 précisément) et nous observons à nouveau le cycle suivant :

#### Tour de boucle 2

Ligne 3 : la condition d'entrée renvoie TRUE (vrai) et nous entrons à nouveau dans la boucle while (puisque \$i 2 < 5).

Ligne 5 : Nous affichons "2 /" (instruction d'affichage echo).

Ligne 6 : Nous incrémentons \$i à 3 (2+1)

#### Tour de boucle 3

Ligne 3 : la condition d'entrée renvoie TRUE (vrai) et nous entrons à nouveau dans la boucle while (puisque \$i 3 < 5).

Ligne 5 : Nous affichons "3 /" (instruction d'affichage echo).

Ligne 6 : Nous incrémentons \$i à 4 (3+1)

#### Tour de boucle 4

Ligne 3 : la condition d'entrée renvoie TRUE (vrai) et nous entrons à nouveau dans la boucle while (puisque \$i 4 < 5).

Ligne 5 : Nous affichons "4 /" (instruction d'affichage echo).

Ligne 6 : Nous incrémentons \$i à 5 (3+1)

Ensuite l'interpréteur tente de retourner ligne 3 pour exécuter à nouveau la boucle mais ça n'est pas possible dans la mesure où \$i est passé à 5 et 5 n'est pas inférieur à 5.

Résultat Final

0 / 1 / 2 / 3 / 4

De 0 à 4, ça fais bien 5 tours au total !

Reprenons les explications une à une :

```
1 | $i = 0;
```

C'est ce qui représente notre valeur de départ.

```
1 | while
```

Nous pouvons remplacer "while" par "tant que".

```
1 | ($i<5)
```

C'est ce qui représente notre condition d'entrée dans la boucle (\$i doit forcément être inférieur à 5 pour que l'on puisse rentrer).

Nous avons mis 5 pour réaliser 5 tours mais nous aurions pu mettre 10, 20, 100 ou le nombre que vous souhaitez.

```
1 | while($i<5)
```

Traduction : Tant que \$i est inférieur à 5 nous rentrons dans la boucle.

Notre condition d'entrée c'est ce qui permet de ne pas avoir une boucle infinie et de pouvoir s'arrêter à un moment donné (après 5 tours).

```
1 | { }
```

Les accolades permettent de fixer le périmètre de la boucle

```
1 | echo "$i /";
```

Affiche le chiffre contenu dans la variable \$i suivi d'un slash "/"

```
1 | $i++;
```

\$i est incrémenté (indispensable pour faire tourner la boucle), c'est l'équivalent de \$i = \$i + 1; et nous en avons besoin pour enclencher "le tour suivant" à chaque fois.

```
1 | <?php
2 | $i = 0;
3 | while ($i<5)
4 | {
5 |   echo "$i /";
6 |   $i++;
7 | }
```

Ce cycle est répété 5 fois (5 tours de boucle).



60

## La boucle DoWhile

Concernant la boucle do-while, il s'agit exactement du même principe que la boucle while, la seule différence réside dans le fait que la condition d'entrée est testée à la fin plutôt qu'au début.

On entre d'abord et ensuite on vérifie si l'on peut continuer à boucler.

```
entraînement.php
```

```
1 | $i = 0;
```

```
2 do {  
3     echo $i;  
4 } while ($i > 0);
```

Cette boucle ne fera qu'un seul tour car \$i n'est pas supérieur à 0.

Nous l'utiliserons plus rarement, plutôt dans des cas spécifique.

*Vous pouvez la garder à l'esprit pour savoir qu'elle existe si on vous en parle ou que vous la croisez dans un code mais vous n'êtes pas obligé de la connaître par cœur.*

61

## La boucle For

Tout comme la boucle while, l'objectif de cette boucle est aussi d'effectuer une répétition de code.

La structure est légèrement différente au niveau de la syntaxe :

```
entraînement.php  
1 for($j = 0; $j <= 15; $j++)  
2 {  
3     echo $j;  
4 }
```

Quelques Explications :

```
1 for($j = 0; $j <= 15; $j++)
```

Voici à quoi correspond notre boucle :

\$j= 0; - Valeur de départ c'est l'initialisation qui indique la variable qui servira de compteur ainsi que sa valeur initiale.  
\$j <= 15; - Condition d'entrée condition pour continuer les boucles. L'instruction s'achève dès que la condition renvoie faux.  
\$j++ - Evolution : évolution du compteur à chaque tour de boucle (généralement une incrémentation ou décrémentation).

```
1 echo $j;
```

L'instruction echo permet d'afficher la valeur contenue dans la variable \$j.

### Résultat

0123456789101112131415

*La boucle fait 16 tours au total, de 0 à 15.*

Exercice : serait-il possible de passer les nombres à la ligne les uns à la suite des autres ?

Voici le résultat attendu :

### Résultat

0  
1  
2  
3  
4

5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15

A vous de jouer ! et ne regardez pas la correction en dessous avant d'y être parvenu, ça vous entraînera !

Correction :

```
entraînement.php
?
1 | for($j = 0; $j <= 15; $j++)
2 | {
3 |   echo "<p>$j</p>";
4 | }
```

Il suffisait de mettre un `<br>` ou de faire appel aux balises `<p> </p>` pour faire succéder à la ligne les différents chiffres écrits par notre boucle.

En effet, avec l'écriture des balise `<p> </p>` ou `<br>` une seule fois, cela se répétera autant de fois que demandé puisque nous sommes dans les accolades de la boucle et donc dans l'effet de répétition.

Pensez à regarder le code-source généré ! **[Ctrl+U]** ou clic droit -> code source de la page

Autre exercice, maintenant que vous avez vu que l'on pouvait mélanger du code php et html dans le cadre des boucles, pourriez-vous proposer un champ date de naissance en prévoyant les 31 jours existants dans notre calendrier :

Voici le résultat attendu :

Résultat attendu Quel est votre jour de naissance ?

L'exercice est donc de parvenir à ce résultat sans faire de copier/coller. A vous de jouer !

Pour vous aider (si vous aviez un doute sur le code html), voici les lignes permettant d'obtenir une liste de choix déroulante :

```
entraînement.php
?
1 | <select>
2 |   <option>1</option>
3 |   <option>2</option>
4 | </select>
5 | 
```

L'objectif est donc d'avoir 1 seul select et 31 options (sans les écrire manuellement, sinon je vous en demande 10 000, on verra si vous faites toujours des copiers/coller =D).

Voici la correction :

```
entraînement.php
?
```

```

1 echo '<select>';
2 for($j = 1; $j <= 31; $j++)
3 {
4     echo "<option>$j</option>";
5 }
6 echo '</select>';

```

Nous voulions qu'il seul `<select>` mais 31 options, c'est la raison pour laquelle le `select` se trouve en dehors des accolades de la boucle (nous ne souhaitons pas qu'il soit répété), les options se trouvent dans les accolades de notre boucle (pour qu'elles bénéficient de l'effet de répétition).

Nous commençons la boucle à 1, car personne est né le 0 ! nous nous arrêtons à 31 (car personne est né le 32). Il y aura donc 31 tours et 31 options.

Nous aurions pu créer cette boucle dans l'autre sens (de 31 à 1 et non de 1 à 31), on parle alors de décrémentation :

### entraînement.php

```

1 echo '<select>';
2 for($j = 31; $j >= 1; $j--)
3 {
4     echo "<option>$j</option>";
5 }
6 echo '</select>';

```

Nous commençons notre compteur (`$j`) à 31,

Tant que nous sommes supérieur ou égal à 1 nous rentrons dans la boucle

`$j--` permet de baisser le nombre à chaque tour afin de faire progresser la boucle

Résultat attendu Quel est votre jour de naissance ?

31

62

## La boucle Foreach

La boucle foreach sera vue dans le chapitre dédié aux arrays.

(*Elle permet d'afficher les valeurs d'un tableau de données array*).

63

## Choix : boucle while / for / foreach / dowhile

Dans tous les cas, l'objectif d'une boucle est de répéter une portion de code.

Selon la situation, il peut être plus aisés d'utiliser la structure de l'une d'entre elles.

Nous y reviendrons plus tard dans le cours et à force de les utiliser, nous serons quelle structure est la plus adaptée dans les différents cas.

64

## Boucle imbriquée (Double boucle)

Voici une boucle permettant d'afficher des données dans une table (tableau) html

### entraînement.php

```

1 echo "<table border='1'><tr>";
2 for($i = 0; $i < 9; $i++)

```

```

3 | {
4 |     echo "<td> $i </td>";
5 | }
6 | echo "</tr></table>";

```

#### Résultat

0	1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---	---

Exercice : essayez de reproduire ces données :

#### Résultat attendu

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

Voici la correction possible :

#### entrainement.php

```

1 | echo '<table border="1">';
2 | for($ligne = 0; $ligne < 10; $ligne++) //ligne est à zéro
3 | {
4 |     echo '<tr>';
5 |     for($cellule = 0; $cellule < 10; $cellule++) // tant que ligne est à zéro, cellule s'incrémente 10 fois - ligne est à 1, cellule s'incrémente
6 |     10 fois - ligne est à 2, cellule s'incrémente 10 fois, etc...
7 |     {
8 |         echo '<td>' . (10 * $ligne + $cellule) . '</td>; // 10 * 0 +0 , 10* 0 + 1, 10* 0 + 2, etc... --- 10* 1+0, 10* 1 + 1, 10* 1+2, etc. Permet
9 |     d'avoir les multiples de 10.
10 |
11 |     echo '</tr>';
} |     echo '</table>';

```

#### Quelques Explications :

Tour de boucle \$ligne 0, \$cellule 0 :

\$ligne est à 0. (<10) nous rentrons dans la première boucle.

Nous faisons le 1er <tr>

\$cellule est à 0 (<10) nous rentrons dans la deuxième boucle.

Nous faisons le 1er <td> avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 0 + 0 = 0$ .

0

Tour de boucle \$ligne 0, \$cellule 1 :

\$ligne reste à 0.

Très important pour la compréhension : nous ne rerentrons pas dans la première boucle mais restons "coincé" dans la deuxième (le temps que ses tours soient terminés).

\$cellule passe à 1 (<10) nous rentrons à nouveau dans la deuxième boucle.

Nous faisons le 2ème <td> avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 0 + 1 = 1$ .

0	1
---	---

Tour de boucle \$ligne 0, \$cellule 2 :

\$ligne reste à 0.

Nous ne rentrons pas dans la première boucle mais restons toujours "coincé" dans la deuxième (le temps que ses tours soient terminés).

\$cellule passe à 2 (<10) nous rentrons à nouveau dans la deuxième boucle.

Nous faisons le 3ème <td> avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 0 + 2 = 2$ .

0	1	2
---	---	---

Et ainsi de suite jusqu'à ce que \$cellule atteigne 10.

0	1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	---	----

C'est là que tout se joue.

Nous rentrons à nouveau dans la 1ère boucle et \$ligne passe à 1.

\$cellule est quant à elle remise à 0 puisque nous repassons dans les lignes qui initialisent la variable.

Tour de boucle \$ligne 1, \$cellule 0 :

\$ligne est à 1. (<10) nous rentrons dans la première boucle.

Nous faisons le 2ème <tr> (ce qui a pour effet de passer à la ligne)

\$cellule est à 0 (<10) nous rentrons dans la deuxième boucle.

Nous faisons le 1er <td> de cette nouvelle ligne (et 11e au total, puisqu'il s'agit du 11e tour) avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 1 + 0 = 11$ .

0	1	2	3	4	5	6	7	8	9	10
11										

Tour de boucle \$ligne 1, \$cellule 1 :

\$ligne reste à 1.(<10)

nous ne rerentrons pas dans la première boucle mais restons "coincé" dans la deuxième (le temps que ses tours soient terminés).

\$cellule passe à 1 (<10) nous rentrons à nouveau dans la deuxième boucle.

Nous faisons le 2ème <td> de cette nouvelle ligne avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 1 + 1 = 12$ .

0	1	2	3	4	5	6	7	8	9	10
11	12									

Tour de boucle \$ligne 1, \$cellule 2 :

\$ligne reste à 1. (<10)

Nous ne rerentrons pas dans la première boucle mais restons "coincé" dans la deuxième (le temps que ses tours soient terminés).

\$cellule passe à 2 (<10)

Nous rentrons à nouveau dans la deuxième boucle.

Nous faisons le 3ème <td> de cette nouvelle ligne avec le calcul suivant  $10 * \$ligne + \$cellule$ , ce qui donne  $10 * 1 + 2 = 13$ .

0	1	2	3	4	5	6	7	8	9	10
11	12	13								

Et ainsi de suite... voici les tours de boucles précis :

\$ligne 0 , \$cellule 0\$ligne 0 , \$cellule 1\$ligne 0 , \$cellule 2\$ligne 0 , \$cellule 3\$ligne 0 , \$cellule 4\$ligne 0 , \$cellule 5\$ligne 0 , \$cellule 6\$ligne 0 , \$cellule 7\$ligne 0 , \$cellule 8\$ligne 0 , \$cellule 9

\$ligne 1 , \$cellule 0\$ligne 1 , \$cellule 1\$ligne 1 , \$cellule 2\$ligne 1 , \$cellule 3\$ligne 1 , \$cellule 4\$ligne 1 , \$cellule 5\$ligne 1 , \$cellule 6\$ligne 1 , \$cellule 7\$ligne 1 , \$cellule 8\$ligne 1 , \$cellule 9

\$ligne 2 , \$cellule 0\$ligne 2 , \$cellule 1\$ligne 2 , \$cellule 2\$ligne 2 , \$cellule 3\$ligne 2 , \$cellule 4\$ligne 2 , \$cellule 5\$ligne 2 , \$cellule 6\$ligne 2 , \$cellule 7\$ligne 2 , \$cellule 8\$ligne 2 , \$cellule 9

\$ligne 3 , \$cellule 0\$ligne 3 , \$cellule 1\$ligne 3 , \$cellule 2\$ligne 3 , \$cellule 3\$ligne 3 , \$cellule 4\$ligne 3 , \$cellule 5\$ligne 3 , \$cellule 6\$ligne 3 , \$cellule 7\$ligne 3 , \$cellule 8\$ligne 3 , \$cellule 9

\$ligne 4 , \$cellule 0\$ligne 4 , \$cellule 1\$ligne 4 , \$cellule 2\$ligne 4 , \$cellule 3\$ligne 4 , \$cellule 4\$ligne 4 , \$cellule 5\$ligne 4 , \$cellule 6\$ligne 4 , \$cellule 7\$ligne 4 , \$cellule 8\$ligne 4 , \$cellule 9

\$ligne 5 , \$cellule 0\$ligne 5 , \$cellule 1\$ligne 5 , \$cellule 2\$ligne 5 , \$cellule 3\$ligne 5 , \$cellule 4\$ligne 5 , \$cellule 5\$ligne 5 , \$cellule 6\$ligne 5 , \$cellule 7\$ligne 5 , \$cellule 8\$ligne 5 , \$cellule 9

\$ligne 6 , \$cellule 0\$ligne 6 , \$cellule 1\$ligne 6 , \$cellule 2\$ligne 6 , \$cellule 3\$ligne 6 , \$cellule 4\$ligne 6 , \$cellule 5\$ligne 6 , \$cellule 6\$ligne 6 , \$cellule 7\$ligne 6 , \$cellule 8\$ligne 6 , \$cellule 9

\$ligne 7 , \$cellule 0\$ligne 7 , \$cellule 1\$ligne 7 , \$cellule 2\$ligne 7 , \$cellule 3\$ligne 7 , \$cellule 4\$ligne 7 , \$cellule 5\$ligne 7 , \$cellule 6\$ligne 7 , \$cellule 7\$ligne 7 , \$cellule 8\$ligne 7 , \$cellule 9

\$ligne 8 , \$cellule 0\$ligne 8 , \$cellule 1\$ligne 8 , \$cellule 2\$ligne 8 , \$cellule 3\$ligne 8 , \$cellule 4\$ligne 8 , \$cellule 5\$ligne 8 , \$cellule 6\$ligne 8 , \$cellule 7\$ligne 8 , \$cellule 8\$ligne 8 , \$cellule 9

\$ligne 9 , \$cellule 0\$ligne 9 , \$cellule 1\$ligne 9 , \$cellule 2\$ligne 9 , \$cellule 3\$ligne 9 , \$cellule 4\$ligne 9 , \$cellule 5\$ligne 9 , \$cellule 6\$ligne 9 , \$cellule 7\$ligne 9 , \$cellule 8\$ligne 9 , \$cellule 9

En conclusion, lorsque nous faisons un tour dans la boucle avec \$ligne, cela en entraîne 10 dans la boucle avec \$cellule.

Au total la première boucle (avec \$ligne) tourne 10 fois et la deuxième boucle (avec \$cellule) tourne 100 fois.

Voici une autre correction possible :

```
entraînement.php
```

```
1 | $z = 0;
2 | echo '<table border="1">';
3 | for($ligne = 0; $ligne < 10; $ligne++) //ligne est à zéro
4 |
5 |     echo '<tr>';
6 |     for($cellule = 0; $cellule < 10; $cellule++)
7 |     {
8 |         echo '<td>' . $z . '</td>';
9 |         $z++;
10|     }
11|     echo '</tr>';
12|
13| echo '</table>';
```

Solution avec une seule boucle PHP et l'utilisation du modulo (code html non valide) :

### entraînement.php

```
1 echo '<table border="1"><tr>';
2 for($i = 0; $i < 100; $i++) //ligne est à zéro
3 {
4     if($i % 10 == 0)
5     {
6         echo '</tr>';
7     }
8     echo '<td>' . $i . '</td>';
9 }
10 echo '</table>';
```

Solution différente :

### entraînement.php

```
1 echo '<table border="1">';
2 $i = 0;
3 while($i <= 99)
4 {
5     echo '<tr>';
6     for($k = 0; $k <= 9 ; $k++)
7     {
8         echo "<td>$i</td>";
9         $i++;
10    }
11    echo '</tr>';
12 }
13 echo '</table>';
```

## Les inclusions de fichiers

65

## Les inclusions de fichiers

PHP permet d'inclure un fichier dans un autre. Nous appelons cela des inclusions de fichiers et c'est très pratique dans le cadre d'un site web !

Pour tester le prochain exemple, je vous recommande de créer un fichier que l'on nommera : **fichier.inc.php**

Inscrivons le contenu suivant :

### fichier.inc.php

```
1 <p>Voici le texte du fichier qui se nomme fichier.php<br>
2 Pour rappel, même si ce fichier à l'extension ".php". Nous ne sommes pas obligés d'écrire du code PHP à l'intérieur !</p>
3
```

Le mot ".inc" n'est pas obligatoire mais par convention il indique que ce fichier n'est pas une page à part entière mais plutôt un fichier qui sera inclus dans une page web.

Ensuite reprenons notre fichier de cours **entraînement.php**.

66

## Include

### entraînement.php

```
1 <?php
2 echo 'Nous sommes dans le fichier entraînement<br>';
3 include("fichier.inc.php");
4 echo 'Nous sommes toujours dans le fichier entraînement<br>';
5
```

Nous venons d'inclure notre fichier **fichier.php** à l'intérieur de notre fichier principal **entraînement.php**

## Résultat

Nous sommes dans le fichier entrainement

Voici le texte du fichier qui se nomme fichier.php

Pour rappel, même si ce fichier à l'extension ".php". Nous ne sommes pas obligés d'écrire du code php à l'intérieur !

Nous sommes toujours dans le fichier entrainement

Ce n'est pas plus compliqué, la fonction prédefinie `include` attend comme argument un nom de fichier valide pour pouvoir l'inclure !

Avouez que ce n'est pas la partie la plus difficile :p

67

## Include\_once

entrainement.php

```
1 | <?php
2 | echo 'Nous sommes dans le fichier entrainement<br>';
3 | include_once("fichier.inc.php");
4 | echo 'Nous sommes toujours dans le fichier entrainement<br>';
5 | 
```

Tout comme `include`, `include_once` permet d'inclure un fichier dans un autre.

Le "`_once`" permet de vérifier qu'il n'a pas déjà été inclue (il n'incluera pas 2 fois le même fichier).

Si le fichier n'a jamais été inclue, l'interpréteur l'inclue, sinon il ignore la ligne et ne l'inclue pas.

68

## Require

entrainement.php

```
1 | <?php
2 | echo 'Nous sommes dans le fichier entrainement<br>';
3 | require("fichier.inc.php");
4 | echo 'Nous sommes toujours dans le fichier entrainement<br>';
5 | 
```

Tout comme `include`, `require` permet d'inclure un fichier dans un autre.

69

## Require\_once

entrainement.php

```
1 | <?php
2 | echo 'Nous sommes dans le fichier entrainement<br>';
3 | require_once("fichier.inc.php");
4 | echo 'Nous sommes toujours dans le fichier entrainement<br>';
5 | 
```

Tout comme `require`, `require_once` permet d'inclure un fichier dans un autre.

Le "`_once`" permet de vérifier qu'il n'a pas déjà été inclue (il n'incluera pas 2 fois le même fichier).

Include est plutôt traduit par "inclue moi ce fichier" et Require par "fichier requis".

La seule différence réside dans le cas d'une erreur (nom de fichier incorrect) :

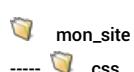
- **Include** fera une erreur et poursuivra l'exécution du code.
- **Require** fera une erreur et stopera l'exécution du code.
- **\_once** est juste présent pour s'assurer que le fichier soit inclus une seule fois dans le code.

Voici l'affichage classique d'un site web statique html et css :

Nom Du Site	Le Slogan du Site	Zone Haut Droite
Accueil	Accueil      Actualités      Contact	Colonne Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil
Titre niveau 2  Voici le texte de notre page d'accueil Voici le texte de notre page d'accueil		
Titre 3  		

© Copyright 2016

Voici l'aborescence classique d'un site web statique html et css :





Voici le code classique d'une page web html et css :

index.html - actualites.html - contact.html

Cette structure de page peut être reprise pour composer les autres rubriques.

Vous pouvez donc copier/coller le code de la page index.html pour faire actualites.html et contact.html.

Seul le texte changera d'une page à l'autre.

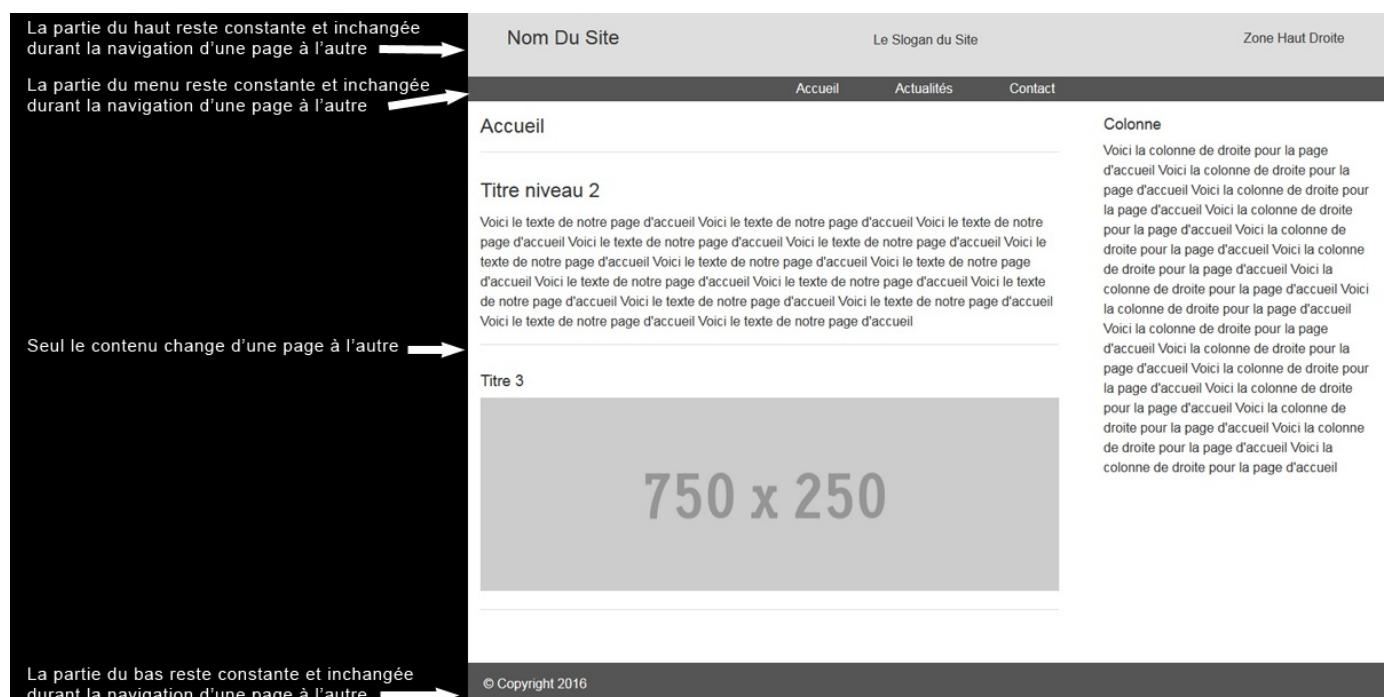
**Problématique : Si vous devez ajouter un logo sur votre site et que vous le faites sur la page index.html, cette modification ne sera pas répercutée sur les autres pages.**

**1 modification s'applique sur 1 seul fichier**

Si on imagine que vous avez 20 pages html statiques, et que vous souhaitez ajouter un logo (ou toutes informations : changement du nom de la feuille de style, ajout d'un slogan, etc..) vous devrez effectuer 20 modifications (1 modification dans chaque page) ce qui risque d'être assez contraignant dans la mesure où on est au moins sur d'une chose : le header (haut de site) et le footer (bas de site) sont communs à toutes les pages. (Seul le contenu change).

Si le header et le footer sont communs à toutes les pages web, pourquoi ne pas en déclarer un seul qui sera appelé partout ?

Ceux d'entre vous qui ont déjà développé un site web statique en Html / Css comprendront aisément que l'objectif est de simplifier la maintenance et les mises à jour.



Voyons l'arborescence suivante :

```
📁 mon_site
---- 📁 css
----- 📄 style.css
---- 📁 js
---- 📁 img
---- 📁 inc
----- 📄 haut.inc.php
----- 📄 menu.inc.php
----- 📄 bas.inc.php
---- 📄 index.php
```

Nous venons de passer les fichiers .html en extension .php et nous avons également créé un dossier nommé "/inc/" avec plusieurs fichiers à l'intérieur qui vont nous permettre de découper notre site en différentes parties.

*Les fichiers inc montrent (pour un développeur qui prendrait connaissance du projet en cours de route) qu'ils sont destinés à l'inclusion et ne sont pas des pages à part entière.*

Pour mieux le gérer, découpons notre site web :

```

<!Doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Base Template Responsive 2 colonnes Full Width</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" lang="fr" content="DESCRIPTION DU SITE">
    <meta name="author" content="AUTEUR">
    <meta name="robots" content="index, follow">

    <!-- Icônes -->
    <link rel="favicon-icon" href="img/favicon.png">
    <link rel="shortcut icon" href="img/favicon.ico">

    <!-- CSS -->
    <link rel="stylesheet" href="css/style.css">

    <!-- JS -->
    <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
  </head>
  <body>
    <header>
      <div class="container">
        <div class="header-logo">
          <h1 class="header-nom-site">Nom Du Site</h1>
        </div>
        <div class="header-slogan">
          <p>Le Slogan du Site</p>
        </div>
        <div class="header-droite">
          <span>Zone Haut Droite</span>
        </div>
        <div class="clear"></div>
      </div>
    </header>

```

HAUT.INC.PHP

```

<nav>
  <div class="container">
    <ul>
      <li><a href="index.html">Accueil</a></li>
      <li><a href="actualites.html">Actualités</a></li>
      <li><a href="contact.html">Contact</a></li>
    </ul>
  </div>
</nav>

```

MENU.INC.PHP

```

<section>
  <div class="container">
    <main>
      <h1>Accueil</h1>
      <hr>

```

```

      <!-- Titre et niveaux -->
      <h2>Titre niveau 2</h2>
      <p>Voici le texte de notre page d'accueil Voici le texte de notre page d'accueil</p>
      <hr>

```

```

      <!-- Image Responsive -->
      <h3>Titre 3</h3>
      <p></p>
      <hr>

```

```

    </main>
    <aside>
      <h3>Colonne</h3>
      <p>Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil</p>

```

```

      </aside>
      <div class="clear"></div>
    </div>
  </section>

```

BAS.INC.PHP

```

  <footer>
    <div class="container">
      <p>© Copyright 2016</p>
    </div>
  </footer>
</body>
</html>

```

La partie du haut : haut.inc.php :

haut.inc.php

```

<!Doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Site Web</title>
    <meta name="viewport" content="width=device-, initial-scale=1">
    <meta name="description" lang="fr" content="DESCRIPTION DU SITE">
    <meta name="author" content="AUTEUR">
    <meta name="robots" content="index, follow">

```

```

<!-- Icônes -->
<link rel="favicon-icon" href="img/favicon.png">
<link rel="shortcut icon" href="img/favicon.ico">

<!-- CSS -->
<link rel="stylesheet" href="css/style.css">

<!-- JS -->
<script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
</head>
<body>
<header>
  <div class="container">
    <div class="header-logo">
      <h1 class="header-nom-site">Nom Du Site</h1>
    </div>
    <div class="header-slogan">
      <p>Le Slogan du Site</p>
    </div>
    <div class="header-droite">
      <span>Zone Haut Droite</span>
    </div>
    <div class="clear"></div>
  </div>
</header>

```

Comme vous pouvez le constater, nous gardons tout le code appartenant au haut du site.

Ce n'est pas grave si certaines balises tel que `<body>` `<html>` etc. ne sont pas fermées car nous n'avons pas terminé la découpe du site web.

La partie du menu de navigation : `menu.inc.php` :

```

menu.inc.php

<nav>
  <div class="container">
    <ul>
      <li><a href="index.php">Accueil</a></li>
      <li><a href="actualites.php">Actualités</a></li>
      <li><a href="contact.php">Contact</a></li>
    </ul>
  </div>
</nav>

```

Nous modifions le menu avec l'extension `.php` au lieu de `.html`.

La partie du bas : `bas.inc.php` :

```

bas.inc.php

<footer>
  <div class="container">
    © Copyright 2016
  </div>
</footer>
</body>
</html>

```

Les balises que nous avions ouvertes plus haut (comme `<body>` `<html>`) sont refermées ici dans ce fichier.

Il reste uniquement la partie du milieu (spécifique à chaque page web) pour chaque fichier :

La page d'Accueil : `index.php`

```

index.php

<?php require_once('inc/haut.inc.php'); ?>
<?php require_once('inc/menu.inc.php'); ?>
<section>
  <div class="container">
    <main>
      <h1>Accueil</h1>
      <hr>

      <!-- Titre et niveaux -->
      <h2>Titre niveau 2</h2>
      <p>Voici le texte de notre page d'accueil Voici le texte de notre page d'accueil</p>
      <hr>

      <!-- Image Responsive -->
      <h2>Titre 3</h2>

```

```
<p></p>
<hr>
</main>
<aside>
    <h2>Colonne</h2>
    <p>Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil Voici la colonne de droite pour la page d'accueil</p>
    </aside>
    <div class="clear"></div>
</div>
</section>
<?php require_once('inc/bas.inc.php'); ?>
```

La page d'actualités : acualites.php

actualites.php

**La page de contact : contact.php**

```
<?php require_once('inc/haut.inc.php'); ?>
<?php require_once('inc/menu.inc.php'); ?>
<section>
  <div class="container">
    <main>
      <h1>Contact</h1>
      <hr>

      <!-- Titre et niveaux -->
      <h2>Titre niveau 2</h2>
      <p>Voici le texte pour nous contacter Voici le texte pour nous contacter </p>
      <hr>

      <!-- Image Responsive -->
      <h2>Titre 3</h2>
      <p></p>
      <hr>
    </main>
    <aside>
      <h2>Colonne</h2>
      <p>Voici la colonne de droite pour la page de contact Voici la colonne de droite pour la page de contact Voici la colonne de droite pour la page de contact Voici la colonne de droite pour la page de contact Voici la colonne de droite pour la page de contact </p>
      <aside>
        <div class="clear"></div>
      </div>
    </aside>
  </section>
<?php require_once('inc/bas.inc.php'); ?>
```

Dans les 3 cas il s'agit à peu près du même code (seul le texte change).

Le header (haut.inc.php), le menu (menu.inc.php) et le bas (bas.inc.php) restent les mêmes, ce qui est normal.

L'avantage avec le système d'inclusion en php, c'est que vous pourrez effectuer 1 seule modification (par exemple dans le fichier haut.inc.php pour l'ajout d'un logo) et cela sera repercuté et diffusé partout puisque ce fichier est inclus dans toutes les autres pages.

Reprenons notre problématique d'origine : si nous devions modifier le logo en haut de site, cela devait nécessiter autant de modifications que ce qu'il y avait de pages.

```
haut.inc.php

<!Doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Site Web</title>
    <meta name="viewport" content="device-, initial-scale=1">
    <meta name="description" lang="fr" content="DESCRIPTION DU SITE">
    <meta name="author" content="AUTEUR">
    <meta name="robots" content="index, follow">

    <!-- Icônes -->
    <link rel="favicon-icon" href="img/favicon.png">
    <link rel="shortcut icon" href="img/favicon.ico">

    <!-- CSS -->
    <link rel="stylesheet" href="css/style.css">

    <!-- JS -->
    <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
  </head>
  <body>
    <header>
      <div class="container">
        <div class="header-logo">
          
          <h1 class="header-nom-site">Nom Du Site</h1>
        </div>
        <div class="header-slogan">
          <p>Le Slogan du Site</p>
        </div>
        <div class="header-droite">
          <span>Zone Haut Droite</span>
        </div>
        <div class="clear"></div>
      </div>
    </header>
  </body>

```

Nous pouvons nous permettre d'ajouter une seule ligne qui sera diffusée dans tous les hauts de pages.

Faites le test ! (*pour ça vous aurez besoin d'une image nommée logo.png présente dans votre dossier img*)

En conclusion pour un site composé de 20 fichiers sous forme de page web :

**Site statique html (sans php) : 20 modifications = 20 fichiers.**

**Site statique html (avec php) : 1 modification = 20 fichiers.**

*Le chiffre 20 est pris à titre d'exemple, cela dépend du site web*

Les inclusions c'est quand même pratique, ça simplifie grandement la maintenance, ça repartit mieux l'organisation et ça permet d'économiser du code ! :smiley:

De nombreux sites utilisent les fonctions include/require et grâce à ces fonctions, les systèmes de fichiers appelés « Template » ont fait leur apparition.

*Les framework et cms incluent des fichiers et utilisent cette technique.*

## Les ARRAY

Une variable Array (appelée aussi tableau de données, ou tableau array) est une sorte variable améliorée, car elle nous permet de conserver en mémoire un ensemble de valeurs (et non pas qu'une seule valeur comme dans le cas d'une variable classique).

**1 variable = 1 valeur**

**1 array = plusieurs valeurs**

Exemple avec 1 variable classique :

```
entraînement.php
?
1 | <?php
2 | $prenom = "Nicolas";
3 | echo "<p>Bonjour $prenom</p>";
```

#### Résultat

Bonjour Nicolas

Jusque là, tout est normal, 1 variable peut conserver 1 valeur.

Si vous tentez de mettre plusieurs valeurs cela ne sera pas de bon augure :

```
entraînement.php
?
1 | <?php
2 | $prenom = "Nicolas, Marie, Grégoire, Sylvain, Celine";
3 | echo "<p>Bonjour $prenom</p>";
```

#### Résultat

Bonjour Nicolas, Marie, Grégoire, Sylvain, Celine

Au moment de l'affichage via l'instruction `echo` nous ne pouvons pas choisir d'afficher 1 seul prénom, c'est donc toute la liste qui s'affiche. Ce qui montre bien qu'une variable n'est pas faite pour conserver plusieurs valeurs.

75

## Notre premier array : une liste de prenom

```
entraînement.php
?
1 | <?php
2 | $listePrenom = array("gregoire","nathalie","emilie","françois","georges");
```

Nous déclarons notre variable à peu près comme d'habitude à une exception près, nous utilisons le mot-clé `array`.

Si nous souhaitons voir l'ensemble des valeurs contenues dans notre variable `array`, nous pouvons utiliser l'instruction `print_r` :

```
entraînement.php
?
1 | <?php
2 | $listePrenom = array("gregoire","nathalie","emilie","françois","georges");
3 | echo '<pre>'; print_r($listePrenom); echo '</pre>';
```

Nous ajoutons les balises html `<pre>` autour du `print_r` afin que les informations soient affichées de ligne en ligne.

`print_r` est une instruction d'affichage améliorée permettant de voir le contenu d'un tableau `array`.

`var_dump` est similaire à `print_r` avec des options supplémentaires comme le type et la taille de la valeur contenue dans notre tableau `array`.

```
entraînement.php
?
1 | <?php
2 | $listePrenom = array("gregoire","nathalie","emilie","françois","georges");
3 | echo '<pre>'; print_r($listePrenom); echo '</pre>';
4 | var_dump($listePrenom);
```

## Résultat

Avec `print_r` :

```
Array
(
    [0] => gregoire
    [1] => nathalie
    [2] => emilie
    [3] => françois
    [4] => georges
)
```

Avec `var_dump` :

```
array(5) { [0]=> string(8) "gregoire" [1]=> string(8) "nathalie" [2]=> string(6) "emilie" [3]=> string(9) "françois" [4]=> string(7) "georges" }
```

Avec une représentation plus compréhensible :

clé	valeur
0	gregoire
1	nathalie
2	emilie
3	françois
4	georges

Le dernier résultat (sous forme de représentation) est légèrement retouché afin de faciliter votre compréhension, il n'apparaîtra pas aussi nettement sous forme de tableau sur votre page web.

La variable `array` crée automatiquement des clés (aussi appelées indices) et range les valeurs à l'intérieur d'elle même.

Il y a bien 5 prénoms, de 0 à 4.

Par conséquent, si nous voulons afficher le prénom "émilie", nous passerons par la clé n°2. Exemple :

```
entraînement.php

1 | <?php
2 | $listePrenom = array("gregoire","nathalie","emilie","françois","georges");
3 | echo '<pre>'; print_r($listePrenom); echo '</pre>';
4 | echo $listePrenom[2] . '<br>';

?
```

## Résultat

emilie

Nous pouvons également affecter des valeurs dans un tableau `array` au coup par coup (*pratique lorsque nous n'avons pas toutes les valeurs sous la main immédiatement*), exemple :

```
entraînement.php

1 | <?php
2 | $listePays[] = 'France';
3 | $listePays[] = 'Italie';
4 | $listePays[] = 'Espagne';
5 | $listePays[] = 'Angleterre';
6 | echo "<pre>"; print_r($listePays); echo "</pre>";

?
```

## Résultat

Avec `print_r`

```
Array
(
    [0] => France
```

```

[1] => Italie
[2] => Espagne
[3] => Angleterre
)

```

Sous forme de représentation :

clé	valeur
0	France
1	Italie
2	Espagne
3	Angleterre

Exercice, pourriez-vous afficher la valeur Italie en passant par votre tableau \$listePays ?

Voici la correction :

```

entraînement.php
?
1 | <?php
2 | $listePays[] = 'France';
3 | $listePays[] = 'Italie';
4 | $listePays[] = 'Espagne';
5 | $listePays[] = 'Angleterre';
6 | echo "<pre>"; print_r($listePays); echo "</pre>";
7 | echo $listePays[1] . '<br>';

```

Résultat

Italie

Et si nous voulions afficher tous les pays d'un seul coup ?

Nous pourrions utiliser la fonction [implode](#). Exemple :

```

entraînement.php
?
1 | <?php
2 | $listePays[] = 'France';
3 | $listePays[] = 'Italie';
4 | $listePays[] = 'Espagne';
5 | $listePays[] = 'Angleterre';
6 | echo "<pre>"; print_r($listePays); echo "</pre>";
7 | echo $listePays[1] . '<br>';
8 | echo implode($listePays, '<br>');

```

`implode` permet d'extraire les valeurs d'un tableau array, cette fonction attend 2 arguments, 1 array à explorer et 1 séparateur (ce qui permettra de séparer les valeurs une fois extraites).

`implode` est l'une des rares fonctions à ne pas respecter d'importance dans l'ordre des arguments (on peut mettre le array à explorer en premier ou en 2e. Idem pour le séparateur du coup).

Résultat

France

Italie

Espagne

Angleterre

Nous aurions également pu utiliser une boucle.

## Parcourir un tableau array avec la boucle for

entrainement.php

```

1 | <?php
2 | for($i = 0; $i < sizeof($listePays); $i++)
3 |
4 | {
5 |     echo $listePays[$i] . '<br>';
}

```

La fonction prédéfinie `sizeof` ou la fonction prédéfinie `count` permettent de connaître la taille totale d'un tableau (dans notre cas, il y a 4 pays, donc nous arrêterons de faire des tours de boucles à 4 soit 0, 1, 2 et 3).

*pour des raisons de performances il serait préférable de compter dans une variable hors de la boucle (pour éviter que le calcul soit refait à chaque tour). Pour le moment, je ne cherche pas à faire une version de code optimisée mais fonctionnelle dans la mesure où l'on débute en php.*

Comment cette boucle fonctionne ?

Nous demandons à `$listePays` de s'afficher et nous devons préciser le numéro d'une clé (ou indice), pour cela nous choisissons `$i` qui sera progressif.

Au tour de boucle 0, `$i` sera à 0, ce qui donnera `echo $listePays[0]` et donc affichera "France".

Au tour de boucle 1, `$i` sera à 1, ce qui donnera `echo $listePays[1]` et donc affichera "Italie".

Au tour de boucle 2, `$i` sera à 2, ce qui donnera `echo $listePays[2]` et donc affichera "Espagne".

Au tour de boucle 3, `$i` sera à 3, ce qui donnera `echo $listePays[3]` et donc affichera "Angleterre".

### Résultat

France

Italie

Espagne

Angleterre

Une boucle est particulièrement adaptée pour parcourir les éléments d'un array, il s'agit de la boucle `foreach` puisqu'il ne sera pas nécessaire de lui fournir un numéro.

## Parcourir un tableau array avec la boucle foreach

entrainement.php

```

1 | <?php
2 | foreach($listePays as $cle => $valeur)
3 |
4 | {
5 |     echo "$cle - $valeur <br>";
}

```

### Résultat

0 - France

1 - Italie

2 - Espagne

Dans la boucle (et dans le echo) les variables \$cle et \$valeur auraient pu être nommées \$x et \$y, leur nom n'a pas d'importance.

L'ordre représente la manière dans laquelle les colonnes seront parcourues : 1ère variable pour la colonne "clé" et 2e variable pour la colonne "valeur".

\$cle ↓	\$valeur ↓
clé	valeur
0	France
1	Italie
2	Espagne
3	Angleterre

le mot clé "as" et symbole "=>" sont quant à eux prédefinis dans le langage, il faut les respecter, c'est une règle de syntaxe lorsqu'on utilise la boucle foreach.

La boucle foreach est prévue pour s'arrêter toute seule lorsqu'il n'y a plus d'éléments à parcourir dans le tableau array.

C'est quand même bien pratique !

78

## Choisir les clés (indices) et créer un tableau array associatif

Il est possible de choisir les clés d'un array (plutôt qu'avoir une numérotation classique), nous appelons cela un tableau array associatif.

Exemple :

entrainement.php

```
1 | $couleur = array("j" => "jaune", "b" => "bleu", "v" => "vert");
2 | echo '<pre>'; print_r($couleur); echo '</pre>';
```

### Résultat

Avec print\_r :

```
Array
(
    [j] => jaune
    [b] => bleu
    [v] => vert
)
```

Sous forme de représentation :

clé	valeur
j	jaune
b	bleu
v	vert

Avec des clés lettrées (non numérique), nous atteindrons les éléments du tableau array de cette manière-là :

entrainement.php

```
1 | $couleur = array("j" => "jaune", "b" => "bleu", "v" => "vert");
```

```
2 | echo '<pre>'; print_r($couleur); echo '</pre>';
3 | echo $couleur['j'] . '<br>' . $couleur['b'] . '<br>' . $couleur['v'] . '<br>';
```

Comme vous pouvez l'apercevoir, nous mettons des lettres entre crochet [] et non pas des numéros afin d'atteindre les éléments par leur clés.

#### Résultat

jaune

bleu

vert

79

## Règle d'écriture pour tableau ARRAY

Pour l'array suivant :

entrainement.php

```
1 | $listeFruit = array("fruit1" => "orange", "fruit2" => "pomme", "fruit3" => "fraise");
2 | echo '<pre>'; print_r($listeFruit); echo '</pre>';
```

Cela donne le résultat suivant :

#### Résultat

Avec `print_r` :

```
Array
(
    [fruit1]  => orange
    [fruit2]  => pomme
    [fruit3]  => fraise
)
```

Sous forme de représentation :

clé	valeur
fruit1	orange
fruit2	pomme
fruit3	fraise

entrainement.php

```
1 | $listeFruit = array("fruit1" => "orange", "fruit2" => "pomme", "fruit3" => "fraise"); // déclaration array
2 | echo "fruit1 : '$listeFruit[fruit1]' . '<br>'; // quotes dans les crochets
3 | echo "fruit1 : $listeFruit[fruit1] <br>"; // pas de quotes dans les crochets
```

Règle d'écriture à connaître : lorsque nous affichons un élément dans un array associatif (avec des clés (indices) lettrées) il faut le faire soit avec de la concaténation et des quotes (ou guillemets) DANS les crochets.

Ou alors, nous pouvons aussi réaliser la totalité de l'affichage entre guillemets à condition de ne pas mettre de quotes ou guillemets dans les crochets.

L'option consistant à écrire la totalité de l'affichage entre guillemets et sans caractère quotes ou guillemets dans les crochets fonctionne pour les array simple mais pas pour les array multidimensionnels

Il est possible de créer des tableaux array à l'intérieur d'autres tableaux array, lorsqu'une imbrication de tableau array est faite, nous parlons de tableaux array multidimensionnels (en gros, il y a plusieurs dimensions).

Plus simplement, un tableau multidimensionnel désigne un ou plusieurs tableaux à l'intérieur d'un tableau principal .

Exemple :

```
entraînement.php ?
```

```
1 | <?php
2 | $tabMultidimensionnel = array(0 => array("prenom" => "Julien", "nom" => "Cottet"), 1 => array("prenom" => "Nicolas", "nom" => "Lafaye"));
3 | echo '<pre>'; print_r($tabMultidimensionnel); echo '</pre>';
```

#### Résultat

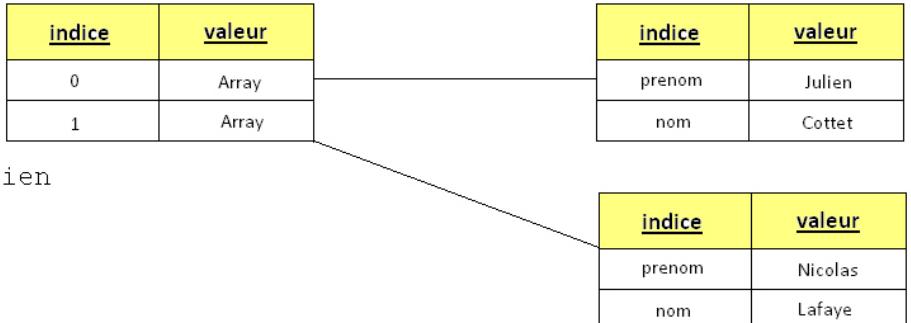
```
Array
(
    [0] => Array
        (
            [prenom] => Julien
            [nom] => Cottet
        )

    [1] => Array
        (
            [prenom] => Nicolas
            [nom] => Lafaye
        )
)
```

### Tableau Multidimensionnel

```
Array
(
    [0] => Array
        (
            [prenom] => Julien
            [nom] => Cottet
        )

    [1] => Array
        (
            [prenom] => Nicolas
            [nom] => Lafaye
        )
)
```



Si nous voulions sortir le prénom "Julien" en passant par notre tableau, nous devrions faire suivre plusieurs crochets :

```
entraînement.php ?
```

```
1 | <?php
2 | $tabMultidimensionnel = array(0 => array("prenom" => "Julien", "nom" => "Cottet"), 1 => array("prenom" => "Nicolas", "nom" => "Lafaye"));
3 | echo $tabMultidimensionnel[0]['prenom'];
```

#### Résultat

Julien

Lors de l'utilisation des tableaux ARRAY, il est possible d'indexer les clés numériquement mais aussi de façon "associative" (autrement dit, avec 1 nom). Lorsque c'est possible, les développeurs utilisent et préfèrent l'indexation "associative" (nominative) afin de pouvoir donner des noms de clés clairs et ayant du sens.

J'espère que ce chapitre est clair pour vous ! sinon n'hésitez pas à le relire autant de fois que nécessaire !

## Classes et Objets

81

## Classes & Objets

Les classes et objets sont avant tout un type de données regroupant des variables (appelées propriétés ou attributs) et des fonctions utilisateurs (appelées méthode).

On parle de programmation orientée objet lorsque leur utilisation est généralisée à tout un site et que la conceptualisation suit cet état d'esprit.

Le vocabulaire diffère légèrement en Programmation Orientée Objet, nous ne parlerons plus de fonctions mais de méthodes, nous ne parlerons plus de variables mais d'attributs (ou propriétés).

Créons une classe nommée Voiture :

```
entrainement.php
```

```
1  <?php
2  class Voiture
3  {
4      public $couleur = "rouge";
5      public $annee = 2010;
6      public function demarrer()
7      {
8          return "Je démarre comme ça...";
9      }
10 }
```

Une classe est donc un regroupement de propriétés (variables) et de méthodes (fonctions) sur un sujet en particulier.

Les variables sont plutôt présentes pour décrire le sujet et les méthodes pour indiquer le comportement du sujet.

Nous avons 2 propriétés qui sont \$couleur et \$annee, et 1 méthode qui est demarrer()

Le mot-clé `public` permet de préciser le niveau de visibilité des éléments (`public` = visible de partout).

3 niveaux de visibilité existent : `public`, `protected` et `private` mais il n'est pas nécessaire de les expliquer en détail pour l'instant (ce n'est pas indispensable pour comprendre cet exemple).

Pour s'en servir, nous allons créer (instancier) un objet (issue de la classe Voiture).

```
entrainement.php
```

```
1  <?php
2  class Voiture
3  {
4      public $couleur = "rouge";
5      public $annee = 2010;
6      public function demarrer()
7      {
8          return "Je démarre comme ça...";
9      }
10 }
11 $voiture = new Voiture;
12 echo 'couleur de la voiture : ' . $voiture->couleur . '<br>';
13 echo 'annee de la voiture : ' . $voiture->annee . '<br>';
14 echo 'manière dont la voiture démarre : ' . $voiture->demarrer() . '<br>';
```

### Résultat

couleur de la voiture : rouge

annee de la voiture : 2010

manière dont la voiture démarre : Je démarre comme ça...

#### Quelques Explications :

`new` est un mot-clé permettant d'instancier une classe et donc de produire un objet.

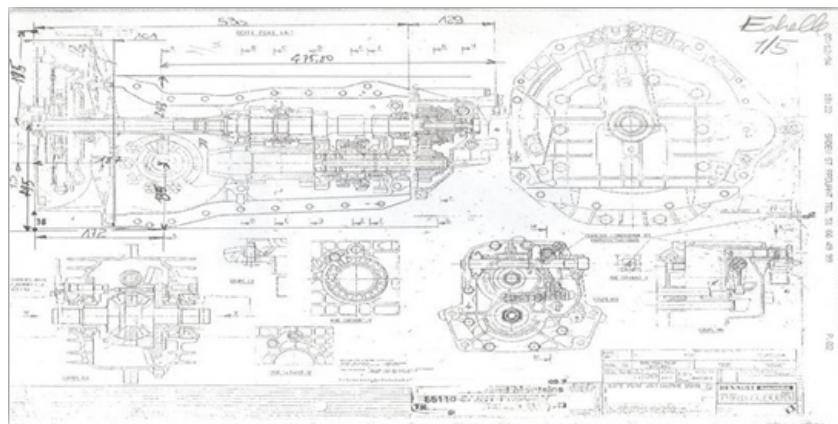
En programmation php, on dit alors que `$voiture` représente un objet (en réalité c'est une référence qui pointe vers un objet en espace mémoire, même principe que pour une [variable](#)).

La flèche "`->`" permet d'atteindre une propriété ou une méthode de l'objet `$voiture` issue de la classe `Voiture`.

82

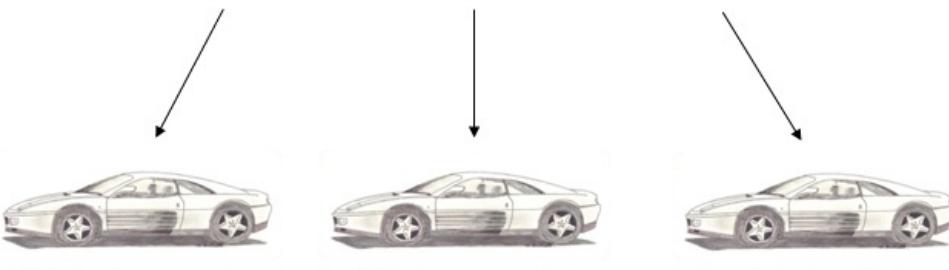
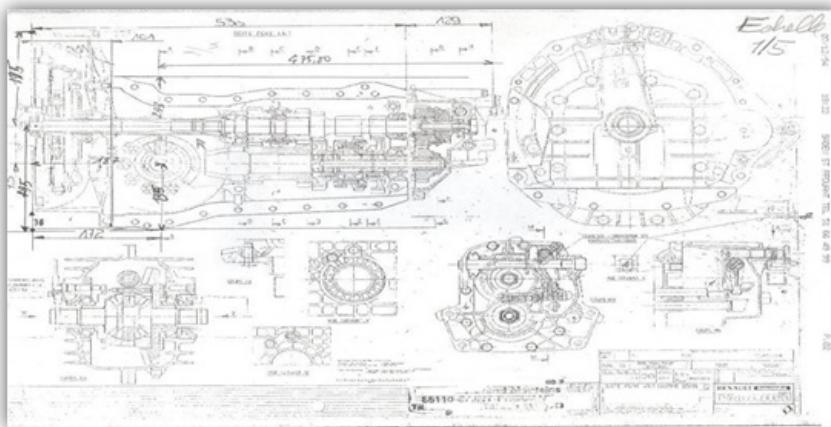
## Qu'est-ce qu'une classe ?

Les classes sont présentes pour « fabriquer » des objets. Dans notre cas, la classe c'est un peu comme le plan de construction d'une voiture :



La classe Voiture n'est pas une voiture, c'est juste son plan de construction.

Un plan de construction voiture (classe Voiture) peut produire et créer plusieurs voitures (objet `$voiture`).



83

## Qu'est-ce qu'un objet ?

En programmation orientée objet, un objet est créé sur le modèle de la classe à laquelle il appartient. Dans notre cas, l'objet c'est la voiture.



*L'objet est une application concrète du plan, c'est une représentation de la classe.*

Pour se servir du code déclaré dans une classe, il faut (dans la majorité des cas) créer un objet.

84

## Développement Orienté Objet

Le développement Orienté Objet ne se résume pas qu'à élaborer quelques classes et créer des objets.

Regrouper son code aide à y voir clair, mais si l'on retrouve du code procédural dans les méthodes d'une classe, ce ne sera qu'un mi-chemin entre le développement procédural et le développement orienté objet.

L'orienté objet c'est avant tout une conceptualisation différente. Nous verrons plus tard comment aborder le code.

## Les superglobales

Les Superglobales sont des variables array, internes et prédéfinies par PHP, qui sont toujours disponibles, quel que soit le contexte.

Pour rappel, une variable array permet de conserver un ensemble de valeurs : [chapitre sur les array](#).

Lorsqu'on parle de disponibilité, on exprime le fait que les superglobales sont à la fois disponibles dans l'espace global (espace par défaut de php) mais aussi dans l'espace local (dans une fonction).

Pour utiliser une superglobale dans une fonction, il ne sera donc pas nécessaire d'effectuer un passage d'argument ou utiliser le mot-clé `global` ;

Les superglobales sont reconnaissables car elles s'inscrivent TOUJOURS en majuscule et commence toutes obligatoirement par un underscore "\_" (sauf \$GLOBALS).

## A quoi servent les superglobales en PHP ?

Superglobale	Description	Exemple d'utilisation
<code>\$GLOBALS</code>	Contient toutes les variables disponibles dans un contexte global	-
<code>\$_SERVER</code>	Contient toutes les informations fournies par le serveur web	Pratique pour connaître le chemin du site, d'un dossier, etc.
<code>\$_GET</code>	Contient les informations fournies en paramètre au script via la méthode GET par l'URL et le protocole HTTP.	Utile pour véhiculer des informations d'une page à l'autre.
<code>\$_POST</code>	Contient les informations fournies par un formulaire via la méthode POST du protocole HTTP.	Utile pour récupérer les saisies postées dans un formulaire par un internaute.
<code>\$_FILES</code>	Contient les informations liées à l'upload d'un (ou plusieurs) fichier(s) par un formulaire (fonctionne en complément de la superglobale <code>\$_POST</code> ).	Utile pour récupérer le(s) fichier(s) uploadé(s) dans un formulaire par un internaute.
<code>\$_COOKIE</code>	Contient les informations fournies par les cookies via le protocole HTTP.	Utile pour conserver des informations sur un internaute.
<code>\$_SESSION</code>	Contient les informations de la session en cours.	Utile pour maintenir une connexion avec un internaute sur un site web
<code>\$_REQUEST</code>	Contient les variables fournies au script (peu importe la méthode utilisée).	Utile pour récupérer des informations sans savoir précisément d'où elles proviennent
<code>\$_ENV</code>	Contient les variables fournies par l'environnement.	-

Très important : Toutes les superglobales sont ARRAY car elles contiennent plusieurs informations.

Ce qui veut dire que nous pourrons utiliser un `print_r` pour afficher leur contenu

## Superglobales `$GLOBALS`, `$_ENV`, `$_REQUEST`

Quelques tests :

```
entraînement.php
```

```

1 | <?php
2 | echo '<pre>'; print_r($GLOBALS); echo '</pre>';
3 | echo '<pre>'; print_r($_ENV); echo '</pre>';
4 | echo '<pre>'; print_r($_REQUEST); echo '</pre>';

```

Comme vous pouvez le constater au vue du résultat, \$GLOBALS contient les autres superglobales.

\$GLOBALS se contient elle même, c'est la raison pour laquelle vous voyez écrit \*RECURSION\* lors de l'affichage.

88

## Superglobale \$\_SERVER

\$\_SERVER est une variable très pratique puisqu'elle nous permet d'obtenir des informations sur le serveur :

entraînement.php

```
1 | <?php
2 | echo '<pre>'; print_r($_SERVER); echo '</pre>';
```

\$\_SERVER est un array (comme toutes les superglobales).

### Résultat

```
Array
(
    [HTTP_HOST] => localhost
    [HTTP_USER_AGENT] => Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101 Firefox/38.0
    [HTTP_ACCEPT] => text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
    [HTTP_ACCEPT_LANGUAGE] => fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3
    [HTTP_ACCEPT_ENCODING] => gzip, deflate
    [HTTP_REFERER] => http://localhost/dev/
    [HTTP_COOKIE] => CKFinder_Path=Images%3A%2F%3A1; SESS91411f2d7290712346b63b053a3ed78b=BA-mJkjV6I
    [HTTP_CONNECTION] => keep-alive
    [PATH] => C:\ProgramData\Oracle\Java\javapath;C:\Program Files (x86)\NVIDIA Corporation\PhysX\Code\{SystemRoot} => C:\Windows
    [COMSPEC] => C:\Windows\system32\cmd.exe
    [PATHEXT] => .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
    [WINDIR] => C:\Windows
    [SERVER_SIGNATURE] =>
    [SERVER_SOFTWARE] => Apache/2.2.17 (Win32) PHP/5.3.5
    [SERVER_NAME] => localhost
    [SERVER_ADDR] => 127.0.0.1
    [SERVER_PORT] => 80
    [REMOTE_ADDR] => 127.0.0.1
    [DOCUMENT_ROOT] => C:/wamp/www/
    [SERVER_ADMIN] => admin@localhost
    [SCRIPT_FILENAME] => C:/wamp/www/dev/test.php
    [REMOTE_PORT] => 65431
    [GATEWAY_INTERFACE] => CGI/1.1
    [SERVER_PROTOCOL] => HTTP/1.1
    [REQUEST_METHOD] => GET
    [QUERY_STRING] =>
    [REQUEST_URI] => /dev/test.php
    [SCRIPT_NAME] => /dev/test.php
    [PHP_SELF] => /dev/test.php
    [REQUEST_TIME] => 1432986966
)
```

Même principe que pour un tableau array classique, si nous souhaitons obtenir une information en particulier, il suffit de passer par la clé (indice) du tableau :

entraînement.php

```
1 | <?php
2 | echo $_SERVER['DOCUMENT_ROOT'];
```

La clé DOCUMENT\_ROOT de la superglobale \$\_SERVER permet d'avoir accès au chemin dans lequel nous travaillons actuellement. Ce qui peut être très pratique dans le cadre d'un développement.

Les autres superglobales n'étant pas détaillées dans cette page possèdent leurs propres exemples dans les chapitres suivants.

### Superglobale GET - Passage d'argument d'une page à l'autre

Dans la phase de construction d'un site web, il peut être utile de véhiculer des informations d'une page à une autre.

Pourquoi ?

Imaginons le catalogue (catalogue.php) d'une boutique ecommerce avec plusieurs produits :

#### Catalogue

<b>1 Tshirt Jaune</b>  10 € <a href="#">ajout au panier</a>	<b>2 Tshirt Bleu</b>  35 € <a href="#">ajout au panier</a>	<b>3 Tshirt Rouge</b>  20 € <a href="#">ajout au panier</a>	<b>4 Tshirt Gris</b>  25 € <a href="#">ajout au panier</a>	<b>5 Tshirt Noir</b>  20 € <a href="#">ajout au panier</a>
<b>6 Tshirt Blanc</b>  15 € <a href="#">ajout au panier</a>	<b>7 Tshirt Vert</b>  25 € <a href="#">ajout au panier</a>			

L'affichage ci-dessus est généré grâce à une requête SQL qui rappatrie toutes les données de la base.

La plupart du temps, nous pouvons cliquer sur ces produits et arriver sur la fiche produit contenant davantages d'informations détaillées.

Maintenant imaginons la fiche produit détaillée (ficheProduit.php) :

## Fiche Produit



### Tshirt Rouge

Ref: TSHIRT\_ROUGE

PROMO

NOUVEAUTE

▼ [Lire la description complète](#)

Couleur :

Taille :

Quantité :

▾

*En stock*

-10 %

**20 €**

*Livraison offerte !*

Ajouter au panier



Êtes vous d'accord sur le fait que s'il y a 1500 produits, il n'y aura pas 1500 fichiers du type `ficheProduit1.php`, `ficheProduit2.php`, `ficheProduit3.php`, `ficheProduit4.php`, `ficheProduit5.php`, etc.

Bien entendu cela serait ingérable et le contraire de dynamique !

Par conséquent, nous aurons qu'un seul fichier se nommant `ficheProduit.php` et pouvant présenter potentiellement tous les produits. C'est ce qu'on appelle un template (un modèle de page pouvant présenter différents contenus)

Mais s'il n'y a qu'un seul fichier, comment saura t'il quel produit il doit présenter ?

Et bien nous n'allons pas arriver sur la page `ficheProduit.php` les mains vides, nous irons sur la page `ficheProduit.php?idProduit=1` ou par exemple `ficheProduit.php?idProduit=3` (pour le tshirt rouge).

Pour rappel, l'`idProduit` représente le numéro du produit dans la base de données

Ce système est le même pour tous les sites web ! Essayez donc de vous balader dans un site web ecommerce et vous verrez bien des informations passées dans l'url (si vous y faites attention).

Voici un exemple avec le site web de ZARA :

Ces informations n'apparaîtront peut être pas aussi nettement que ?idProduit=1 car souvent les sites font de l'url rewriting (réécriture d'url).

L'affichage va donc être généré grâce à une requête SQL qui rappatriera 1 produit en particulier de la base (1 ligne de résultat).

Pour afficher le bon produit dans la fiche produit, vous l'aurez compris, il nous faudra passer et récupérer le bon numéro de produit dans l'url.

Pour passer un numéro de produit c'est très simple puisque nous n'avons besoin que du HTML :

```
catalogue.php
?
1 | <a href="ficheProduit.php?idProduit=1"> Produit 1 </a>
```

Le format a respecter est cle=valeur.

Créez le fichier catalogue.php et ficheProduit.php et vous verrez qu'en cliquant sur le lien de la page catalogue.php, vous arrivez sur la page ficheProduit.php avec le numéro de produit dans l'url.

Pour récupérer cette information, nous aurons besoin de la superglobale \$\_GET :

```
ficheProduit.php
?
1 | echo $_GET['idProduit'];
```

C'est tout simple, il suffit de mettre le nom de la clé (définie auparavant dans le html).

`$_GET` permet de récupérer une ou plusieurs information(s) contenue dans l'url afin de s'en servir dans la page web.

Nous utilisons les crochets car toutes les superglobales sont des tableaux ARRAY.

## Résultat

1

Par précaution, si jamais pour une raison x ou y l'url ne contient pas d'arguments (paramètres) ou que l'internaute tente d' aller directement sur

ficheProduit.php sans passer par notre lien créé sur la page catalogue.php, cela donnera une erreur undefined variable car nous chercherons à récupérer quelque chose de l'url qui n'existe pas.

Pour cette raison, il est préférable de prévoir une condition :

```
ficheProduit.php  
1 | <?php  
2 | if(!empty($_GET['idProduit']))  
3 | {  
4 |     echo $_GET['idProduit'];  
5 | }
```

Nous demandons si l'url n'est pas vide (avec le point d'exclamation "!"), autrement dit, y'a t'il un idProduit à récupérer dans l'url ? Si oui, nous le récupérons. Si non, l'interpréteur ignorera cette portion de code.

Autre exemple avec le fichier de test "get1.php" :

```
get1.php  
1 | <a href="get2.php?produit=tshirt&couleur=bleu&prix=30"> Cliquez ici ! </a>
```

Nous pouvons passer toutes les informations qui nous intéresse dans l'url à condition d'éviter les espaces, les accents, et de respecter le format cle=valeur.

C'est ce format qui permettra de remplir le tableau array de la superglobale \$\_GET.

Voici le fichier get2.php (pour s'entraîner) :

```
get2.php  
1 | <?php  
2 | if(!empty($_GET))  
3 | {  
4 |     echo "produit $_GET[produit] <br>";  
5 |     echo "couleur $_GET[couleur] <br>";  
6 |     echo "prix $_GET[prix] <br>";  
7 |     echo implode (" - ", $_GET);  
8 |     echo '<pre>'; print_r($_GET); echo '</pre>';  
9 | }
```

*Pas de quotes dans les crochets d'un tableau array si l'instruction echo englobe le tout entre guillemets*

#### Résultat

L'instruction echo:

produit tshirt  
couleur bleu  
prix 30

La fonction pré définie implode :

tshirt - bleu - 30

Avec le print\_r :

```
Array  
(  
    [produit] => tshirt  
    [couleur] => bleu  
    [prix] => 30  
)
```

Avec une représentation plus compréhensible :

clé	valeur
produit	tshirt

couleur	bleu
prix	30

C'est donc notre lien html <a> qui envoie des informations dans l'url et qui nous permet de construire le tableau ARRAY de \$\_GET.

Avez-vous compris l'intérêt et à quoi sert \$\_GET ? Avez-vous compris comment fonctionne \$\_GET ? C'est indispensable pour la suite !

90

## Superglobale POST - Récupération des saisies d'un formulaire

Utiliser les formulaires avec PHP permet d'échanger des données avec l'internaute. C'est-à-dire, lui envoyer des données, mais aussi en recevoir de sa part.

Autant dire qu'un site web sans formulaire n'existe pas !

Les formulaires reviennent très régulièrement sur les sites web, par exemple pour un formulaire de contact, un formulaire d'inscription en tant que membre, un formulaire de recherche, un formulaire de recrutement, un formulaire de satisfaction, un formulaire de commande, etc..

Comment récupérer les informations d'un formulaire ?

Nous savons qu'avec les langages Html et Css il est possible de concevoir un formulaire.

Voici donc un simple formulaire demandant une adresse (avec ville et code postal).

```
formulaire1.php
?
1 | <form method="post" action="">
2 |   <label for="ville">ville</label><br>
3 |   <input type="text" name="ville" id="ville" placeholder="saisir 1 ville"><br><br>
4 |   <label for="cp">code postal</label><br>
5 |   <input type="text" name="cp" id="cp" placeholder="saisir 1 code postal"><br><br>
6 |   <label for="adresse">adresse</label><br>
7 |   <textarea name="adresse" id="adresse" placeholder="saisir 1 adresse"></textarea><br><br>
8 |   <input type="submit">
9 | </form>
```

Pour rappel, le formulaire html est contenu entre les balises <form> </form> .

L'attribut method permet de préciser comment vont circuler les données, 2 choix s'offrent à nous : POST ou GET.

Retenez que nous utilisons dans la majorité des cas POST.

L'attribut action permet de préciser la page de destination (sur laquelle l'internaute sera redirigée après avoir soumis le formulaire en cliquant sur le bouton submit) pour effectuer les traitements.

Dans notre cas, nous resterons sur la même page web pour effectuer les traitements et n'emmènerons l'internaute nulle part.

La balise <label> permet de créer une étiquette, son attribut for et est relié à l'attribut id de la balise<input> (lorsqu'on clique sur le texte, le curseur se place dans la case de saisie).

L'id peut être utilisé comme un id css ou javascript, mais dans ce contexte ce n'est pas son rôle premier.

Label for et id n'est donc pas d'une importance capitale pour faire fonctionner notre formulaire mais je vous recommande de toujours le prévoir car cela fait partie des normes accessiweb (pour les mal-voyants). Facilitons leur la navigation !

L'attribut placeholder est également facultatif, il permet d'afficher une indication supplémentaire avec un texte opaque présaisi qui s'effacera lorsque l'internaute commencera à saisir au clavier des informations dans la case <input> .

L'attribut name est capital puisque sans lui nous ne pourrions pas récupérer les données même avec PHP.



#### ➤ Attention

L'attribut `name` est indispensable !!! Un attribut `name` doit être écrit par élément de formulaire (1 `name` = 1 saisie récupérée).

La valeur que vous placerez dans l'attribut `name` ne doit pas comporter d'espace, ni d'accent et de préférence ne pas contenir de majuscule (sensible à la casse).

Nous avons mis le `name` "cp" pour le champ code postal

Le bouton `submit` est présent afin que l'on puisse valider le formulaire.

[Vous pouvez consulter la page récapitulant tous les éléments des formulaires HTML](#)

#### Résultat

ville  
saisir 1 ville

cp  
saisir 1 code postal

adresse  
saisir 1 adresse

Envoyer

Un formulaire est une porte ouverte sur le serveur... Ce qui signifie qu'une personne mal intentionnée peut potentiellement exploiter des failles de sécurité...

Ce formulaire est parfaitement valide en html mais ne pourra pas être exploité sans l'aide de PHP.

Alors voici la partie PHP pour la récupération des données sur un formulaire :

```
formulaire1.php

1 | <?php
2 |   echo 'ville : ' . $_POST['ville'] . '<br>';
3 |   echo 'cp : ' . $_POST['cp'] . '<br>';
4 |   echo 'adresse : ' . $_POST['adresse'] . '<br>';
5 | ?>
6 | <form method="post" action="">
7 |   <label for="ville">ville</label><br>
8 |   <input type="text" name="ville" id="ville" placeholder="saisir 1 ville"><br><br>
9 |   <label for="cp">cp</label><br>
10 |   <input type="text" name="cp" id="cp" placeholder="saisir 1 code postal"><br><br>
11 |   <label for="adresse">adresse</label><br>
12 |   <textarea name="adresse" id="adresse" placeholder="saisir 1 adresse"></textarea><br><br>
13 |   <input type="submit">
14 | </form>
```

#### Quelques Explications :

Comme d'habitude l'instruction `echo` peut être traduit par "affiche moi".

`echo $_POST['ville']` peut être traduit par "affiche moi la ville postée".

Le texte contenu entre crochet est en liaison avec le `name` de l'élément HTML.

Faites le test en remplissant votre formulaire et en cliquant sur le bouton submit!

<b>( ! ) Notice: Undefined index: ville in C:\wamp\www\dev\test.php on line 5</b>				
Call Stack				
#	Time	Memory	Function	Location
1	0.0033	368272	{main}()	..\test.php:0

ville :

<b>( ! ) Notice: Undefined index: cp in C:\wamp\www\dev\test.php on line 6</b>				
Call Stack				
#	Time	Memory	Function	Location
1	0.0033	368272	{main}()	..\test.php:0

cp :

<b>( ! ) Notice: Undefined index: adresse in C:\wamp\www\dev\test.php on line 7</b>				
Call Stack				
#	Time	Memory	Function	Location
1	0.0033	368272	{main}()	..\test.php:0

adresse :

ville

saisir 1 ville

cp

saisir 1 code postal

adresse

saisir 1 adresse

..

Résultat (avant d'avoir cliqué sur submit)

Des erreurs "undefined" s'affichent sur la page web car la toute première fois où l'interpréteur exécute le code, vous lui demandez d'aller chercher des saisies postées mais s'il s'agit du moment où le formulaire est affiché par l'internaute, l'internaute n'aura pas encore eu le temps de poster quoi que ce soit (puisque'il vient d'arriver à la seconde).

L'interpréteur vous indique que les clés du tableau array \$\_POST (respectivement ville, cp, et adresse) ne sont pas connues (ces clés existeront uniquement si l'internaute soumet le formulaire en cliquant sur le bouton submit).

Même si des erreurs apparaissent, vous pouvez quand même faire le test en remplissant le formulaire et en cliquant sur le bouton submit, vous verrez que ce code fonctionne :

Recuperation des données saisies :

ville : Paris

cp : 75015

adresse : 30 rue de la convention

ville

saisir 1 ville

cp

saisir 1 code postal

adresse

saisir 1 adresse

..

Résultat (après avoir cliqué sur submit)

Les saisies de l'internaute vont s'afficher correctement sur la page web.

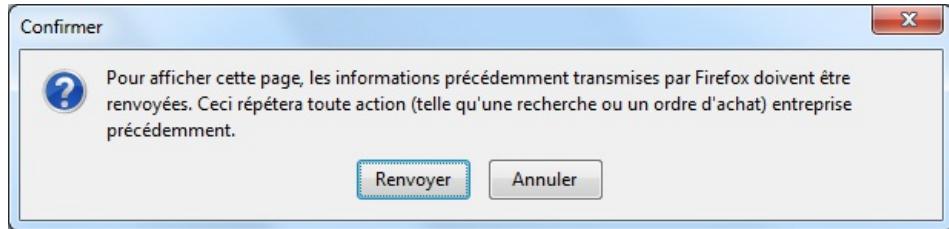
C'est bien là que ça se joue, lorsque le bouton submit est cliqué par l'internaute, LA PAGE SE RECHARGE ET L'INTERPRETEUR RELIE LE CODE, cette fois-ci l'interpréteur est capable d'afficher les saisies ville, cp et adresse postées.

Pour simuler la première visite sur la page, effectuer au clavier **F6+Enter**

Pour recharger la page (et répéter la dernière action effectuée), appuyez sur la touche **F5** ou **Ctrl+R**

Afficher la page pour la première fois (cliquer sur l'url et faire entrer au clavier ou **F6+Entrer**) et recharger la page (**F5** ou **Ctrl+R**) ne sont pas du tout pareil !

Vous avez sans doute dû déjà voir ce message :



Il est important de prendre le temps de le comprendre ! Ne vous contentez pas de l'approximatif, ayez toujours envie de comprendre précisément et creuser un sujet jusqu'au bout.

Ce message vous demande si vous êtes d'accord pour répéter la dernière action effectuée.

Si la dernière action était un POST (un clic sur le bouton submit), cela fera un 2e POST (comme un 2e clic sur le bouton submit).

Il était utile de porter un regard sur ces deux situations pour que vous en ayez connaissance (et prendre le temps de comprendre les mécanismes en détail) mais maintenant que c'est fait, bien entendu nous n'allons pas garder ces erreurs.

Nous allons prévoir une condition IF qui réglera tous nos problèmes, car il n'y a jamais de problèmes mais que des solutions !

Nous allons dire :

...Si l'internaute a posté quelque chose, tu me l'affiches.

Sinon rien (il n'y aura pas de sinon (else)).

Voici le code final :

```
formulaire1.php

1 <?php
2 if(!empty($_POST))
3 {
4     echo 'Recuperation des données saisies : <br>';
5     echo 'ville : ' . $_POST['ville'] . '<br>';
6     echo 'cp : ' . $_POST['cp'] . '<br>';
7     echo 'adresse : ' . $_POST['adresse'] . '<br>';
8 }
9 ?>
10 <form method="post" action="">
11 <label for="ville">ville</label><br>
12 <input type="text" name="ville" id="ville" placeholder="saisir 1 ville"><br><br>
13 <label for="cp">cp</label><br>
14 <input type="text" name="cp" id="cp" placeholder="saisir 1 code postal"><br><br>
15 <label for="adresse">adresse</label><br>
16 <textarea name="adresse" id="adresse" placeholder="saisir 1 adresse"></textarea><br><br>
17 <input type="submit">
18 </form>
```

#### Quelques Explications :

Nous disons à l'interpréteur PHP : si le POST n'est pas (caractère "!=") vide (c'est donc qu'il est rempli et que l'internaute a posté), à ce moment-là on peut afficher les saisies sur la page web.

Dans le cas où l'internaute arrive pour la 1ère fois sur la page web, l'interpréteur ignorera les lignes PHP de récupération puisqu'il ne rentrera pas dans la condition.

Pour être clair :

- La toute première fois quand on clique sur l'url et qu'on fait "entrer", nous ne rentrons pas dans le IF.
- Toutes les fois suivantes si nous cliquons sur le bouton "envoi", le code se ré-exécute et nous rentrons dans le IF.

Le texte contenu entre crochet est en liaison avec le name de l'élément HTML.

#### Résultat

Recuperation des données saisies :

ville : Paris

cp : 75015

adresse : 30 rue de la convention

ville

saisir 1 ville

cp

saisir 1 code postal

adresse

saisir 1 adresse

Envoyer

Puisque toutes les superglobales sont ARRAY, nous pourrions également demandé un affichage avec l'aide de print\_r :

```
1 <?php
2 if(!empty($_POST))
3 {
4     echo 'Recuperation des données saisies : <br>';
5     echo 'ville : ' . $_POST['ville'] . '<br>';
6     echo 'cp : ' . $_POST['cp'] . '<br>';
7     echo 'adresse : ' . $_POST['adresse'] . '<br>';
8     echo '<pre>'; print_r($_POST); echo '</pre>';
9 }
10 ?>
```

#### Résultat

clé	valeur
ville	Paris (saisie de l'internaute)
cp	75015 (saisie de l'internaute)
adresse	30 rue de la convention (saisie de l'internaute)

#### Formulaire modèle avec récupération de données \$\_POST

Si nous voulions ajouter la structure d'une page web (avec les entêtes, doctype, etc.), nous ferions comme ceci :

formulaireModele.php

```
<?php
$message = "";
if(!empty($_POST))
{
    $message .= 'Recuperation des données saisies : <br>';
    $message .= 'ville : ' . $_POST['ville'] . '<br>';
    $message .= 'cp : ' . $_POST['cp'] . '<br>';
    $message .= 'adresse : ' . $_POST['adresse'] . '<br>';
}
?>
<!Doctype html>
<html>
    <head>
        <link rel="stylesheet" href="structure.css">
        <meta charset="utf-8">
    </head>
    <body>
```

```

<div><?php echo $message; ?></div>
<form method="post" action="">
  <label for="ville">ville</label><br>
  <input type="text" name="ville" id="ville" placeholder="saisir 1 ville"><br><br>
  <label for="cp">cp</label><br>
  <input type="text" name="cp" id="cp" placeholder="saisir 1 code postal"><br><br>
  <label for="adresse">adresse</label><br>
  <textarea name="adresse" id="adresse" placeholder="saisir 1 adresse"></textarea><br><br>
  <input type="submit">
</form>
</body>
</html>

```

`$_POST` permet de récupérer et d'afficher la saisie qui a été postée.

Dans le cadre d'une page web complète, nous mettons le code php le plus en haut possible : d'abord les traitements avant l'affichage (norme et architecture MVC).

Nous ne mettons plus d'instruction `echo` car sinon les informations s'afficheraient au dessus du doctype, leurs positionnements seraient assez contraignant et le code ne passerait pas la validation w3c.

Par conséquent, nous passons par une variable `$message` qui va retenir l'affichage (les messages seront affectés à la variable).

La variable `$message` ne sera pas affichée immédiatement mais un peu plus bas, à l'endroit de notre choix, dans les balises `body /body` afin de respecter la structure de la page web.

Vous pouvez garder ce formulaire comme modèle pour vos prochains développements avec la superglobale `$_POST`.

## Sauvegarde des données venant d'un formulaire

Après avoir récupéré et affiché les saisies d'un formulaire que fait-on ?

Nous pourrions, plus tard, envoyer ces données dans une base de données (c'est comme cela que fonctionnent les pages d'inscription d'un site web).

Nous pourrions également afficher un message de confirmation à l'internaute pour dire que sa validation a été enregistrée.

## Formulaire avec récupération de données `$_POST` sur 2 pages

Créons deux fichiers : `formulaire2.html` et `formulaire3.php`

L'objectif sera d'avoir le formulaire HTML dans le fichier `formulaire2.html` et les traitements dans le fichier `formulaire3.php`

Nous prévoirons 2 champs : `pseudo` & `email`.

```

formulaire2.html

<!Doctype html>
<html>
  <head>
    <link rel="stylesheet" href="structure.css">
    <meta charset="utf-8">
  </head>
  <body>
    <form method="post" action="formulaire3.php">
      <label for="pseudo">pseudo</label><br>
      <input type="text" name="pseudo" id="pseudo" placeholder="saisir 1 pseudo"><br><br>
      <label for="email">email</label><br>
      <input type="email" name="email" id="email" placeholder="saisir 1 email"><br><br>
      <input type="submit">
    </form>
  </body>
</html>

```

Le fichier HTML `formulaire2.html` contient le formulaire, l'action pointe vers le fichier `formulaire3.php`

Par conséquent, en cliquant sur le bouton submit, l'internaute sera automatiquement envoyé sur le fichier `formulaire3.php`

C'est donc dans ce fichier `formulaire3.php` que nous ferons les traitements PHP.

### formulaire3.php

```
<?php  
if(empty($_POST))  
{  
    echo 'Recuperation des données saisies : <br>';  
    echo 'pseudo : ' . $_POST['pseudo'] . '<br>';  
    echo 'email : ' . $_POST['email'] . '<br>';  
}  
?>
```

91

## Sauvegarder les informations d'un formulaire dans un fichier texte

Voici Votre dossier est composé de 2 fichiers :

-----  formulaire2.html  
-----  formulaire3.php

Nous allons faire en sorte de sauvegarder les données saisies (postées par les internautes) dans un fichier texte qui sera généré à la volée.

Pour cela, utilisons les fonctions pré définie de PHP : [fopen](#), [fwrite](#), [fclose](#).

Reprends le code Html suivant :

### formulaire2.html

```
<!Doctype html>  
<html>  
  <head>  
    <link rel="stylesheet" href="structure.css">  
    <meta charset="utf-8">  
  
  </head>  
  <body>  
    <form method="post" action="formulaire3.php">  
      <label for="pseudo">pseudo</label><br>  
      <input type="text" name="pseudo" id="pseudo" placeholder="saisir 1 pseudo"><br><br>  
      <label for="email">email</label><br>  
      <input type="email" name="email" id="email" placeholder="saisir 1 email"><br><br>  
      <input type="submit">  
    </form>  
  </body>  
</html>
```

Et voici le code PHP correspondant:

### formulaire3.php

```
<?php  
if(empty($_POST))  
{  
    echo 'Recuperation des données saisies : <br>';  
    echo 'pseudo : ' . $_POST['pseudo'] . '<br>';  
    echo 'email : ' . $_POST['email'] . '<br>';  
  
    $f = fopen("sauvegarde.txt","a");  
    fwrite($f, $_POST['pseudo'] . " - ");  
    fwrite($f, $_POST['email'] . "\n");  
    $f = fclose($f);  
}  
?>
```

La fonction pré définie `fopen()` permet d'ouvrir un fichier, il attend 2 arguments, son nom et le mode d'ouverture.

Le mode "a" (que nous utilisons dans cet exemple) permet d'ouvrir le fichier, ou de le créer s'il n'est pas trouvé.

Le fichier est donc ouvert, ou bien créé + ouvert, la variable `$f` représente le fichier ouvert.

`fwrite()` permet d'écrire dans le fichier représenté par `$f`.

\n entre guillemets permet de passer à la ligne dans un fichier (équivalent de `<br>` dans une page web).

`fclose()`, n'est pas indispensable pour faire fonctionner l'exemple, cette fonction pré définie permet de fermer le fichier et ainsi libérer la ressource.

Après avoir posté un pseudo et un email, vous pouvez retourner dans votre dossier et vous devriez voir apparaître un nouveau fichier nommé :

## sauvegarde.txt

```
📁 /post/
---- 📁 formulaire2.html
---- 📁 formulaire3.php
---- 📁 sauvegarde.txt
```

Ouvrez le fichier **sauvegarde.txt** avec notepad, vous verrez que les enregistrements se trouvent à l'intérieur.

La 1ère fois le fichier est créé et ouvert, toutes les fois suivantes le fichier est simplement ouvert.

Contexte et cas d'utilisation : Si l'on souhaite enregistrer des membres à une newsletter et que l'on ne possède pas de BDD, il serait intéressant de le faire via un fichier texte.

## Résultat

pseudo	1	le pseudo posté - l'email posté
saisir 1 pseudo	2	

email  
saisir 1 email

Envoyer

## Lire dans une page web les informations contenues dans un fichier texte

Suite à l'exemple précédent, le fichier **liste.txt** a été créé (généré) et contient des informations (pseudo et email) suite à la validation de notre formulaire.

Nous pouvons effectuer l'opération inverse, c'est à dire, se connecter au fichier en php et lire les informations contenues dedans à l'intérieur :

```
lecture.php

$nomFichier = "liste.txt";
$fichier = file($nomFichier); // la fonction file() fait tout le travail, elle retourne chaque ligne d'un fichier à des indices différents d'un tableau array.
print "<pre>", print_r($fichier), print "</pre>"; // Affichage du tableau Array dans sa structure.

foreach($fichier as $ligne) // Parcours du tableau Array pour un affichage plus conventionnel.
{
    echo $ligne."<br>";
}

echo "";
echo implode($fichier, "<br>"); // Affichage du tableau Array avec un passage à la ligne.
```

## Explications

**\$nomFichier** représente le nom du fichier à explorer (il faut que le fichier existe et soit dans le même dossier de préférence).

**file** est une fonction PHP prédéfinie permettant d'ouvrir un fichier et de retourner son contenu sous forme de tableau ARRAY (chaque ligne représentera une nouvelle clé du tableau array). Nous lui transmettons la variable **\$nomFichier** afin que le bon fichier soit ouvert.

**print\_r** permet d'afficher le contenu et d'observer la structure du tableau Array

**foreach** permet de boucler et d'afficher toutes les informations du tableau Array

**implode** permet également d'afficher toutes les informations du tableau Array

## Résultat

Avec **print\_r** :

```
Array
(
    [0] => le pseudo posté
    [1] => l'email posté
```

)

Avec `foreach` :

le pseudo posté - l'email posté

Avec `implode` :

le pseudo posté - l'email posté

92

## Créer un formulaire de contact et envoyer un email

### Envoyer un email en PHP

Pour envoyer un email en PHP, il existe la fonction [mail\(\)](#) très pratique.

La fonction mail prend 4 arguments (pas plus, pas moins) dans un ordre précis :

```
<?php  
mail("monadressedereception@gmail.com", "Le sujet", "Le message", "adresseemailexpéditeur@gmail.com");
```

Comme dans la plupart des fonctions, il faut respecter le NOMBRE et l'ORDRE des arguments que l'on transmet à une fonction prédefinie de PHP.

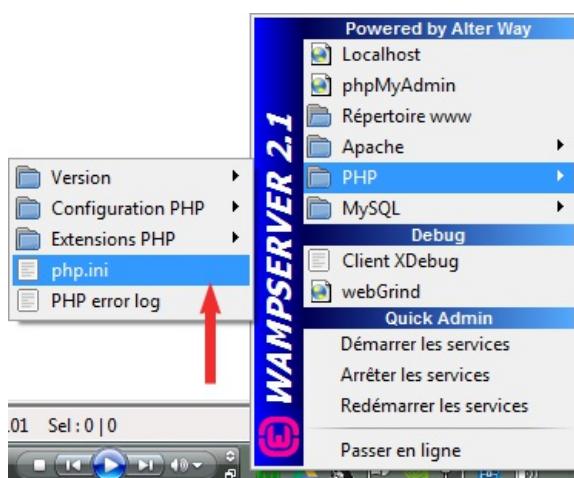
La fonction mail() reçoit toujours 4 ARGUMENTS dans cet ordre précis : adresse email du destinataire, le sujet, le message, adresse email de l'expéditeur.

Bien entendu les adresses email doivent être valides, l'idéal est de privilégier gmail pour sa rapidité et sa flexibilité au niveau de la reception d'emails.

Si vous tester ce code en ligne (sur un hébergeur et une page web), ce code d'1 seule ligne fonctionnera instantanément.

Si vous êtes sous WAMP en localhost, vous devrez configurer légèrement votre fichier php.ini.

Pour cela, rendez-vous dans wamp > php > php.ini :



Effectuer au clavier : **Ctrl+F** (édition>rechercher)

Rechercher la chaîne : "smtp".

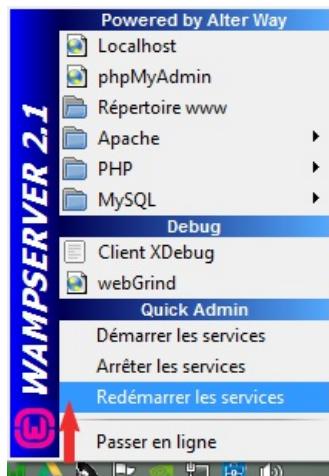
Modifier la ligne smtp=localhost par votre SMTP.

Dans mon cas (j'ai la freebox), cela donnera : smtp=smtp.free.fr

Ensuite, vous pouvez effectuer un **Ctrl+S** (fichier>enregistrer)

Et pour finir : **Alt+F4** (fermeture)

Une fois que vous avez modifié un fichier de configuration, pensez à redémarrer les services de wamp en cliquant sur l'icône "Redémarrer les services / Restart All services" :



Si vous avez un antivirus (firewall), il peut bloquer le port 25 et donc l'envoi d'email, il faudra peut être le configurer ou le désactiver le temps des tests (pareil pour le parefeu).

Si la configuration est difficile, n'hésitez pas à faire les tests directement sur internet (pour cela il faut posséder un hébergement).

Nous avons vu précédemment que la fonction `mail()` permettait d'envoyer un email.

Plutôt qu'avoir des valeurs fixes, nous pouvons gérer cela avec un formulaire d'envoi d'email ou un formulaire de contact (très répandu sur la plupart des sites).

Prenons ce formulaire HTML :

```
email1.php
```

```
<form method="post" action="">
<label for="destinataire">Destinataire</label><br>
<input type="text" name="destinataire" id="destinataire" placeholder="destinataire"><br><br>
<label for="expediteur">Expediteur</label><br>
<input type="text" name="expediteur" id="expediteur" placeholder="expediteur"><br><br>
<label for="sujet">Sujet</label><br>
<input type="text" name="sujet" id="sujet" placeholder="sujet"><br><br>
<label for="message">Message</label><br>
<textarea name="message" placeholder="message"></textarea><br><br>
<input type="submit" value="envoyer l'email">
</form>
```

Ajoutons le code PHP :

```
email1.php
```

```
<?php
if(empty($_POST))
{
    // affichage des saisies pour être sur de les obtenir avant de les exploiter.
    echo "destinataire : $_POST[destinataire] <br>";
    echo "sujet : $_POST[sujet] <br>";
    echo "message : $_POST[message] <br>";
    echo "expediteur : $_POST[expediteur] <br>";

    // entête email
    $headers = 'MIME-Version: 1.0' . "\n";
    $headers .= 'Content-type: text/html; charset=ISO-8859-1' . "\n";
    $headers .= 'Reply-To: ' . $_POST['expediteur'] . "\n";
    $headers .= 'From: "' . ucfirst(substr($_POST['expediteur'], 0, strpos($_POST['expediteur'], '@'))) . "<$_POST[expediteur]>' . "\n";
    $headers .= 'Delivered-to: ' . $_POST['destinataire'] . "\n";

    mail($_POST['destinataire'], $_POST['sujet'], $_POST['message'], $headers);
}
?>
<form method="post" action="">
<label for="destinataire">Destinataire</label><br>
<input type="text" name="destinataire" id="destinataire" placeholder="destinataire"><br><br>
<label for="expediteur">Expediteur</label><br>
```

```

<input type="text" name="expediteur" id="expediteur" placeholder="expéditeur"><br><br>
<label for="sujet">Sujet</label><br>
<input type="text" name="sujet" id="sujet" placeholder="sujet"><br><br>
<label for="message">Message</label><br>
<textarea name="message" placeholder="message"></textarea><br><br>
<input type="submit" value="envoyer l'email">
</form>

```

Dans cet exemple, nous prévoyons une condition IF permettant de dire "si l'internaute à posté".

Quelques instructions echo sont présentes afin d'avoir un affichage et ainsi être certain que les saisies soient bien récupérées.

Ensuite, par l'intermédiaire de la variable \$headers une partie est en place pour définir les entêtes du mail :

**MIME-Version** (Multipurpose Internet Mail Extensions). Il s'agit d'un standard.

Seule la version 1 existe, cette ligne n'est donc pas destinée à changer.

**Content-type** permet de définir l'encodage et le format html.

**Reply-To** permet d'indiquer une adresse email de réponse.

**From** permet de préciser le nom de l'expéditeur et l'adresse email de l'expéditeur

Cela pourrait être géré avec un champ prenom et nom dans le formulaire html, dans notre cas nous avons préféré isoler le pseudo de l'expéditeur (avant le caractère '@') avec la combinaison des fonctions substr (coupe une chaîne), strpos (donne la position d'un caractère dans une chaîne) et ucfirst (met la 1ère lettre en majuscule).

**Delivered-to** indique l'adresse du destinataire à qui l'email est adressé.

Ensuite, c'est bien la fonction mail qui permet d'envoyer le message :

1er argument : adresse email du destinataire.

2e argument : sujet / objet.

3e argument : message.

4e argument : adresse email de l'expéditeur avec les entêtes.

Nous utiliserons \$headers (plus complet en informations et qui contient \$\_POST['expediteur']) plutôt que \$\_POST['expediteur'] tout seul (contenant seulement l'adresse email de l'expéditeur).

Faites le test !

Vous pouvez aussi saisir du html :

```
<div style="background: #f80c43;">Bonjour!</div>
```

Résultat

Destinataire  
votreadresse@gmail.c

Expéditeur  
perenoel@noel.com

Sujet  
Tes cadeaux de noel !

Message

Salut comment ça va ?  
 Je viens récupérer ta liste de  
 cadeaux pour noël.  
 Réponds moi vite et gros bisous !

envoyer l'email

Gmail ▾

Actualiser Plus ▾

1–50 sur 5 713 < > Fr ▾

**NOUVEAU MESSAGE**

Boîte de réception (1) Messages suivis

Perenoel Tes cadeaux de noel ! - Salut comment ça va ? Je viens récupérer ta liste de cad 19:44

Gmail ▾

Retour Archiver Spam Supprimer Déplacer vers ▾ Libellés ▾ Plus ▾ 1 sur 5 713

**NOUVEAU MESSAGE**

Boîte de réception

Messages suivis

Messages envoyés

Brouillons (1)

Spam

Corbeille

compta@evogue.fr

equipe@evogue.fr

liamtardieu@evogue.fr

peggytardieu@gmail.com

Plus ▾

**Tes cadeaux de noel !** Boîte de réception x

Perenoel perenoel@noel.com via free.fr 19:44 (Il y a 1 minute) ★ Répondre ▾

À moi ▾

Salut comment ça va ? Je viens récupérer ta liste de cadeaux pour noël. Réponds moi vite et gros bisous !

Cliquez ici pour Répondre ou pour Transférer le message.

8,83 Go (58 %) utilisés sur 15 Go Gérer Conditions d'utilisation - Confidentialité Dernière activité sur le compte : Il y a 39 minutes Détails

Et oui, vous pouvez prendre l'adresse email de nimporte qui. On peut toujours rêver !

*Mais ce n'est pas le but, restez sage, pas d'usurpation d'identité sauf si c'est pour rire que ça ne porte préjudice à personne ;)*

### Les entêtes complet d'un email

En théorie, pour envoyer un email de qualité (et mettre toutes les chances de votre côté pour que celui-ci ne soit pas catégorisé en spam chez vos internautes), vous devrez communiquer avec un serveur SMTP en lui fournissant certaines informations.

Voici un tableau récapitulatif :

Entête	Description
Subject	Le sujet/objet du message.
From	Adresse email de l'auteur du message.
Sender	Adresse email de l'expéditeur (souvent la même que From, sauf si l'envoyeur est différent de celui qui a écrit le message).
Reply-To	Adresse email de réponse au message (pratique pour communiquer avec un site web lors d'envoi de mail automatique).
To	Listing des adresse(s) email(s) des destinataire(s) direct du message (un même email peut être envoyé à plusieurs personnes).
Cc Carbon copy	Listing des adresse(s) email(s) des destinataire(s) en copie du message.
Bcc Blind carbon copy	Listing des adresse(s) email(s) des destinataire(s) en copie invisible du message.
Message-ID	Code unique d'identification du message.
In-Reply-To	Représente l'id du message (pratique pour grouper des échanges d'emails et en faire des conversations)
References	Représente l'id de la conversation (lors d'échanges emails).
Comments	Commentaires éventuels en lien avec le message.
X-Mailer	Messagerie ou Logiciel émetteur du message.
Keywords	Mots-clés devrant le message.
Date	Date et heure d'expédition.
MIME-Version	Version MIME du mail (seule la version 1.0 existe).

<b>Priority</b>	Niveau de priorité du mail.
<b>Content-Type</b>	Type de message (par exemple : html ou texte).
<b>Content-transfer-encoding</b>	Encodage du message.
<b>Content-Description</b>	Description du message.

### Créer un formulaire de contact et envoyer un email en PHP pour son site web

Si vous souhaitez être la seule personne à recevoir des emails, il suffit de retirer la case destinataire du formulaire HTML et adapter légèrement le code PHP en conséquence :

Ajoutons le code PHP :

```
email2.php

<?php
if(empty($_POST))
{
    // affichage des saisies pour être sur de les obtenir avant de les exploiter.
    echo "sujet : $_POST[sujet] <br>";
    echo "message : $_POST[message] <br>";
    echo "expéditeur : $_POST[expéditeur] <br>";

    // entête email
    $headers = 'MIME-Version: 1.0' . "\n";
    $headers .= 'Content-type: text/html; charset=ISO-8859-1' . "\n";
    $headers .= 'Reply-To: ' . $_POST[expéditeur] . "\n";
    $headers .= 'From: "' . ucfirst(substr($_POST[expéditeur], 0, strpos($_POST[expéditeur], '@'))) . "<$_POST[expéditeur]>" . "\n";
    $headers .= 'Delivered-to: monadresse@gmail.com' . "\n";
    mail("monadresse@gmail.com", $_POST['sujet'], $_POST['message'], $headers);
}
?>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="expéditeur">Expéditeur</label><br>
    <input type="text" name="expéditeur" id="expéditeur" placeholder="expéditeur"><br><br>
    <label for="sujet">Sujet</label><br>
    <input type="text" name="sujet" id="sujet" placeholder="sujet"><br><br>
    <label for="message">Message</label><br>
    <textarea name="message" placeholder="message"></textarea><br><br>
    <input type="submit" value="envoyer l'email">
</form>
```

#### Résultat

**Expéditeur**

**Sujet**

**Message**

Nous n'utilisons pas le `$_POST['destinataire']`, puisque ce sera toujours vous, nous avons retiré la case de saisie du formulaire HTML.

Ceci est mieux que l'utilisation d'un `mailto()` dans la mesure où votre adresse (dans le code php exécuté côté serveur) ne sera pas visible par les robots et vous ne risquerez pas d'être spammé !

### Créer un formulaire de contact et envoyer un email en PHP avec plusieurs informations

Si vous souhaitez ajouter récolter d'autres informations sur l'internaute, il suffit d'ajouter d'autres champs de formulaire HTML et de prévoir le code de

récupération en PHP :

Prenons ce code :

```
email3.php

<?php
if(!empty($_POST))
{
    // affichage des saisies pour être sur de les obtenir avant de les exploiter.
    echo "expéditeur : $_POST[expéditeur] <br>";
    echo "nom : $_POST[nom] <br>";
    echo "prénom : $_POST[prénom] <br>";
    echo "société : $_POST[société] <br>";
    echo "sujet : $_POST[sujet] <br>";
    echo "message : $_POST[message] <br>";

    // entête email
    $headers = 'MIME-Version: 1.0' . "\n";
    $headers .= 'Content-type: text/html; charset=ISO-8859-1' . "\n";
    $headers .= 'Reply-To: ' . $_POST['expéditeur'] . "\n";
    $headers .= 'From: "' . ucfirst(substr($_POST['expéditeur'], 0, strpos($_POST['expéditeur'], '@'))) . "<$_POST['expéditeur'].>'" . "\n";
    $headers .= 'Delivered-to: monadresse@gmail.com' . "\n";

    $message = "Nom : " . $_POST[nom] . "\nPrénom : " . $_POST[prénom] . "\nSociété : " . $_POST[société] . "\nMessage : " .
    $_POST[message];

    mail("monadresse@gmail.com", $_POST['sujet'], $message, $headers);
}
?>
<form method="post" action=">">
<label for="nom">Nom</label><br>
<input name="nom" id="nom" placeholder="nom" type="text"><br><br>
<label for="prénom">Prénom</label><br>
<input name="prénom" id="prénom" placeholder="prénom" type="text"><br><br>
<label for="société">Société</label><br>
<input name="société" id="société" placeholder="société" type="text"><br><br>
<label for="expéditeur">Expéditeur</label><br>
<input type="text" name="expéditeur" id="expéditeur" placeholder="expéditeur"><br><br>
<label for="sujet">Sujet</label><br>
<input type="text" name="sujet" id="sujet" placeholder="sujet"><br><br>
<label for="message">Message</label><br>
<textarea name="message" placeholder="message"></textarea><br><br>
<input type="submit" value="envoyer l'email">
</form>
```

## Résultat

nom : Mon Nom  
prénom : Mon Prénom  
société : Ma Société  
expéditeur : expediteur@site.com  
sujet : le sujet  
message : le message  
Nom  
Mon Nom

Prénom  
Mon Prénom

Société  
Ma Société

Expéditeur  
expediteur@site.com

Sujet  
le sujet

Message  
le message

envoyer l'email



➤ **Attention**

Nous ne pouvons pas ajouter plus d'arguments à la fonction mail() et par conséquent nous ne pouvons pas lui passer tous les \$\_POST de notre formulaire.

Notre code permet de mettre toutes les informations à l'intérieur de \$\_POST['message']. Et puisqu'on le redéfinit, il ne faut pas oublier de ré-affecter le message lui même.

## Les cookies

### Objectifs

- ✓ Conserver et échanger des informations avec un internaute.

93

## Découverte des cookies

Qu'est-ce qu'un cookie ? Un cookie est un fichier sauvegardé sur l'ordinateur de l'internaute avec des informations à l'intérieur.

Les informations à l'intérieur d'un cookie ne sont pas sensibles (pas de mot de passe), il s'agit en général de préférence, exemple : langue dans laquelle l'internaute souhaite visiter le site, derniers produits consultés dans une boutique (cela permettra de relancer l'internaute sur d'autres produits durant ses prochaines navigations, voir le remarketing en référencement), etc.

Depuis quelques années, les sites web Européens ont l'obligation d'informer les internautes lorsque des informations liées à leur navigation et utilisation sont retenues dans un fichier cookie.

[Yahoo \(exemple\) : cliquez ici pour connaître le genre d'informations retenues sur vous et aussi leur utilisation](#)

Pour créer un cookie, cela se fait avec la fonction prédéfinie de PHP setCookie :

```
cookie1.php
??
<?php
setCookie("nomCookie","contenuCookie",time()+31536000);
```

### Explications

setCookie est donc une fonction prédéfinie permettant de créer un fichier (un fichier cookie) sur l'ordinateur de l'internaute pour sauvegarder des informations.

Les arguments attendus sont :

- 1. Le nom du cookie, dans notre cas nous l'avons nommé "nomCookie".
- 2. Le contenu du cookie (les informations), ici nous avons inscrit "contenuCookie".
- 3. La date d'expiration du cookie (péremption), dans l'exemple ci-dessus nous avons pris le moment actuel (par l'intermédiaire de la fonction time(), timestamp) et avons ajouté le nombre de secondes écoulées en 1 année, soit : 31536000 (secondes)

*A noter, un internaute peut supprimer 1 cookie à tout moment (puisque le fichier se trouve sur son ordinateur)!*

Qu'est-ce que la fonction time() en PHP :

La fonction PHP prédéfinie time() permet de donner le timestamp, cela représente le nombre de secondes écoulées entre le 01 janvier 1970 (date clé en informatique) et maintenant (le moment présent).

Plus d'informations ici : [Wikipedia TimeStamp](#)

Pour voir le timestamp, essayons ce code :

```
echo time();
```

Rechargez la page plusieurs fois (**F5** ou **Ctrl+R**), vous verrez le résultat changer en conséquence (puisque à chaque instant des secondes s'écoulent).

## Résultat

1459185195

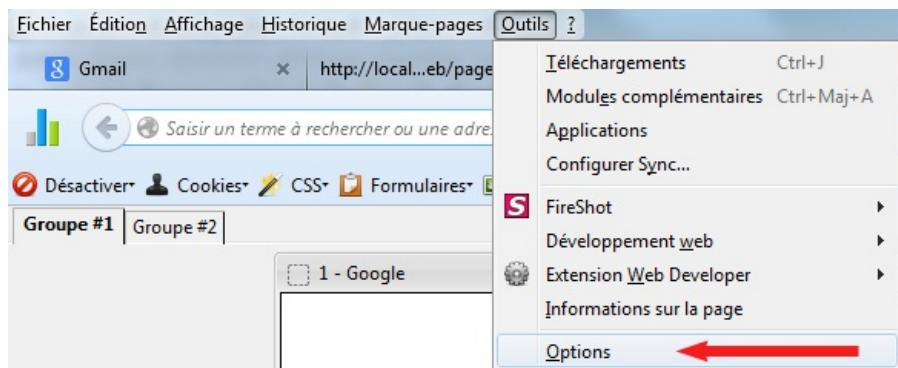
Avant de nous intéresser et d'observer les cookies sur notre ordinateur, je vous invite à faire du ménage en supprimant tous vos cookies, voici le raccourci clavier : **Ctrl+Shift+Suppr** (attention vous serez déconnecté des différents sites web où vous aviez enregistré votre connexion).

Reprenons notre code :

```
cookie1.php
```

```
<?php  
setCookie("nomCookie","contenuCookie",time()+31536000);
```

Exécuter la page `cookie1.php` et ensuite pour voir le cookie apparaître (sous le navigateur mozilla firefox), vous pouvez vous rendre dans la partie suivante : Outils > Options > Vie privée > Règle de conservation : Utiliser les paramètres personnalisés pour l'historique > Afficher les cookies.

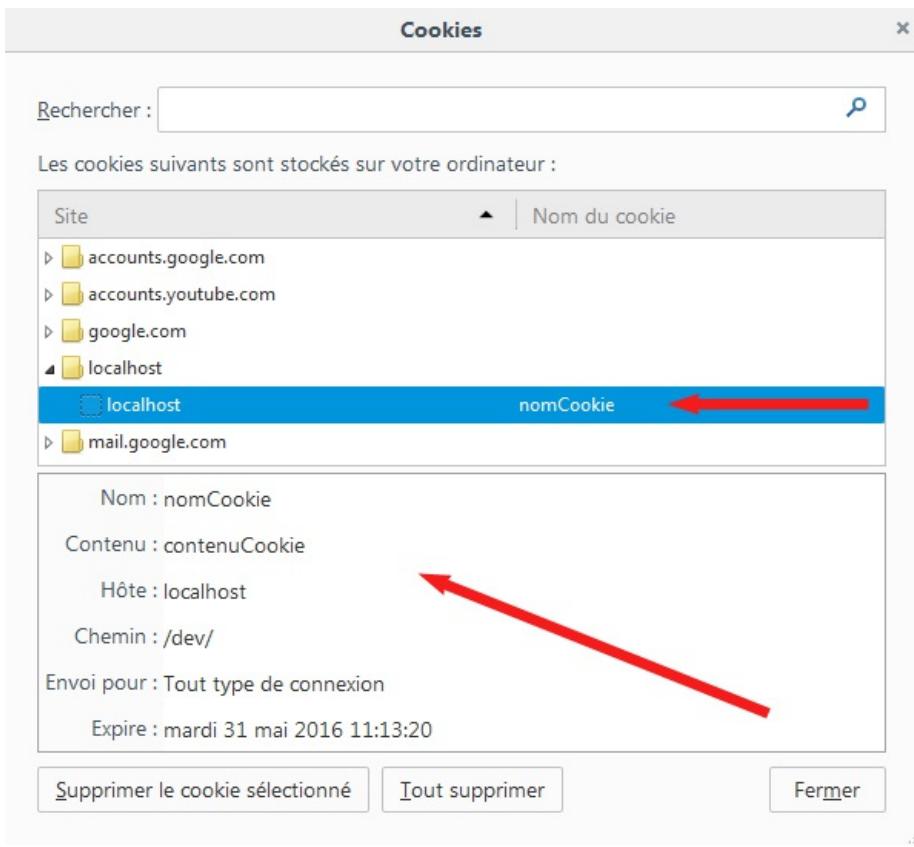


Vie privée

**Pistage**  
 Indiquer aux sites que je ne souhaite pas être pisté  
[En savoir plus](#)

**Historique**  
Règles de conservation : ▼  
 Toujours utiliser le   
 Conserver l'historique  
 Ne jamais conserver l'historique  
 Utiliser les paramètres personnalisés pour l'historique  
 Accepter les cookies  
Accepter les cookies tiers : ▼  
Les conserver jusqu'à : ▼  
 Vider l'historique lors de la fermeture de Firefox

[Exceptions...](#) [Afficher les cookies...](#) [Paramètres...](#)



Après avoir exécuté dans le navigateur le fichier `cookie1.php`. Nous pouvons apercevoir notre cookie et son contenu.

`setCookie()` est une fonction prédéfinie permettant de créer un cookie, cependant il n'y a pas de fonction prédéfinie permettant de le supprimer. Pour rendre inactif un cookie, on le met généralement à jour avec une date périmée.

94

## Cas pratique : une page multilingue avec des cookies

Pour mieux se représenter à quoi peuvent nous servir les cookies dans le cadre d'un site web, je vous propose de créer une page multilingue. Ce sera plus concret !

Je vous invite à créer le début de l'exemple vous même (sous forme d'exercice), nous ferons la partie cookie ensemble.

Voici les consignes :

Exercice :

1. Créer 1 fichier nommé `langue.php`
2. Créer 4 liens HTML pointant vers la même page (soit un `href` sur `langue.php`)  
Les liens seront : French - English - Italy - Spain (vous pouvez aussi ajouter des drapeaux en image)
3. Lorsque l'on clique sur l'un des liens HTML, nous n'irons nulle part et devrons rester sur la même page (soit `langue.php`), cependant il faudra transmettre une information dans l'url du type : `?pays=fr` (pour french), `?pays=en` (pour english), `?pays=it` (pour Italy), `?pays=es` (pour Spain)
4. Ensuite, l'idéal serait de récupérer l'argument passé dans l'url (via `$_GET`) et de l'afficher sur la page web.  
Si l'internaute clique sur French on affichera fr, si l'internaute clique English on affichera en, si l'internaute clique Italy on affichera it, et enfin si l'internaute clique Spain on affichera es. Toujours dans la même page web !

Voici la correction :

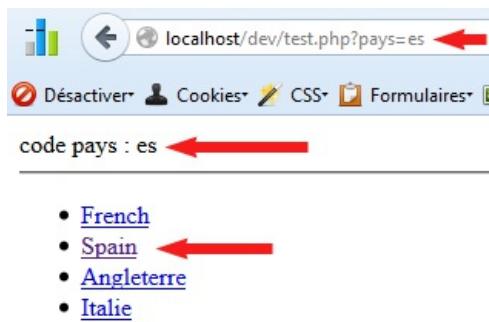
```
<?php
if(empty($_GET['pays']))
{
    echo 'code pays : ' . $_GET['pays'];
}
?>

<ul>
<li><a href="?pays=fr"> French </a></li>
<li><a href="?pays=es"> Spain </a></li>
<li><a href="?pays=an"> Angleterre </a></li>
<li><a href="?pays=it"> Italie </a></li>
</ul>
```

Tant qu'il s'agit d'un entraînement (et tant que notre page n'est pas en ligne, sur le web), nous ne sommes pas obligé de mettre les balises d'en-tête doctype etc.

## Résultat

Exemple avec un clic sur "Spain" pour la langue espagnol :

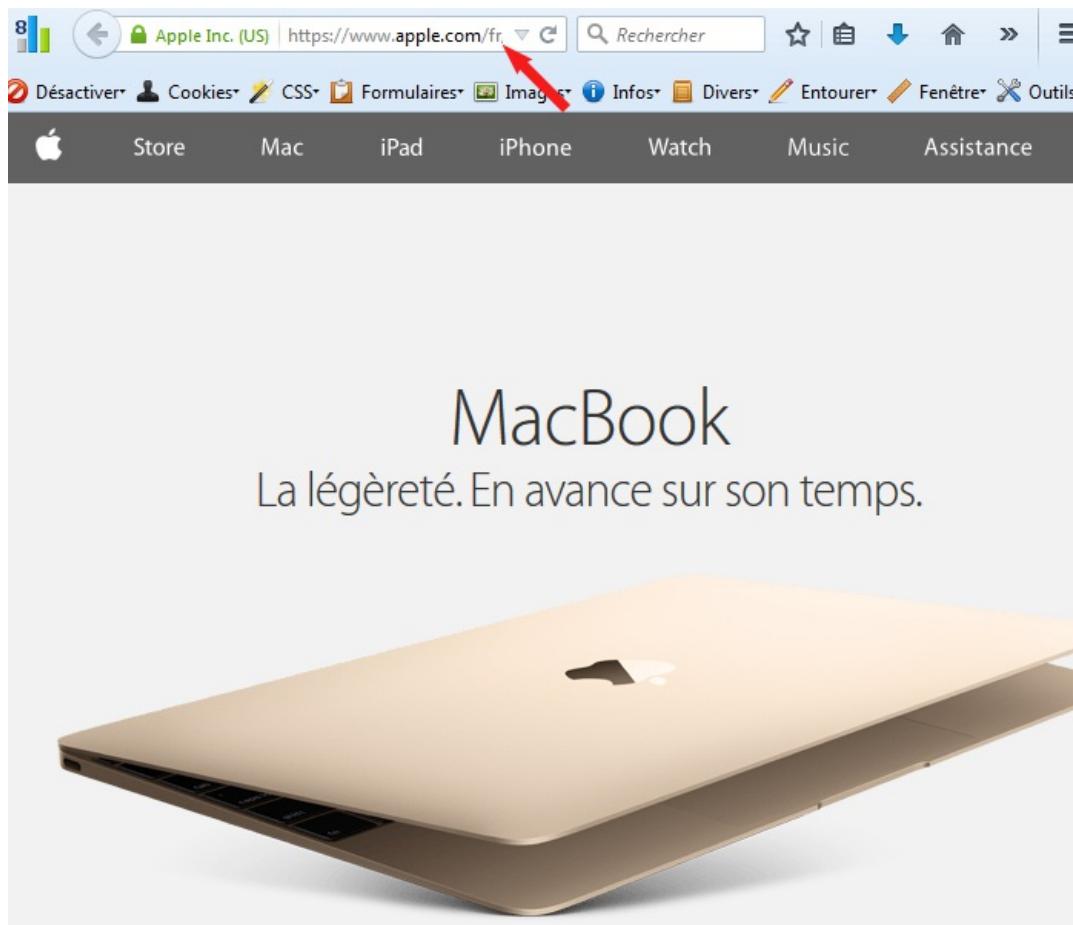


Si vous cliquez sur "Spain", vous verrez afficher le code pays es, cela fonctionne très bien mais votre préférence (dans le choix de la langue espagnole) ne sera pas gardée si vous partez en fermant la page et que vous revenez sur le site web plus tard.

Si ce code n'est pas clair pour vous, il faudra prendre le temps de relire le chapitre sur le [passage d'argument dans une url et la récupération avec la superglobale \\$\\_GET](#).

Plusieurs sites fonctionnent comme ça ! avec un argument dans l'url.

Cliquez ici et regardez bien la fin de l'url le site d'Apple (pour exemple) : [Apple FR](#) - [Apple IT](#)



Adaptons et modifions un peu le code pour sauvegarder la langue de l'internaute dans un cookie :

```

langue.php

<?php
if(isset($_GET['pays']))
{
    $pays=$_GET['pays'];
}
elseif(isset($_COOKIE['pays']))
{
    $pays=$_COOKIE['pays'];
}
else
{
    $pays='fr';
}

$expiration = 365*24*3600;
setCookie("pays",$pays,time()+$expiration);
?>

<!Doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>Page Multilingue</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <ul>
        <li><a href="?pays=fr"> French </a></li>
        <li><a href="?pays=es"> Spain </a></li>
        <li><a href="?pays=an"> Angleterre </a></li>
        <li><a href="?pays=it"> Italie </a></li>
    </ul><br>
    <?php
    switch($pays)
    {
        case 'fr':
            echo '<p>Bonjour, <br> Vous visitez actuellement le site en français <br>Effectivement, la sauvegarde du cookie permet de retenir la
langue avec laquelle vous naviguez sur le site pour vos prochaines visites. <br>A bientôt.</p>';
            break;
        case 'es':
            echo '<p>¡Hola! En este momento estás visitando el sitio en español <br>De hecho, la cookie permite la copia de seguridad de
conservar el idioma que utilizas para futuras visitas. <br>Pronto.</p>';
            break;
        case 'en':
            echo '<p>Hello <br>You are currently visiting the site in English <br>Indeed, the cookie allows the backup to retain the language that you
use the site for future visits. <br>Soon.</p>';
            break;
        case 'it':
    }
    </?php

```

```

echo '<p>Ciao <br>Si sta attualmente visitando il sito in Italiano <br>Infatti, il cookie consente il backup di mantenere la lingua che si
usa il sito per visite future. <br>Presto.</p>';
break;
?
</body>
</html>

```

### Explications et décomposition du code

1ère condition IF :

```

if(isset($_GET['pays']))
{
    $pays=$_GET['pays'];
}

```

Cette condition permet de savoir si un code pays est défini.

En gros : est-ce que l'internaute a cliqué sur l'un des liens pour afficher le site dans une langue particulière ?

Si l'url contient un code pays, c'est donc qu'un lien a été cliqué, nous l'ajoutons à la variable \$pays

2ème condition ELSEIF :

```

elseif(isset($_COOKIE['pays']))
{
    $pays = $_COOKIE['pays'];
}

```

Elseif = Sinon si, un cookie nommé "pays" existe sur l'ordinateur de l'internaute.

Nous le récupérons et affectons la variable \$pays avec.

Cette condition s'exécutera uniquement si nous ne sommes pas rentrés dans le IF précédent et si l'internaute est déjà venu sur le site (sinon le cookie n'existerait pas).

3ème condition ELSE :

```

else
{
    $pays = 'en';
}

```

Else = Sinon, dans le scénario où le if (pas de clic sur 1 lien de la part de l'internaute) et le elseif (pas de cookie) ne se déclenchent pas, le cas par défaut sera appliqué.

Nous mettons le code pays "en" dans la variable \$pays

Ce cette manière, le site s'affichera par défaut en Anglais.

*Cette condition (par défaut) s'exécutera uniquement si nous ne sommes pas rentrés dans le IF, ni dans le ELSEIF précédent.*

Nous ressortirons forcément de cette série de conditions IF / ELSEIF / ELSE avec la variable \$pays affectée par un code pays:

Affichage de \$pays :

```
echo $pays;
```

Si vous souhaitez avoir connaissance du code pays dans la variable \$pays, rien ne vous empêche d'écrire la ligne ci-dessus (le temps des tests).

Cela affichera le contenu de la variable \$pays

Création / Mise à jour du cookie :

```

$expiration = 365*24*3600;
setCookie("pays",$pays,time()+$expiration);

```

\$expiration nous permet de conserver le calcul d'1 année en secondes (365 jours x 24 heures x 3600 vient de 60sec x 60min nombre de seconde dans 1 heure).

setCookie nous permet de déposer le fichier cookie sur l'ordinateur de l'internaute.

Argument 1 - Nom du cookie : pays

Argument 2 - Valeur du cookie : "fr" ou "en" ou "it" ou "es"

Argument 3 - Date d'expiration : dans 1 an à partir d'aujourd'hui

Puisque ce code ne se trouve pas dans une condition, nous créerons (dans tous les cas) un cookie (sur l'ordinateur de l'internaute) avec le code pays contenu dans la variable \$pays.

Un cookie sera valable 1 an après la dernière visite de l'internaute (l'internaute qui se connecte tous les mois verra son choix gardé à l'infini sans problème d'affichage de texte pour la langue, puisqu'à chacun de ses passages son cookie est relancé pour 1 année à partir de sa dernière visite).

#### Condition Switch :

```
switch($pays)
{
    case 'fr':
        echo '<p>Bonjour, <br> Vous visitez actuellement le site en français <br> Effectivement, la sauvegarde du cookie permet de retenir la langue avec laquelle vous naviguez sur le site pour vos prochaines visites. <br> A bientôt.</p>';
        break;
    case 'es':
        echo '<p>Hola <br> En este momento está visitando el sitio en español <br> De hecho, la cookie permite la copia de seguridad de conservar el idioma que utilice el sitio para futuras visitas. <br> Pronto.</p>';
        break;
    case 'en':
        echo '<p>Hello <br> You are currently visiting the site in English <br> Indeed, the cookie allows the backup to retain the language that you use the site for future visits. <br> Soon.</p>';
        break;
    case 'it':
        echo '<p>Ciao <br> Si sta attualmente visitando il sito in Italiano <br> Infatti, il cookie consente il backup di mantenere la lingua che si usa il sito per visite future. <br> Presto.</p>';
        break;
}
```

La condition switch test la variable \$pays et permet de faire sortir le bon texte à afficher en fonction de la langue de l'internaute.

Faites les tests !

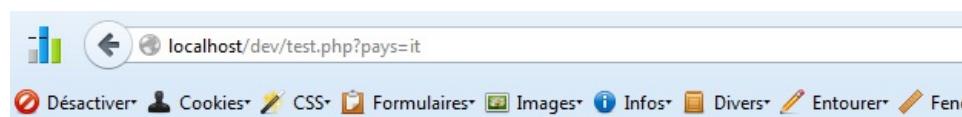
Résultat et tests ! Etape 1 : Cliquez sur le lien "Italy", le texte s'affiche en italien.

Etape 2 : Fermer votre navigateur (ou onglet).

Etape 3 : Rouvrez votre navigateur (ou onglet) et allez sur la page web langue.php (sans argument pays dans l'url).

Etape 4 : Le texte reste affiché en italien. Ca fonctionne ! 1 fichier cookie a bien été créé sur l'ordinateur de l'internaute et a été pris en compte.

Etape 5 : Observer le cookie, dans firefox : outils > option > Vie Privée > Conserver les paramètres personnalisés pour l'historique > Afficher les cookies > localhost > pays : voir les détails affichés.



- [French](#)
- [Spain](#)
- [Angleterre](#)
- [Italie](#)

Ciao

Si sta attualmente visitando il sito in Italiano

Infatti, il cookie consente il backup di mantenere la lingua che si usa il sito per visite future.

Presto.

**Cookies**

Rechercher :

Les cookies suivants sont stockés sur votre ordinateur :

Site	Nom du cookie
evogue.fr	
fredericdemarquet.com	
google.com	
localhost	
localhost	pays
mail.google.com	

Détails du cookie sélectionné :

Nom : pays  
Contenu : it  
Hôte : localhost  
Chemin : /dev/  
Envoi pour : Tout type de connexion  
Expire : mardi 31 mai 2016 13:37:27

[Supprimer le cookie sélectionné](#) [Tout supprimer](#) [Fermer](#)

*Si votre test ne fonctionne pas, pensez à vérifier le code (voir s'il n'y a pas d'erreur) et aussi regardez si l'option "vider l'historique lors de la fermeture de firefox" n'est pas cochée, ce qui empêcherait de garder les cookies.*

L'anglais est choisi comme langue par défaut, mais il aurait par exemple été possible de détecter la langue de l'internaute par l'utilisation de la superglobale `$_SERVER` afin d'adapter l'affichage de la page web en conséquence.

#### Quelques Questions :

Question 1 : Est-ce possible de tomber dans plusieurs conditions (if/else/elseif) en même temps ?

Réponse 1 : Non.

Question 2 : Dans quelle condition rentre l'internaute lors de sa toute 1ère visite ?

Réponse 2 : else et donc l'affichage du site par défaut est en Anglais

Question 3 : Dans quelle condition l'internaute rentre lorsqu'il clique sur un lien ?

Réponse 3 : if

Question 4 : Dans quelle condition l'internaute rentre lorsqu'il revient sur le site 15 jours plus tard ?

Réponse 4 : elseif, car un cookie existe

Question 4 : Dans quelle condition rentre l'internaute lorsqu'il revient sur le site 15 jours plus tard après avoir cliqué sur 1 lien ?

Réponse 4 : if, car même si un cookie existe, nous vérifions d'abord si l'internaute a cliqué sur 1 lien (dans l'ordre du code) et si nous rentrons dans le if, nous ne rentrerons donc pas dans le elseif.

Question 5 : Lors de la toute 1ère visite d'un internaute, est-ce qu'un cookie est créé ?

Réponse 5 : oui, à chaque fois, le `setCookie()` ne se trouve pas dans une condition et est toujours appliqué.

Question 6 : Lors des visites suivantes, que se passe t'il ? et que se passe t'il si l'internaute change de langue ?

Réponse 6 : Le cookie est mis à jour sur la durée de vie. Si la langue est changée, le cookie est mis à jour sur la durée de vie et le contenu.

Question 7 : Est-ce qu'on pourra mettre les informations (produits, prix, etc) d'un panier dans un cookie ?

Réponse 7 : Surtout pas ! l'internaute pourra modifier le prix, ça sera son fichier texte, étant sur son pc cela lui appartient.

Nous garderons donc l'utilisation des fichiers cookies pour des préférences mineures

## Les sessions

### Objectifs

- ✓ Conserver et échanger des informations avec un internaute.

95

### Qu'est-ce qu'une session ?

Une session est un système mis en oeuvre dans le code PHP permettant de conserver sur le serveur, dans un fichier temporaire, des informations relatives à un internaute.

L'avantage d'une session c'est que les données seront enregistrées dans un fichier sur le serveur disponible et consultable par toutes les pages durant toute la navigation de l'internaute.

96

### Contexte : Quels sont les cas d'utilisation ?

Habituellement pour sauvegarder des données, nous enregistrons des informations dans une base.

Mais nous ne devons pas avoir toujours recours à une base de données, parfois il est utile de sauvegarder des informations dans 1 fichier (1 fichier de session).

#### Exemple pour un panier :

Nous ferons le panier d'un site ecommerce avec un système de session.

Nous n'enregistrerons pas les produits ajoutés au panier dans une base de données puisque la plupart du temps les paniers ne sont pas payés (les internautes ne finalisent pas toujours la commande). Cela évitera de "polluer" la base de données pour rien. En revanche, (si le panier et le paiement sont validés) la commande sera enregistrée dans une base de données.

#### Exemple avec la connexion d'un internaute (membre) :

➤ Lorsqu'un internaute s'inscrit à 1 site, nous l'enregistrons dans une base de données.

➤ Lorsqu'il se connecte (avec le bon pseudo et le bon mot de passe), nous gardons l'information (comme quoi l'internaute est actuellement connecté) en mémoire dans 1 fichier de session.

En effet, nous n'allons pas solliciter une éventuelle table de la base "internaute\_actuellement\_connecte" car cela change souvent et les requêtes sql pour cette tâche allourdiraient le site web.

#### Plus généralement, voici des cas d'utilisation :

- La connexion d'un membre à un site web.
- Le panier d'un site ecommerce (vente en ligne).
- Les formulaires à plusieurs étapes.
- D'autres informations générales. Cela peut être marketing (derniers produits vus, suggestions de produits adaptés aux préférences de l'internaute selon sa navigation), etc.

Pour conclure, nous utiliserons les sessions pour sauvegarder des informations temporaires et garderons l'utilisation de base de données pour sauvegarder des informations durables.

97

### A quoi ça sert ?

Comme vous l'aurez compris, les sessions en PHP vont nous permettre de conserver des informations sur un internaute côté serveur.

Sans ce mécanisme, vous ne pourriez pas connecter un internaute à votre site et maintenir sa connexion durant la navigation (de page en page).

De la même manière, lorsqu'un internaute ajoute des produits dans un panier (ecommerce), comment se fait-il que les produits soient toujours présents en mémoire même lorsque vous naviguez de page en page (pour partir et revenir du panier), ce n'est pas magique, là aussi ce sont grâce aux sessions !

1 page (ou fichier) représente 1 script qui s'exécute une fois, puis l'internaute se déplace dans le site pour se rendre vers un autre contenu (ce qui exécute 1 autre script).

Pour ne pas que les informations se perdent d'une page à un autre (d'un fichier script à un autre), nous utiliserons les sessions !

98

## Comment cela fonctionne ?

2 besoins reviennent régulièrement :

- Créer 1 fichier de session pour une première utilisation
- Lire 1 fichier de session existant pour une seconde (ou énième) utilisation.

La bonne nouvelle c'est que pour ces 2 actions nous utiliserons la même fonction pré définie de PHP !

Pour créer ou lire un fichier de session, nous utiliserons la fonction pré définie `session_start()`.

```
session1.php  
=?php  
session_start();?
```

Quelques Explications :

`session_start()` permet de créer un fichier de session ou de l'ouvrir s'il existe déjà.

Où se trouve les fichiers de session ?

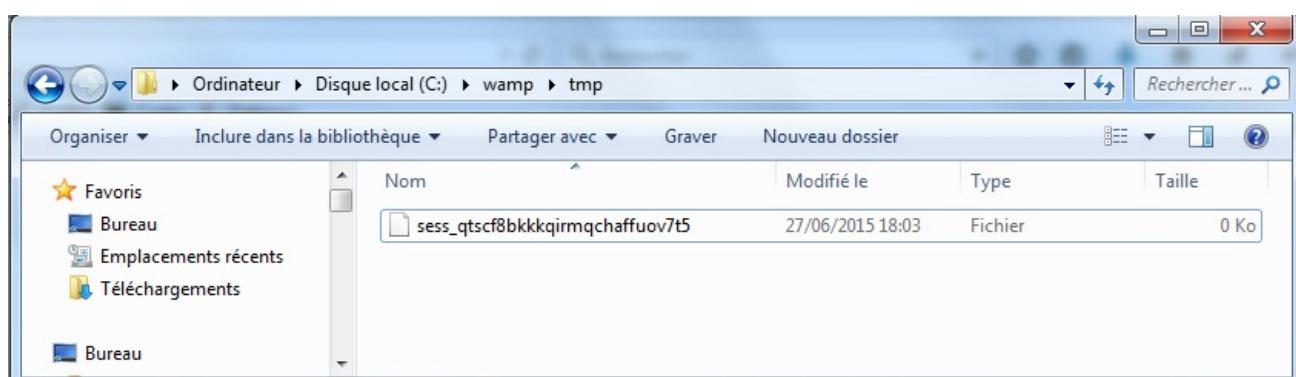
Les fichiers de session se trouvent sur le serveur, dans un dossier nommé `/tmp/` (`tmp` pour temporaire).

Etant donné que nous possédons un serveur web local (sur notre pc), vous pourrez vous rendre à l'emplacement de wamp.

En passant dans "ordinateur", voici des chemins d'exemple : `c:/wamp/tmp/` ou `c:/programmes/wamp/tmp/`

Votre page de code `session1.php` restera vierge mais pas votre dossier `/tmp/` puisque vous devriez y retrouver un fichier temporaire :

Résultat



Nous avons donc 1 fichier temporaire sur le serveur.

Si nous avons plusieurs internautes différents, nous aurons donc plusieurs fichiers de session.

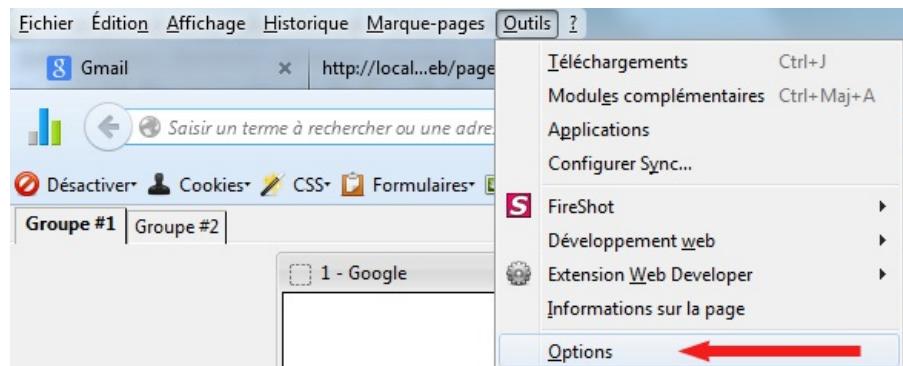
Comment le serveur peut savoir à qui appartient tel ou tel fichier ?

Pour cela, nous allons avoir besoin de créer des cookies, des cookies de session !

Et bonne nouvelle, c'est déjà fait ! Lorsque nous utilisons `session_start()`, cela crée 1 fichier de session (sur le serveur) mais aussi 1 fichier cookie (sur l'ordinateur de l'internaute), comme si nous avions utilisé `setCookie()` !

## Résultat

Exécuter la page `session1.php` et ensuite pour voir le cookie apparaître (sous le navigateur mozilla firefox), vous pouvez vous rendre dans la partie suivante : Outils > Options > Vie privée > Règle de conservation : Utiliser les paramètres personnalisés pour l'historique > Afficher les cookies.



The screenshot shows the 'Vie privée' (Privacy) settings page. On the left is a sidebar with icons for Général, Recherche, Contenu, Applications, Vie privée (selected), Sécurité, Sync, and Avancé. The main area has a heading 'Pistage' with a checkbox for 'Indiquer aux sites que je ne souhaite pas être pisté' and a link 'En savoir plus'. Under 'Historique', there's a dropdown menu set to 'utiliser les paramètres personnalisés pour l'historique'. Other options include 'Toujours utiliser le même historique', 'Conserver l'historique', 'ne jamais conserver l'historique', and 'Accepter les cookies'. The 'Accepter les cookies tiers' dropdown is set to 'toujours'. There are also dropdowns for 'Les conserver jusqu'à:' (set to 'leur expiration') and buttons for 'Exceptions...', 'Afficher les cookies...', and 'Paramètres...'.

**Cookies**

Rechercher :  

Les cookies suivants sont stockés sur votre ordinateur :

Site	Nom du cookie
localhost	PHPSESSID

Nom : PHPSESSID  
Contenu : qtscf8bkkqirmqchaffuov7t5  
Hôte : localhost  
Chemin : /  
Envoi pour : Tout type de connexion  
Expire : À la fin de la session

[Supprimer le cookie sélectionné](#) [Tout supprimer](#) [Fermer](#)

A noter : Le nom du fichier de session correspond au contenu du fichier cookie, dans mon cas : qtscf8bkkqirmqchaffuov7t5

C'est ce qui permet de les relier ensemble !

## coté client (pc de l'internaute)

cookie "pays"
nom : pays
contenu : es
hôte : localhost
chemin : /cookie/
expire : 2014...

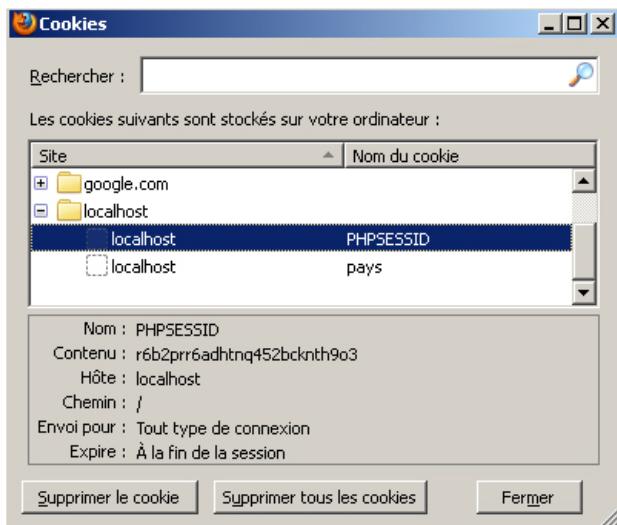
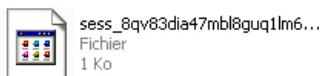
  

cookie "PHPSESSID"
nom : PHPSESSID
contenu : r6b2prr6adhtnq452bcknth9o3
hôte : localhost
chemin : /cookie/
expire : À la fin de la session

## coté serveur (site internet)

session "r6b2prr6adhtnq452bcknth9o3"
prenom : Mathieu
nom : Balmont

C:\Program Files\wamp\tmp



Dans le cas d'un panier (sur un site ecommerce), les informations relatives aux produits seront conservées dans le fichier de session (sur le serveur) et seront reliées à l'internaute grâce au cookie.

Si l'internaute supprime son cookie, cela "casse" le lien avec le fichier de session (le panier sera donc vide, vous pouvez essayer ça sur nimporte quel site web).

De la même manière, ce système (session+cookie) permet de maintenir une connexion à un site web. L'internaute ne peut pas supprimer le fichier de session (puisque'il n'a pas accès au serveur) mais il peut supprimer son cookie (sur son ordinateur) et donc la connexion sera perdue (pour faire le test : n'hésitez pas à supprimer vos cookies, vous perdrez toutes vos connexions en cours).

### Pour résumé :

`session_start()` permet de créer un fichier de session (ou de l'ouvrir s'il existe déjà), mais aussi de créer un cookie (ou s'il existe déjà, de le relier à un fichier de session déjà existant).

Il faut toujours utiliser le code `session_start()` au plus haut de la page et si possible dans les toutes premières lignes !

Pour ne pas obtenir une erreur PHP du type : "Cannot modify header information - headers already sent by ...", ou encore "Cannot send session cookie - headers already sent by ...". Il faut éviter d'inscrire une balise html, un espace, ou une instruction d'affichage du type "echo, print, var\_dump, print\_r", etc. avant `session_start()`.

Maintenant que nous avons obtenu 1 fichier de session, l'objectif est de conserver des données et donc pour cela d'écrire des informations.

Pour cela, une superglobale `$_SESSION` est prévue à cet effet.

Toutes les superglobales sont (par défaut) ARRAY, nous utiliserons donc des crochets pour y inscrire des valeurs.

```
session1.php

<?php
session_start();
$_SESSION['pseudo'] = "Mathieu";
$_SESSION['mdp'] = "Balmont";
print_r($_SESSION);
```

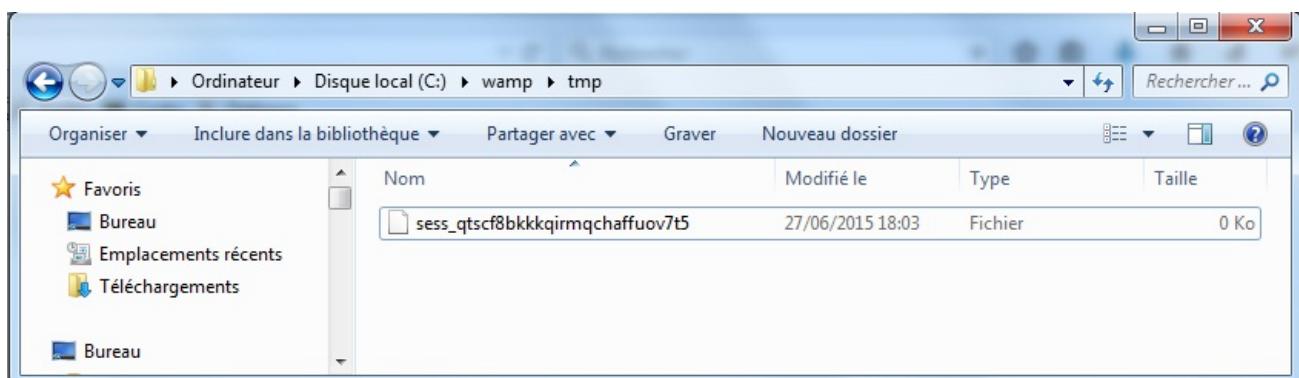
Quelques Explications :

`$_SESSION['pseudo'] = "Mathieu";` La présence des crochets permet d'affecter des valeurs dans le tableau array superglobale `$_SESSION`.

`print_r($_SESSION);` L'instruction `print_r` permet d'afficher le contenu d'un tableau ARRAY et, dans notre cas, d'explorer la superglobale `$_SESSION`.

Retournons dans notre dossier `/tmp/` (présent à l'intérieur de wamp).

#### Résultat



Ouvrons ce fichier (clic droit > éditer avec notepad++) :

`pseudo:s:7:"Mathieu";mdp:s:7:"Balmont";`

**s** correspond à string (chaîne de caractère)

**7** correspond à la taille du string (taille de la chaîne de caractère)

Créons un deuxième fichier, qui n'a aucun rapport, et qui peut être placé dans le même dossier ou ailleurs sur le serveur (prenez l'emplacement de votre choix dans le répertoire www).

Nous nommerons ce fichier : `bonjour.php`

```
bonjour.php

<?php
session_start();
echo 'Bonjour ' . $_SESSION['pseudo'] . '<br>';
```

#### Résultat

Bonjour Mathieu

Ce fichier n'a rien à voir avec l'autre, il n'y a pas d'inclusion, il pourrait être dans un autre dossier, s'appeler nimporte comment, les informations contenues

seraient quand même accessibles ! c'est la puissance et l'intérêt des sessions !

#### Quelques Explications :

Ca fonctionne ! mais ce n'est pas magique, voici l'explication :

Dans notre cas, `session_start();` permet d'ouvrir le fichier de session présent dans le dossier `/tmp/` (il n'est pas créé ou recréé puisqu'un fichier de session relié au cookie de l'internaute existe déjà).

```
echo 'Bonjour ' . $_SESSION['pseudo'] . '
```

' ; Avec ce code, nous affichons le pseudo gardé en mémoire dans le fichier de session.

Nous pouvons le traduire par (`echo`) affiche moi, (bonjour) bonjour, () suivi de..., (`$_SESSION['pseudo']`) le pseudo conservé dans la session

Pour conclure, après avoir utilisé `session_start()` pour travailler avec les sessions, la superglobale `$_SESSION` nous permet d'intéroger le fichier de session sur le serveur (présent dans le dossier `/tmp/`).

L'avantage c'est que les informations contenues dans le fichier de session seront disponibles sur tout le site durant la navigation de l'internaute !

C'est la raison pour laquelle vous êtes parfois toujours connecté lorsque vous retournez sur un site web.

*Le fichier `bonjour.php` pourrait retourner une erreur si la session était détruite car en l'absence d'une condition IF, nous tenterons toujours d'afficher le pseudo contenu dans la session, il y a donc 1 risque d'erreur undefined.*

100

#### Vider une partie du fichier de session

Pour vider un fichier de session, nous utilisons la fonction prédéfinie `unset()` :

session1.php

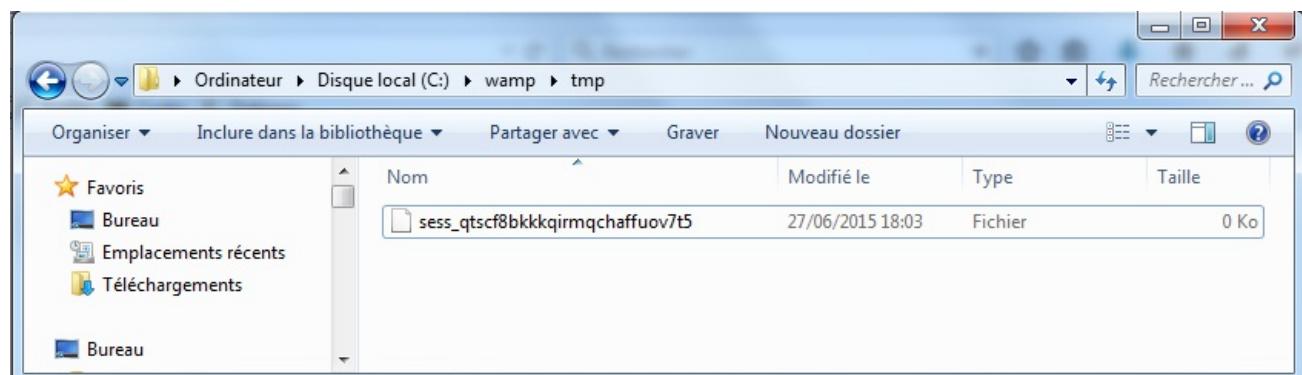
```
<?php  
session_start();  
$_SESSION['pseudo'] = "Mathieu";  
$_SESSION['mdp'] = "Balmont";  
print_r($_SESSION);  
unset($_SESSION['mdp']);  
print_r($_SESSION);
```

#### Quelques Explications :

`unset($_SESSION['mdp']);` permet de vider la partie "mdp" (mot de passe).

Rouvrez le contenu de votre fichier temporaire (dossier `/tmp/`, présent à l'intérieur de wamp).

#### Résultat



Ouvrons ce fichier (clic droit > éditer avec notepad++) :

```
pseudo|s:7:"Mathieu";
```

## Supprimer un fichier de session

Pour supprimer un fichier de session, nous utilisons la fonction prédefinie `session_destroy()` :

```
session1.php
```

```
<?php
session_start();
$_SESSION['pseudo'] = "Mathieu";
$_SESSION['mdp'] = "Balmon";
print_r($_SESSION);
unset($_SESSION['mdp']);
print_r($_SESSION);
session_destroy();
print_r($_SESSION);
```

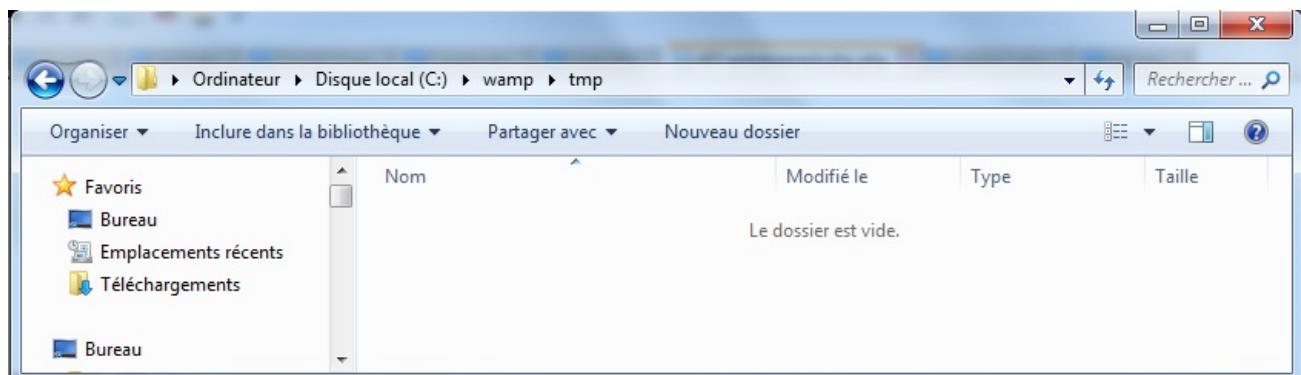
Quelques Explications :

`session_destroy();` permet de supprimer physiquement le fichier de session dans le dossier `/tmp/` et donc toutes les informations contenues dans le fichier.

Petite particularité : il faut savoir que le `session_destroy()` est vu par l'interpréteur, gardé, puis exécuté qu'à la fin du script.

C'est la raison pour laquelle le dernier `print_r` vous donnera toujours accès aux informations de la session car l'exécution du script n'est pas terminé.

Cependant, si vous retournez dans votre dossier `/tmp/`, présent à l'intérieur de wamp, vous verrez que le fichier n'existe plus !



## Explications supplémentaires

Comment se fait-il, techniquement, que quand vous rendez sur un site et que vous mettez le bon pseudo et le bon mdp, vous restez connecté pendant 1h si votre navigation dure 1h ? pourquoi on ne vous demande pas de vous reconnecter à chaque changement de page ? ...car il existe une session prouvant que vous êtes déjà passé par la porte d'entrée (vous êtes déjà identifié) ! Sans session, la connexion ne serait pas maintenue.

Les informations d'une session sont enregistrées dans la session côté serveur, cela crée (dans le même temps) un cookie précisément à l'identifiant de la session, sur le pc du client.

C'est la raison pour laquelle nous enregistrerons le prix d'un panier dans un fichier de session, de cette manière il ne pourra pas être modifié (par l'internaute) car il s'agit d'un fichier enregistré directement sur le serveur du site. Et, dans le cookie nous reliant au site, nous retrouverons uniquement l'identifiant qui permet de mener à la bonne session (la notre).

Si le prix d'un panier était enregistré dans le cookie, l'internaute pourrait le modifier car il s'agit d'un fichier sur son ordinateur.

Sur 1 site web il peut y avoir des dizaines, des centaines, des milliers ou des millions de sessions, il ne suffit pas d'en créer 1 à un internaute, il faut aussi que l'internaute soit relié à la sienne, c'est donc aussi grâce au cookie que le lien est établi. Si l'internaute supprime son cookie ou si le site décide de supprimer la session : le lien est cassé.

Quand vous effectuez un `session_start()`, un fichier de session est créé côté serveur (`/tmp/`) et un cookie de session est créé automatiquement côté client (pc) afin que le lien entre le site et le pc de l'internaute soit bien établi.

Quand vous cliquez sur "se déconnecter" :

- cela effectue un "`unset()`" pour vider votre partie de membre mais quand même conserver le reste des informations (produit dans le panier par exemple).
- cela peut aussi effectuer un "`session_destroy()`" pour supprimer le fichier de session complètement.

TOUS les sites qui vous proposent une connexion fonctionne avec les sessions. N'hésitez donc pas à relire ce chapitre s'il n'est pas parfaitement clair.

103

## Exemple d'un formulaire avec session

Pour mieux comprendre le système de session, munissons-nous d'un formulaire simple.

```
formulaire_session.php

<?php
session_start();
if($_POST)
{
    $_SESSION['pseudo'] = $_POST['pseudo'];
}
//-----
if(isset($_SESSION['pseudo']))
{
    echo "votre pseudo est: " . $_SESSION['pseudo'] . "<br>";
}
else
{
    echo ' <form method="post" action="">
        <label for="pseudo">Pseudo:</label><br>
        <input type="text" name="pseudo" value=""><br>
        <input type="submit" value="envoi">
    </form>';
}
```

Quelques Explications :

`session_start();` nous permet de créer 1 fichier de session la première fois (et de l'ouvrir les fois suivantes).

`if($_POST)` permet de déclarer une condition "si l'internaute a posté en cliquant sur le bouton submit", nous rentrons dans les accolades du IF et exécutons le code présent

- Enregistrement du pseudo posté par l'internaute via le formulaire, dans le fichier de session.

`if(isset($_SESSION['pseudo']))` permet de déclarer une condition "si un pseudo est défini dans le fichier de session de l'internaute", nous rentrons dans les accolades du IF et exécutons le code présent

- Affichage du pseudo de l'internaute conservé dans le fichier de session.

`else` (ce else se réfère au IF précédent) permet de déclarer un cas par défaut "si aucun pseudo n'est défini dans le fichier de session de l'internaute", nous rentrons dans les accolades du ELSE et exécutons le code présent

- Affichage d'un formulaire.

Résultat

Pseudo:

1 2 3 4 5

Le formulaire sera affiché tant qu'un pseudo n'est pas saisi, posté, et conservé dans le fichier de session.

Lorsqu'un pseudo est conservé dans le fichier de session, le formulaire n'apparaît plus.

Pour ré-initialiser la page et faire ré-apparaître le formulaire, vous pouvez vider vos cookies (**ctrl+shift+suppr**) cela cassera le lien avec le fichier de session.

104

## Exemple avec l'inactivité d'une session

Il arrive parfois que la durée de vie des sessions soit limitée sur certains sites (par exemple sur les sites des banques vous êtes déconnecté automatiquement au bout de 10 minutes d'inactivité).

Le réglage de la durée de vie d'une session peut se faire dans le paramétrage du serveur ou directement dans le script.

Voici un exemple :

```
inactivitee.php
```

```
<?php
session_start();
echo 'Temps Actuel : ' . time() . '<br>';
print "<pre>"; print_r($_SESSION); print "</pre>";

if(isset($_SESSION['temps']))
{
    if(time() > $_SESSION['limite'] + $_SESSION['temps'])
    {
        session_destroy();
        echo "déconnexion";
    }
    else
    {
        $_SESSION['temps'] = time();
        echo "connexion mise à jour";
    }
}
else
{
    echo "connexion";
    $_SESSION['limite'] = 20;
    $_SESSION['temps'] = time();
}
```

Quelques Explications et déroulement du script :

*Pour les tests il serait judicieux de spliter l'écran en deux avec d'un côté votre page web et de l'autre votre dossier /tmp/.*

`session_start()` nous permet de créer 1 fichier de session la première fois (et de l'ouvrir les fois suivantes).

`echo 'Temps Actuel : ' . time() . '`  
'; permet de voir le timestamp actuel

`if(isset($_SESSION['temps']))` si une session du nom de "temps" existe nous rentrons dans le IF (ce qui n'est pas le cas, la toute première fois), nous rentrerons donc dans le ELSE

`else` Nous rentrons dans le ELSE car aucune session du nom de "temps" existe.

`$_SESSION['limite'] = 20;` Nous rentrons le chiffre 20 dans une partie de la session que nous nommerons "limite". Cela nous permettra plus tard de limiter la session à 20 secondes.

`$_SESSION['temps'] = time();` Nous rentrons le timestamp actuel (nombre de secondes écoulées entre le 01/01/1970 et maintenant) dans une partie de la session que nous nommerons "temps". Cela nous permettra plus tard d'interroger la dernière trace d'activité sur le site web.

...Relancer l'exécution de la page avec l'aide de **F5** ou **Ctrl+R**.

...Le code se ré-exécute.

```
if(isset($_SESSION['temps'])) si une session du nom de "temps" existe nous rentrons dans le IF (ce qui est le cas, la seconde première fois, nous ne rentrerons donc pas dans le ELSE
```

```
if(time()>$_SESSION['limite'] + $_SESSION['temps'])) si le timestamp actuel est supérieur au dernier timestamp enregistré dans la session + la partie limite enregistrée dans la session (pour rappel, 20 secondes), c'est donc que nous avons dépassé notre temps de connexion inactive autorisée, nous rentrons dans les accolades du IF
```

```
session_destroy(); permet de supprimer le fichier de session.
```

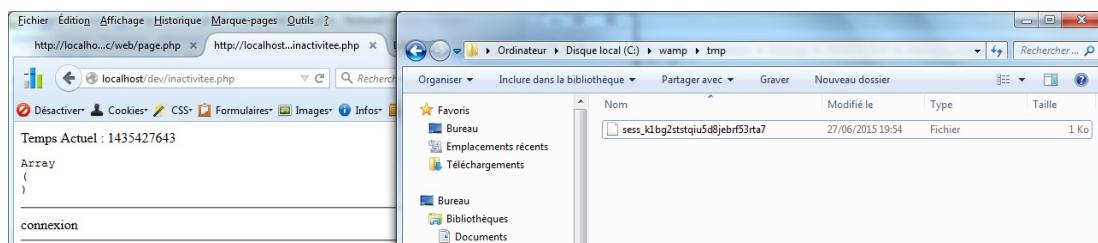
... Dans l'autre cas, si vous affichez la page web une première fois et que vous la rafraîchissez rapidement (moins de 20 secondes après), nous rentrons dans le 1er IF puis dans le ELSE (présent à l'intérieur du premier IF)

```
$_SESSION['temps'] = time(); cela nous permet de mettre la connexion à jour en affectant le timestamp actuel dans la session (cela le modifiera pour l'actualiser).
```

Pour conclure :

Nous n'avons pas autorisé l'internaute à garder une connexion de 20 secondes au total mais nous l'avons autorisé à conserver une connexion de 20 secondes après son dernier mouvement (clic, rafraîchissement de page, etc.).

## Résultat



Les résultats diffèrent selon l'état de la page web.

Démo conseillée : Lancer l'exécution de la page...attendre 2 sec, effectuer un **ctrl+r**, attendre 2 sec, effectuer un **ctrl+r**, attendre 30 secondes, effectuer un **ctrl+r**. Et surtout (au même moment) observer le dossier /tmp/ avec un écran splité en deux.

105

## Gérer la durée de vie des sessions

### Gérer la durée de vie des sessions et/ou supprimer les fichiers de session du serveur

Dans le fichier de configuration php.ini (pas toujours disponible, selon l'hébergement souscrit, mutualisé, dédié, vps, etc.)

`session.gc_maxlifetime` : permet de déterminer la durée de vie des fichiers de sessions du dossier /tmp/ avant qu'ils ne soient supprimés. Cela commence par "gc" pour Garbage Collector.

106

## Sécurité des fichiers de sessions

Dans le fichier de configuration php.ini, nous pouvons modifier 2 ou 3 paramètres liés à l'utilisation des sessions :

```
session.use_cookies 1 permet de préciser que l'identifiant de session sera transmis par un cookie
```

```
session.use_only_cookies 1 permet de préciser que seulement le cookie sera en mesure de transmettre l'identifiant de session (les autres moyens seront interdits et refusés)
```

```
session.use_trans_sid 0 L'argument/paramètre PHPSESSID transmis parfois dans les adresses url des sites web sera strictement refusé dans notre
```

## Requête SQL avec PDO et MYQLI

### Objectifs

- ✓ Sauvegarder, utiliser et exploiter pleinement une base de données en faisant communiquer les technologies PHP/SQL.

107

### Pourquoi mélanger le PHP et le SQL ?

Vous avez appris SQL ? Vous avez appris PHP ?

Il est maintenant temps d'apprendre le mélange de PHP + SQL pour insérer, modifier, supprimer et consulter des données existantes.

Bref, en 1 mot, ce que tous les sites du monde (ou presque) font à longueur de journée !

Dans un site web, il est utile de pouvoir manipuler des données.

Quelques exemples rapides :

➤ **Insérer des données :**

côté FRONT : Lorsqu'un internaute s'inscrit sur votre site web,

côté BACK : Lorsque vous ajoutez un nouveau produit dans votre boutique.

➤ **Modifier des données :**

côté FRONT : Lorsqu'un internaute modifie son profil sur votre site web,

côté BACK : Lorsque vous modifiez le prix d'un produit de votre boutique.

➤ **Supprimer des données :**

côté FRONT : Lorsqu'un internaute supprime son compte de votre site web,

côté BACK : Lorsque vous supprimez un produit de votre boutique.

➤ **Consulter des données :**

côté FRONT : Lorsqu'un internaute souhaite se connecter à votre site web,

côté BACK : Lorsque vous affichez les produits de votre boutique.

108

### Comment faire pour échanger des informations entre la base de données et la page web ?

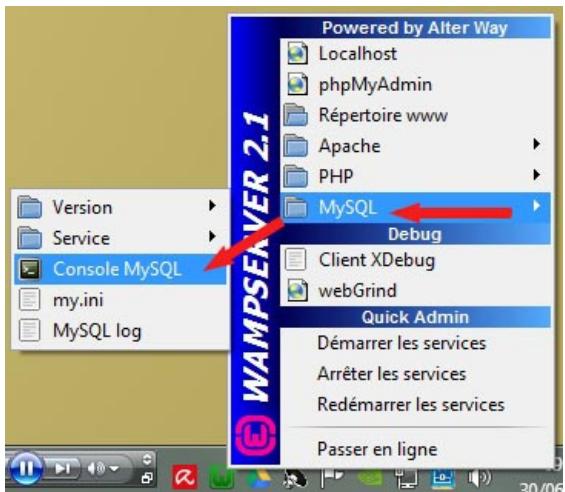
Pour commencer, il vous faut une base de données.

Nous pouvons créer une base de données à l'aide de la console Mysql ou du gestionnaire PhpMyAdmin.



Création de la Base de données : entreprise

Avec la console Mysql :



```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

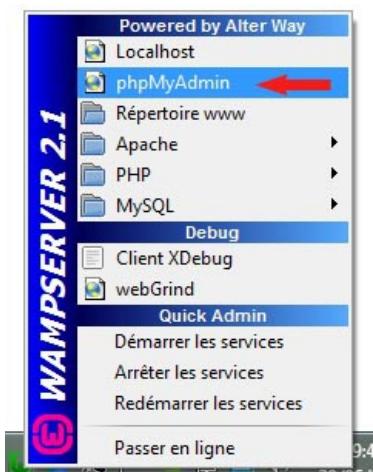
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> create database entreprise ;
```

CREATE DATABASE entreprise;

Si vous êtes en mode console, nous rajouterons une ligne de code pour dire au SGBD que nous souhaitons travailler sur notre base de données entreprise :

```
USE entreprise;
```

Avec le gestionnaire PhpMyAdmin :



The screenshot shows the phpMyAdmin dashboard for 'localhost'. The 'Bases de données' tab is selected. A red arrow points to the 'Créer une base de données' button. The input field contains 'entreprise'. Below it, the 'Interclassement pour la connexion MySQL:' dropdown is set to 'utf8\_general\_ci'. The 'Actions' and 'Interface' sections are also visible.

Une fois la base de données "entreprise" créée et utilisée, nous aurons besoin d'une table pour contenir des enregistrements :

**Création de la table : employes**

**Base de données entreprise - Table employes**

```
CREATE TABLE IF NOT EXISTS employes (
    id_employes int(3) NOT NULL AUTO_INCREMENT,
    prenom varchar(20) DEFAULT NULL,
    nom varchar(20) DEFAULT NULL,
    sexe enum('m','f') NOT NULL,
    service varchar(30) DEFAULT NULL,
    date_embauche date DEFAULT NULL,
    salaire float DEFAULT NULL,
    PRIMARY KEY (id_employes)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ;
```

#### Insertion et enregistrement des informations :

Dans tous les cas, voici les enregistrements à insérer dans la base de données entreprise afin d'avoir du contenu sur lequel travailler :

```
INSERT INTO employes (id_employes, prenom, nom, sexe, service, date_embauche, salaire) VALUES
(350, 'Jean-pierre', 'Laborde', 'm', 'direction', '1999-12-09', 5000),
(388, 'Clement', 'Gallet', 'm', 'commercial', '2000-01-15', 2300),
(415, 'Thomas', 'Winter', 'm', 'commercial', '2000-05-03', 3550),
(417, 'Chloe', 'Dubar', 'f', 'production', '2001-09-05', 1900),
(491, 'Elodie', 'Fellier', 'f', 'secretariat', '2002-02-22', 1600),
(509, 'Fabrice', 'Grand', 'm', 'comptabilite', '2003-02-20', 1900),
(547, 'Melanie', 'Collier', 'f', 'commercial', '2004-09-08', 3100),
(592, 'Laura', 'Blanchet', 'f', 'direction', '2005-06-09', 4500),
(627, 'Guillaume', 'Miller', 'm', 'commercial', '2006-07-02', 1900),
(655, 'Celine', 'Perrin', 'f', 'commercial', '2006-09-10', 2700),
(699, 'Julien', 'Cottet', 'm', 'secretariat', '2007-01-18', 1390),
(701, 'Mathieu', 'Vignal', 'm', 'informatique', '2008-12-03', 2000),
(739, 'Thierry', 'Desprez', 'm', 'secretariat', '2009-11-17', 1500),
(780, 'Amandine', 'Thoyer', 'f', 'communication', '2010-01-23', 1500),
(802, 'Damien', 'Durand', 'm', 'informatique', '2010-07-05', 2250),
(854, 'Daniel', 'Cheval', 'm', 'informatique', '2011-09-28', 1700),
(876, 'Nathalie', 'Martin', 'f', 'juridique', '2012-01-12', 3200),
(900, 'Benoit', 'Lagarde', 'm', 'production', '2013-01-03', 2550),
(933, 'Emilie', 'Sennard', 'f', 'commercial', '2014-09-11', 1800),
(990, 'Stephanie', 'Lafaye', 'f', 'assistant', '2015-06-02', 1775);
```

**Création du fichier :**  `mysqli.php`

Ensuite, il vous faut aussi un fichier où vous pourrez écrire le code (script).

Pour cela, rien de plus facile puisque'il s'agit de créer un simple fichier nommé "mysqli.php" sur le serveur local (repertoire /www/).

Avant de pouvoir formuler des requêtes, il faut absolument relier la base de données à votre fichier (script).

Pour cela, nous devons le préciser en haut de fichier :

```
mysqli.php
```

```
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
```

Quelques Explications :

`Mysqli` est une classe prédefinie de PHP permettant de se connecter à une base de données et de formuler des requêtes SQL.

Les arguments/paramètres entrants vont directement dans le constructeur de la classe (méthode exécutée par défaut).

Les arguments/paramètres correspondent à :

- nom du serveur = localhost
- pseudo = root
- mot de passe = (*sous wamp, le mot de passe est vide - sous mamp, il sera nécessaire d'écrire "root"*)
- nom de la base de données = entreprise

Techniquement, nous pouvons dire que `$mysqli` est une variable représentant 1 objet issu de la classe `Mysqli`.

Humainement, nous pouvons dire que `$mysqli` est une variable permettant d'être connecté à la base et de formuler des requêtes.

Pour mieux connaitre la classe `Mysqli`, vous pouvez consulter la documentation officielle : [Classe Mysqli](#)

Vous pouvez aussi demander l'affichage des propriétés avec l'instruction `print_r` et aussi la fonction prédefinie `get_class_methods()` :

```
mysqli.php
```

```
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
echo '<pre>'; print_r($mysqli); echo '</pre>';
echo '<pre>'; print_r(get_class_methods($mysqli)); echo '</pre>';
```

*Nous aurons l'occasion d'utiliser plusieurs des propriétés et méthodes présentes dans cette liste.*

## MYSQLI : Ecrire une première requête SQL

Pour formuler une première requête SQL dans une page web, nous aurons besoin de passer par l'objet `$mysql` et d'utiliser la méthode `query()` :

```
mysqli.php
```

```
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
echo '<pre>'; print_r($mysqli); echo '</pre>';
echo '<pre>'; print_r(get_class_methods($mysqli)); echo '</pre>';
//-----
$mysqli->query("mauvaise requete volontaire ....");
echo $mysqli->error . '<br>';
```

### Debug

Comme vous pouvez le voir, la propriété nommée `error` de l'objet `Mysqli` nous permet d'avoir accès aux éventuelles erreurs SQL.

Pour cet exemple, vous aurez donc une erreur puisque c'était volontaire de montrer comment se débugger dans le cas d'une mauvaise requête SQL.

## MYSQLI : Insérer des données

Voici le code permettant d'insérer des données avec la classe Mysqli :

```
mysql.php
```

```
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
// Insertion :
$result = $mysqli->query("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000);");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
```

Profitez en pour vous attribuer un bon salaire, c'est vous qui décidez :p.

**Query** permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable **\$result** juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, **\$result** contiendra : (boolean) FALSE
- Dans le cas d'une bonne requête SQL, **\$result** contiendra : (boolean) TRUE

**affected\_rows** permet d'observer le nombre d'enregistrements affectés par une requête.

Résultat 1 enregistrement(s) affecté(s) par la requête INSERT.

Après l'INSERT, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé à bien été ajouté !

## MYSQLI : Modifier des données

Voici le code permettant de modifier des données avec la classe Mysqli :

```
mysql.php
```

```
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
// Insertion (INSERT) :
$result = $mysqli->query("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000);");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $mysqli->query("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----
```

Nous modifions le salaire de l'employé 802 (Damien Durand), initialement à 2250, il passera à 2500.

**Query** permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable **\$result** juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, **\$result** contiendra : (boolean) FALSE
- Dans le cas d'une bonne requête SQL, **\$result** contiendra : (boolean) TRUE

**affected\_rows** permet d'observer le nombre d'enregistrements affectés par une requête.

Résultat 1 enregistrement(s) affecté(s) par la requête UPDATE.

Avant et après l'UPDATE, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé à bien été modifié !

## MYSQLI : Supprimer des données

Voici le code permettant de supprimer des données avec la classe Mysqli :

```
mysql.php
```

```

<?php
$mysqli = new mysqli("localhost", "root", "", "entreprise");
//-----
// Insertion (INSERT) :
$result = $mysqli->query("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000)");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $mysqli->query("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----
// Suppression (DELETE) :
$result = $mysqli->query("DELETE FROM employes WHERE id_employes = 388");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
//-----

```

Nous supprimons l'employé 388 (Clement Gallet).

Query permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable \$result juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) false
- Dans le cas d'une bonne requête SQL, \$result contiendra : (boolean) TRUE

affected\_rows permet d'observer le nombre d'enregistrements affectés par une requête.

Résultat 1 enregistrement(s) affecté(s) par la requête DELETE.

Avant et après la requête DELETE, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé à bien été supprimé !

112

## MYSQLI : Sauvegarder des informations

### MYSQLI : Sauvegarder les informations d'un formulaire dans une base de données (SQL/PHP)

Maintenant que l'on sait échanger des informations entre la base de données et la page web, nous pourrions travailler avec des formulaires !

La création d'un formulaire permettant d'ajouter des employés serait judicieux afin qu'une personne non familière des langages informatiques puisse mettre à jour la base de données facilement.

Création d'un nouveau fichier :  formulaire-mysqli.php

#### formulaire-mysqli.php

```

<?php
$mysqli = new mysqli("localhost", "root", "", "entreprise");
if($_POST)
{
    $result = $mysqli->query("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('$_POST[prenom]', '$_POST[nom]', '$_POST[sexe]', '$_POST[service]', '$_POST[date_embauche]', '$_POST[salaire]')");
    echo '<div style="background: green; padding: 5px;">l'employé a bien été ajouté</div>';
}
?>

<form method="post">
    <label for="prenom">Prenom</label><br>
    <input type="text" name="prenom" placeholder="prenom" id="prenom" required=""><br><br>
    <label for="prenom">Nom</label><br>
    <input type="text" name="nom" placeholder="nom" id="nom" required=""><br><br>
    <label for="sexe">Sexe</label><br>
    Homme <input type="radio" name="sexe" placeholder="sexe" id="sexe" value="m" checked=""> - 
    Femme <input type="radio" name="sexe" placeholder="sexe" id="sexe" value="f"> <br><br>
    <label for="service">Service</label><br>
    <input type="text" name="service" placeholder="service" id="service"><br><br>
    <label for="date_embauche">Date d'embauche <i>(Format: AAAA-MM-JJ)</i></label><br>
    <input type="text" name="date_embauche" placeholder="ex: 2015-07-30" id="prenom"><br><br>
    <label for="salaire">Salaire</label><br>
    <input type="text" name="salaire" placeholder="salaire" id="salaire"><br><br>
    <input type="submit"><br><br>
</form>

```

Lorsque nous arrivons la première fois sur la page, le code PHP se trouvant dans le IF est ignoré puisque personne n'a validé le formulaire.  
Lorsque l'internaute clique sur le bouton submit, cela recharge la page (le script), et la partie de code PHP dans le IF est exécutée pour enregistrer l'employé qu'on nous aura posté.

Bien entendu, ce formulaire peut être amélioré.

Dans l'idéal, il faudrait réaliser des contrôles (est-ce que tous les champs ont été correctement remplis, etc.), le sécuriser (se protéger d'éventuelles attaques), et augmenter l'expérience utilisateur (en proposant par exemple 1 calendrier pour sélectionner une date d'embauche).

En l'état, ce formulaire est fonctionnel et c'est bien là l'essentiel pour l'apprentissage de notre cours.

#### Résultat

l'employé a bien été ajouté

Prenom

Nom

Sexe  
Homme  - Femme

Service

Date d'embauche (Format: AAAA-MM-JJ)

Salaire

113

## MYSQLI : Selectionner et afficher des données

### MYSQLI : Selectionner et afficher 1 enregistrement

Pour afficher dans la page web les données d'un enregistrement à l'intérieur de la base, le code sera légèrement différent.

Voici le code permettant d'afficher un enregistrement avec la classe Mysqli :

```
mysql.php

<?php
$mysqli = new mysqli("localhost", "root", "", "entreprise");
// Insertion (INSERT) :
$result = $mysqli->query("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000)");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
// Modification (UPDATE) :
$result = $mysqli->query("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
// Suppression (DELETE) :
$result = $mysqli->query("DELETE FROM employes WHERE id_employes = 388");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
// Affichage (SELECT) :
$result = $mysqli->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch_assoc();
echo "<pre>"; print_r($employe); echo "</pre>";
echo "Bonjour je suis $employe[prenom] $employe[nom] du service $employe[service]<br>";
```

```
//-----
```

## Query

Nous selectionnons un employé en particulier (Julien, à vous de vérifier s'il est bien présent dans votre table, sinon il faudra changer le prénom ou selectionner directement par un id existant).

Nous obtenons l'enregistrement demandé dans la variable `$result` prévue pour récupérer le retour de la requête SQL.

## `$result`

Nous avons prévu une variable `$result` juste avant la requête pour obtenir un retour.

Par défaut, le retour de la requête SQL crée un nouvel objet, issu de la classe `MYSQLI_RESULT`

- Dans le cas d'une erreur de requête SQL, `$result` contiendra : (boolean) `false`
- Dans le cas d'une bonne requête SQL, `$result` contiendra : (object) `MYSQLI_RESULT` Si la requête est bonne, vous obtiendrez un autre objet issu d'une autre classe (`MYSQLI_RESULT`) !

## `fetch_assoc`

Le résultat (ressource) sous forme d'objet `MYSQLI_RESULT` n'est pas exploitable en l'état.

Pour accéder aux données que nous avons sélectionnées précédemment, nous avons besoin d'utiliser la méthode (fonction) `fetch_assoc()` de la classe `Mysqli_Result` (`$result->fetch_assoc()`). La méthode `fetch_assoc()` permet de rendre un résultat exploitable sous forme de tableau ARRAY associatif.

## `$employe`

`$employe` est donc un tableau ARRAY associatif (associatif = avec des clés nominatives).

Nous pouvons donc afficher notre tableau ARRAY par l'intermédiaire d'un `print_r` ou d'un `echo`

`affected_rows` permet toujours d'observer le nombre d'enregistrements affectés par une requête.

## résultat

```
Array
(
    [id_employees] => 7699
    [prenom] => julien
    [nom] => cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 1390
)
```

Bonjour je suis Julien Cottet du service secretariat

Avec cette requête de `SELECT`, nous avons pu afficher 1 enregistrement de la table `employees`.

## MYSQLI : Selectionner et afficher plusieurs enregistrements

Pour afficher dans la page web les données de PLUSIEURS enregistrements à l'intérieur de la base, le code sera encore différent.

Voici le code permettant d'effectuer un affichage de données avec la classe `Mysqli` :

```
mysqli.php
??
<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
// Insertion (INSERT) :
$result = $mysqli->query("INSERT INTO employees (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000)");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
// Modification (UPDATE) :
$result = $mysqli->query("UPDATE employees SET salaire = 2500 WHERE id_employees = 802");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
// Suppression (DELETE) :
$result = $mysqli->query("DELETE FROM employees WHERE id_employees = 388");
echo $mysqli->affected_rows . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
// Affichage (SELECT) :
$result = $mysqli->query("SELECT * FROM employees");
while($employe = $result->fetch_assoc())
```

```

{
    echo "<pre>"; print_r($employe); echo "</pre>";
    echo "Bonjour je suis $employe[prenom] $employe[nom] du service $employe[service]";
}
echo $result->num_rows . ' enregistrement(s) récupéré(s) par la requête SELECT<br>';
//-----

```

## Query

Nous sélectionnons tous les employés.

Nous obtenons les enregistrements demandés dans la variable \$result prévue pour récupérer le retour de la requête SQL.

## \$result

Nous avons prévu une variable \$result juste avant la requête pour obtenir un retour.

Par défaut, le retour de la requête SQL crée un nouvel objet, issu de la classe MYSQLI\_RESULT

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) false
- Dans le cas d'une bonne requête SQL, \$result contiendra : (object) MYSQLI\_RESULT Si la requête est bonne, vous obtiendrez un autre objet issu d'une autre classe (MYSQLI\_RESULT) !

## While

while est une boucle permettant d'effectuer une répétition.

Pourquoi effectuer une répétition ?

Parce que nous avons plusieurs lignes d'enregistrements à récupérer. Il est donc nécessaire de répéter le traitement fetch\_assoc() afin de rendre le résultat exploitable sous forme d'array.

La boucle While permet d'afficher chaque ligne de la table (tant que l'on possède des enregistrements, on les affiche).



### ➤ Informations

votre requête sort plusieurs résultats ? : une boucle !

votre requête ne doit sortir qu'un seul et unique résultat ? : Pas de boucle !

votre requête ne sort qu'un seul résultat à l'instant T mais peut potentiellement en sortir plusieurs plus tard ? : une boucle !

## fetch\_assoc

Le résultat (ressource) sous forme d'objet MYSQLI\_RESULT n'est pas exploitable en l'état.

Pour accéder aux données que nous avons sélectionnées précédemment, nous avons besoin d'utiliser la méthode (fonction) fetch\_assoc() de la classe MYSQLI\_RESULT (\$result->fetch\_assoc()). La méthode fetch\_assoc() permet de rendre un résultat exploitable sous forme de tableau ARRAY associatif.

Avec la présence de la boucle while, nous obtiendrons un nouveau tableau ARRAY dans la variable \$employe, à chaque tour de boucle.

! Il y aura pas un tableau ARRAY avec tous les enregistrements à l'intérieur mais bien un nouveau tableau ARRAY par enregistrement, un array par employé !

## \$employe

\$employe est donc un tableau ARRAY associatif (associatif = avec des clés nominatives).

Nous pouvons donc afficher notre tableau ARRAY par l'intermédiaire d'un print\_r ou d'un echo

num\_rows permet d'observer le nombre d'enregistrements récupérés et affichés par une requête.

## résultat

```

Array
(
    [id_employees] => 7350
    [prenom] => jean-pierre
    [nom] => laborde
    [sexe] => m
    [service] => direction
    [date_embauche] => 1999-12-09
    [salaire] => 5000
)

```

Bonjour je suis jean-pierre laborde du service direction

```

Array
(
    [id_employes] => 7388
    [prenom] => clement
    [nom] => gallet
    [sexe] => m
    [service] => commercial
    [date_embauche] => 2015-05-15
    [salaire] => 2300
)

```

Bonjour je suis clement gallet du service commercial

...etc...

Avec cette requête de SELECTION, nous avons pu afficher tous les enregistrements de la table employes.

### MYSQLI : Une table SQL sous forme de table HTML

Voici le code qui permettrait de reproduire la table sql (de la base de données) dans une table html (dans la page web).

Table SQL

```

<?php
$mysqli = new mysqli("localhost", "root", "", "entreprise");
$résultat = $mysqli->query("SELECT * FROM employes");

echo '<table border="5"> <tr>';
while($colonne = $résultat->fetch_field())
{
    echo '<th>' . $colonne->name . '</th>';
}
echo "</tr>";

while ($ligne = $résultat->fetch_assoc())
{
    echo '<tr>';
    foreach ($ligne as $indice => $information)
    {
        echo '<td>' . $information . '</td>';
    }
    echo '</tr>';
}
echo '</table>';
?>

```

### Résultat

id_employes	prenom	nom	sexe	service	date_embauche	salaire
350	Jean-pierre	Laborde	m	direction	1999-12-09	5000
388	Clement	Gallet	m	commercial	2000-01-15	2300
415	Thomas	Winter	m	commercial	2000-05-03	3550
417	Chloe	Dubar	f	production	2001-09-05	1900
491	Elodie	Fellier	f	secretariat	2002-02-22	1600
509	Fabrice	Grand	m	comptabilite	2003-02-20	1900
547	Melanie	Collier	f	commercial	2004-09-08	3100
592	Laura	Blanchet	f	direction	2005-06-09	4500
627	Guillaume	Miller	m	commercial	2006-07-02	1900
655	Celine	Perrin	f	commercial	2006-09-10	2700
699	Julien	Cottet	m	secretariat	2007-01-18	1390
701	Mathieu	Vignal	m	informatique	2008-12-03	2000
739	Thierry	Desprez	m	secretariat	2009-11-17	1500
780	Amandine	Thoyer	f	communication	2010-01-23	1500

802	Damien	Durand	m	informatique	2010-07-05	2250
854	Daniel	Chevel	m	informatique	2011-09-28	1700
876	Nathalie	Martin	f	juridique	2012-01-12	3200
900	Benoit	Lagarde	m	production	2013-01-03	2550
933	Emilie	Sennard	f	commercial	2014-09-11	1800
990	Stephanie	Lafaye	f	assistant	2015-06-02	1775

### Toutes les explications

Nous allons tâcher d'expliquer chaque ligne pour une compréhension optimale.

```
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
```

Cette ligne représente la connexion à notre base de données avec les informations suivantes : nom du serveur, pseudo, mot de passe, nom de la base de données.

Humainement, \$mysqli représente la variable permettant de communiquer et de gérer des échanges avec la base de données.

Techniquement, \$mysqli est un objet issu de la classe Mysqli.

```
$resultat = $mysqli->query("SELECT * FROM employes");
```

Cette ligne permet de formuler une requête SQL afin de sélectionner tous les employés de la table via la méthode query() de l'objet \$mysqli.

Nous récupérons le résultat de la requête dans la variable \$resultat.

```
echo '<table border="5"> <tr>';
```

Nous créons une table html avec une première ligne (<tr>)

```
while($colonne = $resultat->fetch_field())
```

Pour consulter le nom des champs/colonnes de la table SQL, nous aurons besoin d'utiliser la méthode fetch\_field() qui permet de récolter des informations sur les champs/colonnes.

La boucle While est présente pour répéter cette action puisqu'il y a plusieurs champs/colonnes à afficher

\$colonne est 1 objet (de la StdClass) car la méthode fetch\_field() que nous utilisons est prévue de récolter des informations sur les champs/colonnes d'une table SQL et de faire un retour sous forme d'objet.

En 1 mot : fetch\_field() est une méthode de l'objet Mysqli\_Result qui nous permet de récupérer des informations sur les champs sous forme d'objet.

Nous faisons tourner une première boucle while pour parcourir tous les champs et obtenir des informations via la méthode fetch\_field() afin d'écrire les en-têtes de notre tableau HTML sur la première ligne (balise tr).

```
echo '<th>' . $colonne->name . '</th>';
```

Nous inscrivons des <th> (en html) pour commencer l'affichage des entêtes de la table.

A l'intérieur des balises html <th>, nous avons besoin d'inscrire les noms des champs.

Nous passons donc par la variable \$colonne (remplie précédemment avec fetch\_field()) et nous demandons le name du champ.

Nous aurions pu demander d'autres informations tel que le type \$colonne->type.

Pour voir les informations auxquelles nous avons accès, il est possible d'effectuer un `print_r($colonne);` ou de [consulter la documentation officielle](#) à ce sujet !

A chaque tour de boucle, le nom du champ sera affiché.

```
echo "</tr>";
```

Nous terminons l'affichage de la 1ère ligne en fermant la balise `</tr>`.

```
while ($ligne = $resultat->fetch_assoc())
```

`fetch_assoc` permet de traiter le résultat et de le rendre exploitable sous forme de tableau ARRAY (nous utilisons la méthode `fetch_assoc` de l'objet `mysqli_result`).

`$ligne` représente un tableau array associatif contenant le résultat.

`while` permet de répéter cette action tant qu'il y a des résultats et de passer à la ligne d'enregistrement suivante pour faire avancer les traitements.

Nous mettons en place une boucle `while` pour parcourir chaque enregistrement (soit 1 ligne par enregistrement) de la table SQL.

```
echo "<tr>";
```

Nous préparons l'affichage d'une prochaine ligne en ouvrant la balise `<tr>`.

```
foreach($ligne as $indice => $information)
```

La boucle `foreach` va nous permettre de parcourir notre tableau array représenté par la variable `$ligne` et d'afficher chacune des informations contenues à l'intérieur.

Chaque employé possède 7 informations : `id_employes`, `prenom`, `nom`, `sexe`, `service`, `date_embauche`, `salaire`.

La boucle `while` est présente pour traiter chaque employé (et avancer sur l'employé suivant), tandis que, la boucle `foreach` est présente pour traiter et afficher chaque information pour chaque employé.

Lorsque nous faisons 1 tour dans la boucle `while`, cela entraîne 7 tours dans la boucle `foreach`.

S'il y a 20 employés, la boucle `while` fera 20 tours et la boucle `foreach` 140 tours (20 employés x 7 informations).

Voici ce que ça donne :

o While : Employé n°1

- Foreach : information n°1
- Foreach : information n°2
- Foreach : information n°3
- Foreach : information n°4
- Foreach : information n°5
- Foreach : information n°6
- Foreach : information n°7

o While : Employé n°2

- Foreach : information n°1
- Foreach : information n°2
- Foreach : information n°3
- Foreach : information n°4
- Foreach : information n°5
- Foreach : information n°6
- Foreach : information n°7

o While : Employé n°3

- Foreach : information n°1

- Foreach : information n°2
- Foreach : information n°3
- Foreach : information n°4
- Foreach : information n°5
- Foreach : information n°6
- Foreach : information n°7

etc.

```
echo '<td>' . $information . '</td>';
```

Cette ligne (présente à l'intérieur de la boucle foreach) permet d'afficher chaque information d'un employé via `$information` dans une nouvelle balise `<td>`.

```
echo "</tr>";
```

Nous terminons l'affichage de la ligne en fermant la balise `</tr>`.

```
echo "</table>";
```

Nous terminons l'affichage de notre table en fermant la balise `</table>`.

Dans les grandes lignes :

- Nous accédons aux entêtes des colonnes/champs avec la méthode `fetch_field()` (après une requête SELECT) - cela retourne un objet (de type OBJECT).
- Nous accédons aux informations sur les enregistrements avec `fetch_assoc` (ou `fetch_row`, `fetch_array`, `fetch_object`, `fetch_all`, après une requête SELECT) - cela retourne un tableau ou un objet (de type array ou object)
- Nous accédons au nombre d'enregistrements avec `num_rows` (après une requête SELECT) et `affected_rows` (après une requête INSERT, UPDATE, DELETE) - cela retourne le nombre (de type INT).

Vous pouvez garder ce code en référence pour vos besoins. Il suffira juste d'adapter le nom de la table sur laquelle vous effectuez une requête SQL.  
C'est une première approche, ne prenez pas peur, nous reviendrons dessus jusqu'à ce que chaque mot, chaque détail, soit compris !

Pour résumé:

#### Mysqli et Mysqli\_Result

`$mysqli` représente un objet [1] issu de la classe pré définie `Mysqli`

Lorsqu'on exécute une requête de SELECTION via la méthode `QUERY()` sur l'objet `Mysqli` :

- En cas de succès : On obtient un autre objet[2] issue de la classe `Mysqli_Result`. Cet objet a donc des méthodes et des propriétés différentes !
- En cas d'échec : On obtient un boolean `False`

Lorsqu'on exécute une requête INSERT/UPDATE/DELETE via la méthode `QUERY()` sur l'objet `Mysqli`, c'est plus simple :

- En cas de succès : On obtient un boolean `TRUE`.
- En cas d'échec : On obtient un boolean `FALSE`

Pour comprendre techniquement dans la précision, n'hésitez pas à lancer `var_dump($mysqli)` ou `var_dump($result)`;

#### While : boucle ou pas boucle ?

- S'il n'y a qu'une seule ligne d'enregistrement à récupérer dans le résultat, nous ne ferons pas de boucle while.
- S'il y a plusieurs lignes d'enregistrements à récupérer dans le résultat, il faut prévoir une boucle while.

Techniquement, la boucle while permet de faire avancer le curseur sur la ligne suivante dans la table.

#### affected\_rows et num\_rows

- Dans le cas d'une action (INSERT, UPDATE, DELETE), nous utiliserons `affected_rows` pour voir le nombre d'enregistrements affectés (touchés) par une requête.
- Dans le cas d'une question/réponse (SELECT), nous utiliserons `num_rows` pour voir le nombre d'enregistrements récupérés (et affichés) par une requête.

#### Pas d'affichage ou une erreur SQL ?

Dans le cas d'une erreur ou d'un affichage incohérent, n'hésitez pas à vérifier votre code PHP mais aussi votre code SQL (puisque maintenant nous mélangeons les 2 langages !).

Pour cela, vous pouvez utiliser (après une requête), le code suivant : `$mysqli->error`.

#### Fetch

La méthode `fetch_assoc` permet de traiter les résultats afin d'obtenir un tableau ARRAY.

Cette étape est indispensable si l'on souhaite produire un affichage par la suite.

D'autres méthodes de traitement existent.

#### fetch\_assoc

```
fetch_assoc

<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
$result = $mysqli->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch_assoc();
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe[prenom] . '<br>';
?>
```

#### Résultat

Avec `print_r` :

Array

```
{
    [id_employes] => 699
    [prenom] => Julien
    [nom] => Cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 1390
}
```

Avec une représentation plus compréhensible :

clé	valeur
id_employes	699
prenom	Julien
nom	Cottet
sexe	m
service	secretariat

date_embauche	2007-01-18
salaire	1390

Nous pouvons dire que `fetch_assoc` a l'avantage de donner des clés associatives/nominatives (correspondant au nom des champs/colonnes dans la table SQL) au tableau ARRAY.

#### fetch\_row

```
fetch_row

<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
$result = $mysqli->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch_row();
echo $employe[0] . "<br>";
echo "<pre>"; print_r($employe); echo "</pre>";
?>
```

#### Résultat

Avec `print_r` :

```
Array
(
    [0] => 699
    [1] => Julien
    [2] => Cottet
    [3] => m
    [4] => secretariat
    [5] => 2007-01-18
    [6] => 1390
)
```

Avec une représentation plus compréhensible :

clé	valeur
0	699
1	Julien
2	Cottet
3	m
4	secretariat
5	2007-01-18
6	1390

Nous pouvons dire que `fetch_row` donne des clés numériques (correspondant à l'ordre des champs/colonnes dans la table SQL) au tableau ARRAY.

#### fetch\_array

```
fetch_array

<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
$result = $mysqli->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch_array();
echo "<pre>"; print_r($employe); echo "</pre>";
?>
```

```
echo $employe['prenom'] . $employe[0] . '<br>';  
?>
```

## Résultat

Avec `print_r` :

```
Array  
(  
    [0] => 699  
    [id_employes] => 699  
    [1] => Julien  
    [prenom] => Julien  
    [2] => Cottet  
    [nom] => Cottet  
    [3] => m  
    [sexe] => m  
    [4] => secretariat  
    [service] => secretariat  
    [5] => 2007-01-18  
    [date_embauche] => 2007-01-18  
    [6] => 1390  
    [salaire] => 1390  
)
```

Avec une représentation plus compréhensible :

clé	valeur
0	699
id_employes	699
1	Julien
prenom	Julien
2	Cottet
nom	Cottet
3	m
sexe	m
4	secretariat
service	secretariat
5	2007-01-18
date_embauche	2007-01-18
6	1390
salaire	1390

Nous pouvons dire que `fetch_array` est une combinaison du `fetch_assoc` et du `fetch_row`, cela donne des clés numériques (correspondant à l'ordre des champs/colonnes dans la table SQL) et des clés associatives/nominatives (correspondant au nom des champs/colonnes dans la table SQL) au tableau ARRAY.

## fetch\_object

### fetch\_object

```
<?php  
$mysqli = new mysqli("localhost", "root", "", "entreprise");
```

```

//-----
$result = $mysqli->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch_object();
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe->prenom . '<br>';
?>

```

## Résultat

```

stdClass Object
(
    [id_employes] => 7699
    [prenom] => julien
    [nom] => cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 2500
)

```

`fetch_object` ne produit pas un tableau mais un objet (de la `stdClass`, class standard en PHP). Les champs/colonnes de la tables SQL deviennent le nom des propriétés de l'objet. Pour accéder à l'une d'entre-elles, il sera nécessaire d'utiliser la syntaxe suivante : `$employe->prenom`.

## fetch\_all

### fetch\_all

```

<?php
$mysqli = new Mysqli("localhost", "root", "", "entreprise");
//-----
$result = $mysqli->query("SELECT * FROM employes");
$employe = $result->fetch_all();
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe->prenom . '<br>';
?>

```

## Résultat

```

Array
(
    [0] => Array
        (
            [0] => 7350
            [1] => jean-pierre
            [2] => laborde
            [3] => m
            [4] => direction
            [5] => 1999-12-09
            [6] => 2500
        )

    [1] => Array
        (
            [0] => 7388
            [1] => clement
            [2] => gallot
            [3] => m
            [4] => commercial
            [5] => 2015-05-15
            [6] => 2500
        )
)

```

etc...

Cette fois-ci nous avons modifié la requête SQL de départ pour sélectionner tous les enregistrements (tous les employés).

Habituellement nous avons un nouveau tableau ARRAY représentant chaque ligne d'enregistrement.

Avec fetch\_all cela sera exactement pareil sauf que nous aurons un tableau supplémentaire englobant tous les autres tableaux.

Pour résumé :

fetch\_array retourne une ligne de résultats MySQL sous la forme d'un tableau doté de clés nominatives et aussi numériques.

fetch\_row retourne une ligne de résultats MySQL sous la forme d'un tableau doté de clés numériques.

fetch\_assoc retourne une ligne de résultats MySQL sous la forme d'un tableau doté de clés nominatives. Cette solution est souvent préconisée.

fetch\_object retourne une ligne de résultats MySQL sous la forme d'un objet avec comme attributs les champs de la table sur laquelle la requête a été effectuée.

La classe **Mysqli** :

mysqli — La classe mysqli

mysqli::\$affected\_rows — Retourne le nombre de lignes affectées par la dernière opération MySQL

mysqli::autocommit — Active ou désactive le mode auto-commit

mysqli::begin\_transaction — Démarrer une transaction

mysqli::change\_user — Change l'utilisateur de la connexion spécifiée

mysqli::character\_set\_name — Retourne le jeu de caractères courant pour la connexion

mysqli::\$client\_info — Récupère les informations du client MySQL

mysqli::\$client\_version — Retourne la version du client MySQL sous la forme d'une chaîne de caractères

mysqli::close — Ferme une connexion

mysqli::commit — Valide la transaction courante

mysqli::\$connect\_errno — Retourne le code d'erreur de la connexion MySQL

mysqli::\$connect\_error — Retourne le message d'erreur de connexion MySQL

mysqli::\_\_construct — Ouvre une connexion à un serveur MySQL

mysqli::debug — Effectue des actions de débogage

mysqli::dump\_debug\_info — Écrit les informations de débogage dans les logs

mysqli::\$errno — Retourne le dernier code d'erreur produit

mysqli::\$error\_list — Retourne une liste d'erreurs depuis la dernière commande exécutée

mysqli::\$error — Retourne une chaîne décrivant la dernière erreur

mysqli::\$field\_count — Retourne le nombre de colonnes pour la dernière requête

mysqli::get\_charset — Retourne un objet représentant le jeu de caractères

mysqli::get\_client\_info — Récupère des informations sur le client MySQL

mysqli\_get\_client\_stats — Retourne des statistiques sur le client, par processus

mysqli\_get\_client\_version — Retourne la version du client MySQL sous forme d'un entier

mysqli::get\_connection\_stats — Retourne des statistiques sur la connexion

mysqli::\$host\_info — Retourne une chaîne contenant le type de connexion utilisée

mysqli::\$protocol\_version — Retourne la version du protocole MySQL utilisé

mysqli::\$server\_info — Retourne la version du serveur MySQL

mysqli::\$server\_version — Retourne un entier représentant la version du serveur MySQL

mysqli::get\_warnings — Lit le résultat de SHOW WARNINGS

mysqli::\$info — Retourne des informations à propos de la dernière requête exécutée

mysqli::init — Initialise MySQLi et retourne une ressource à utiliser avec mysqli\_real\_connect()

mysqli::\$insert\_id — Retourne l'identifiant automatiquement généré par la dernière requête

mysqli::kill — Demande au serveur de terminer un thread MySQL

mysqli::more\_results — Vérifie s'il y a d'autres jeux de résultats MySQL disponibles

mysqli::multi\_query — Exécute une requête MySQL multiple

mysqli::next\_result — Prépare le prochain résultat d'une requête multiple

mysqli::options — Définit les options

mysqli::ping — Ping la connexion au serveur et reconnecte si elle n'existe plus

mysqli::poll — Vérifie l'état de la connexion

mysqli::prepare — Prépare une requête SQL pour l'exécution

mysqli::query — Exécute une requête sur la base de données

mysqli::real\_connect — Ouvre une connexion à un serveur MySQL

mysqli::real\_escape\_string — Protège les caractères spéciaux d'une chaîne pour l'utiliser dans une requête SQL, en prenant en compte le jeu de caractères

courant de la connexion  
mysqli::real\_query – Exécute une requête SQL  
mysqli::reap\_async\_query – Lit un résultat pour une requête asynchrone  
mysqli::refresh – Rafraîchit  
mysqli::release\_savepoint – Supprime le point de sauvegarde nommé du jeu des points de sauvegarde de la transaction courante  
mysqli::rollback – Annule la transaction courante  
mysqli::rpl\_query\_type – Retourne le type de requête RPL  
mysqli::savepoint – Définit un point de sauvegarde nommé de la transaction  
mysqli::select\_db – Sélectionne une base de données par défaut pour les requêtes  
mysqli::send\_query – Envoie la requête et retourne  
mysqli::set\_charset – Définit le jeu de caractères par défaut du client  
mysqli::set\_local\_infile\_default – Rétablit le gestionnaire par défaut pour la commande LOAD LOCAL INFILE  
mysqli::set\_local\_infile\_handler – Définit une fonction de rappel pour la commande LOAD DATA LOCAL INFILE  
mysqli::\$sqlstate – Retourne l'erreur SQLSTATE de la dernière opération MySQL  
mysqli::ssl\_set – Utilisée pour établir une connexion sécurisée avec SSL  
mysqli::stat – Obtient le statut courant du système  
mysqli::stmt\_init – Initialise une commande MySQL  
mysqli::store\_result – Transfère un jeu de résultats à partir de la dernière requête  
mysqli::\$thread\_id – Retourne l'identifiant du thread pour la connexion courante  
mysqli::thread\_safe – Indique si le support des threads est activé ou pas  
mysqli::use\_result – Initialise la récupération d'un jeu de résultats  
mysqli::\$warning\_count – Retourne le nombre d'avertissements générés par la dernière requête

#### La classe Mysqli\_result :

mysqli\_result – La classe mysqli\_result  
mysqli\_result::\$current\_field – Récupère la position courante d'un champ dans un pointeur de résultat  
mysqli\_result::data\_seek – Déplace le pointeur interne de résultat  
mysqli\_result::fetch\_all – Lit toutes les lignes de résultats dans un tableau  
mysqli\_result::fetch\_array – Retourne une ligne de résultat sous la forme d'un tableau associatif, d'un tableau indexé, ou les deux  
mysqli\_result::fetch\_assoc – Récupère une ligne de résultat sous forme de tableau associatif  
mysqli\_result::fetch\_field\_direct – Récupère les métadonnées d'un champ unique  
mysqli\_result::fetch\_field – Retourne le prochain champs dans le jeu de résultats  
mysqli\_result::fetch\_fields – Retourne un tableau d'objets représentant les champs dans le résultat  
mysqli\_result::fetch\_object – Retourne la ligne courante d'un jeu de résultat sous forme d'objet  
mysqli\_result::fetch\_row – Récupère une ligne de résultat sous forme de tableau indexé  
mysqli\_result::\$field\_count – Récupère le nombre de champs dans un résultat  
mysqli\_result::field\_seek – Déplace le pointeur de résultat sur le champ spécifié  
mysqli\_result::free – Libère la mémoire associée à un résultat  
mysqli\_result::\$lengths – Retourne la longueur des colonnes de la ligne courante du jeu de résultats  
mysqli\_result::\$num\_rows – Retourne le nombre de lignes dans un résultat

Source : <http://php.net/manual/fr/book.mysql.php>

[Pdo](#) (Php Data Object) est une classe prédéfinie de PHP (par extension) permettant la connexion et l'exécution de requête sur le SGBD MySQL en PHP.

Cet extension est plus populaire que Mysqli, notamment pour son style orienté objet et la possibilité de migrer d'un SGBD à un autre sans avoir à réécrire la totalité du code.

Ensuite, il vous faut aussi un fichier où vous pourrez écrire le code (script).

Pour cela, rien de plus facile puisque'il s'agit de créer un simple fichier nommé "pdo.php" sur le serveur local (repertoire /www/).

Avant de pouvoir formuler des requêtes, il faut absolument relier la base de données à votre fichier (script).

Pour cela, nous devons le préciser en haut de fichier :

```
pdo.php
```

```
<?php  
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
```

#### Quelques Explications :

**Pdo** est une classe prédéfinie de PHP permettant de se connecter à une base de données et de formuler des requêtes SQL.

Les arguments/paramètres entrants vont directement dans le constructeur de la classe (méthode exécutée par défaut).

Les arguments/paramètres correspondent à :

- host = nom du serveur = localhost
- dbname = nom de la base de données = entreprise
- pseudo = root
- mot de passe = (sous wamp, le mot de passe est vide - sous mamp, il sera nécessaire d'écrire "root")
- Erreur mode activé pour faire apparaître d'éventuelles erreurs de requête SQL

Techniquement, nous pouvons dire que \$pdo est une variable représentant 1 objet issu de la classe Pdo.

Humainement, nous pouvons dire que \$pdo est une variable permettant d'être connecté à la base et de formuler des requêtes.

Pour mieux connaître la classe Pdo, vous pouvez consulter la documentation officielle : [Classe Pdo](#)

Vous pouvez aussi demander l'affichage des propriétés avec l'instruction print\_r et aussi la fonction prédéfinie get\_class\_methods() :

```
pdo.php
```

```
<?php  
$pdo = new Pdo("localhost", "root", "", "entreprise");  
echo '<pre>'; print_r($pdo); echo '</pre>';  
echo '<pre>'; print_r(get_class_methods($pdo)); echo '</pre>';
```

Nous aurons l'occasion d'utiliser plusieurs des propriétés et méthodes présentes dans cette liste.

### PDO : Ecrire une première requête SQL

Pour formuler une première requête SQL dans une page web, nous aurons besoin de passer par l'objet \$pdo et d'utiliser la méthode query() ou exec() par exemple :

```
pdo.php
```

```
<?php  
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));  
//---  
echo '<pre>'; print_r($pdo); echo '</pre>';  
echo '<pre>'; print_r(get_class_methods($pdo)); echo '</pre>';
```

### PDO : Insérer des données

Voici le code permettant d'insérer des données avec la classe Pdo :

```
pdo.php
```

```
<?php
```

```

$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
// Insertion :
$result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique',
'2015-07-30', 5000);");
echo $result . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----

```

Profitez en pour vous attribuer un bon salaire, c'est vous qui décidez :p.

**Exec** (ou Query) permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable \$result juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) FALSE
- Dans le cas d'une bonne requête SQL, \$result contiendra : (int) 1 ou N selon le nombre de résultat(s) touché(s) par la requête

\$result contient le nombre de résultat(s) touché(s) par la requête grâce à l'utilisation de la méthode exec().

Nous aurions également pu utiliser la méthode query() pour cette requête mais nous n'aurions pas eu accès aux nombres de résultat(s) touchés par la requête.

résultat 1 enregistrement(s) affecté(s) par la requête INSERT.

Après l'INSERT, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé à bien été ajouté !

## PDO : Modifier des données

Voici le code permettant de modifier des données avec la classe Pdo :

Pdo.php

```

<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
// Insertion (INSERT) :
$result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique',
'2015-07-30', 5000);");
echo $result . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $pdo->exec("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $result . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----

```

Nous modifions le salaire de l'employé 802 (Damien Durand), initialement à 2250, il passera à 2500.

**Exec** (ou Query) permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable \$result juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) FALSE
- Dans le cas d'une bonne requête SQL, \$result contiendra : (int) 1 ou N selon le nombre de résultat(s) touché(s) par la requête

\$result contient le nombre de résultat(s) touché(s) par la requête grâce à l'utilisation de la méthode exec().

Nous aurions également pu utiliser la méthode query() pour cette requête mais nous n'aurions pas eu accès aux nombres de résultat(s) touché(s) par la requête.

résultat 1 enregistrement(s) affecté(s) par la requête UPDATE.

Avant et après l'UPDATE, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé à bien été modifié !

## PDO : Supprimer des données

Voici le code permettant de supprimer des données avec la classe Pdo :

pdo.php

```

<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
// Insertion (INSERT) :
$result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique',
'2015-07-30', 5000);");
echo $result . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $pdo->exec("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $result . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----

```

```
echo $result . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----
// Suppression (DELETE) :
$result = $pdo->exec("DELETE FROM employes WHERE id_employes = 388");
echo $result . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
//-----
```

Nous supprimons l'employé 388 (Clement Gallet).

**Query** permet de formuler une requête SQL dans le code PHP.

Nous avons prévu une variable \$result juste avant pour obtenir un retour :

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) FALSE
  - Dans le cas d'une bonne requête SQL, \$result contiendra : (int) 1 ou N selon le nombre de résultat(s) touché(s) par la requête

**\$result** contient le nombre de résultat(s) touché(s) par la requête grâce à l'utilisation de la méthode exec().

Nous aurions également pu utiliser la méthode `query()` pour cette requête mais nous n'aurions pas eu accès aux nombre de résultat(s) touché(s) par la requête.

résultat 1 enregistrement(s) affecté(s) par la requête DELETE.

Avant et après la requête DELETE, allez voir le contenu de votre table (en passant par la console Mysql ou le gestionnaire PhpMyAdmin), vous verrez que l'employé a bien été supprimé !

**PDO : Sauvegarder les informations d'un formulaire dans une base de données (SQL/PHP)**

Maintenant que l'on sait échanger des informations entre la base de données et la page web, nous pourrions travailler avec des formulaires !

La création d'un formulaire permettant d'ajouter des employés serait judicieux afin qu'une personne non familière des langages informatiques puisse mettre à jour la base de données facilement.

Création d'un nouveau fichier :  formulaire-pdo.php

```
formulaire-pdo.php

<?php
$pdo = new Pdo("localhost", "root", "", "entreprise");
if($_POST)
{
    $result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('$_POST[prenom]', '$_POST[nom]', '$_POST[sexe]', '$_POST[service]', '$_POST[date_embauche]', '$_POST[salaire]')");
    echo '<div style="background: green; padding: 5px;">l'employé a bien été ajouté</div>';
}
?>

<form method="post">
    <label for="prenom">Prenom</label><br>
    <input type="text" name="prenom" placeholder="prenom" id="prenom" required=""><br><br>
    <label for="nom">Nom</label><br>
    <input type="text" name="nom" placeholder="nom" id="nom" required=""><br><br>
    <label for="sexe">Sexe</label><br>
    Homme <input type="radio" name="sexe" placeholder="sexe" id="sexe" value="m" checked="" - 
    Femme <input type="radio" name="sexe" placeholder="sexe" id="sexe" value="f"> <br><br>
    <label for="service">Service</label><br>
    <input type="text" name="service" placeholder="service" id="service"><br><br>
    <label for="date_embauche">Date d'embauche <i>(Format: AAAA-MM-JJ)</i></label><br>
    <input type="text" name="date_embauche" placeholder="ex 2015-07-30" id="date_embauche"><br>
    <label for="salaire">Salaire</label><br>
    <input type="text" name="salaire" placeholder="salaire" id="salaire"><br><br>
    <input type="submit"><br><br>
</form>
```

### Explications

Lorsque nous arrivons la première fois sur la page, le code PHP se trouvant dans le IF est ignoré puisque personne n'a validé le formulaire.

Lorsque l'internaute clique sur le bouton submit, cela recharge la page (le script), et la partie de code PHP dans le IF est exécutée pour enregistrer l'employé qu'on nous aura posté.

Bien entendu, ce formulaire peut être amélioré.

Dans l'idéal, il faudrait réaliser des contrôles (est-ce que tous les champs ont été correctement remplis, etc.), le sécuriser (se protéger d'éventuelles attaques), et augmenter l'expérience utilisateur (en proposant par exemple 1 calendrier pour sélectionner une date d'embauche).

En l'état, ce formulaire est fonctionnel et c'est bien là l'essentiel pour l'apprentissage de notre cours.

#### Résultat

l'employé a bien été ajouté

Prenom  
prenom

Nom  
nom

Sexe  
Homme  - Femme

Service  
service

Date d'embauche (Format: AAAA-MM-JJ)  
ex: 2015-07-30

Salaire  
salaire

**Envoyer**

119

## PDO : Selectionner et afficher des données

### PDO : Selectionner et afficher 1 enregistrement

Pour afficher dans la page web les données d'un enregistrement à l'intérieur de la base, le code sera légèrement différent.

Voici le code permettant d'effectuer un affichage de données avec la classe Pdo :

```
Pdo.php

<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
// Insertion (INSERT) :
$result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000);");
echo $result . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $pdo->exec("UPDATE employes SET salaire = 2500 WHERE id_employes = 802");
echo $result . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----
// Suppression (DELETE) :
$result = $pdo->exec("DELETE FROM employes WHERE id_employes = 388");
echo $result . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
//-----
// Affichage (SELECT) :
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch(PDO::FETCH_ASSOC);
echo "<pre>"; print_r($employe); echo "</pre>";
echo "Bonjour je suis $employe[prenom] $employe[nom] du service $employe[service]<br>";
//-----
```

#### Query

Nous sélectionnons un employé en particulier (Julien, à vous de vérifier s'il est bien présent dans votre table, sinon il faudra changer le prénom ou sélectionner directement par un id existant).

Nous obtenons l'enregistrement demandé dans la variable \$result prévue pour récupérer le retour de la requête SQL.

## \$result

Nous avons prévu une variable \$result juste avant la requête pour obtenir un retour.

Par défaut, le retour de la requête SQL crée un nouvel objet, issu de la classe PDOStatement

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) false
- Dans le cas d'une bonne requête SQL, \$result contiendra : (object) PDOStatement Si la requête est bonne, vous obtiendrez un autre objet issu d'une autre classe (PDOStatement) !

## fetch(PDO::FETCH\_ASSOC)

Le résultat (ressource) sous forme d'objet PDOStatement n'est pas exploitable en l'état.

Pour accéder aux données que nous avons sélectionnées précédemment, nous avons besoin d'utiliser la méthode (fonction) fetch(PDO::FETCH\_ASSOC) de la classe PDOStatement (\$result->fetch(PDO::FETCH\_ASSOC)). La méthode fetch(PDO::FETCH\_ASSOC) permet de rendre un résultat exploitable sous forme de tableau ARRAY associatif.

## \$employe

\$employe est donc un tableau ARRAY associatif (associatif = avec des clés nominatives).

Nous pouvons donc afficher notre tableau ARRAY par l'intermédiaire d'un print\_r ou d'un echo

résultat

```
Array
(
    [id_employees] => 7699
    [prenom] => julien
    [nom] => cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 1390
)
```

Bonjour je suis Julien Cottet du service secretariat

Avec cette requête de SELECTION, nous avons pu afficher 1 enregistrement de la table employes.

## PDO : Selectionner et afficher plusieurs enregistrements

Pour afficher dans la page web les données de PLUSIEURS enregistrements à l'intérieur de la base, le code sera encore différent.

Voici le code permettant d'effectuer un affichage de données avec la classe Pdo :

```
pdo.php

<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
// Insertion (INSERT) :
$result = $pdo->exec("INSERT INTO employes (prenom, nom, sexe, service, date_embauche, salaire) VALUES ('prenom', 'nom', 'm', 'informatique', '2015-07-30', 5000);");
echo $result . ' enregistrement(s) affecté(s) par la requête INSERT<br>';
//-----
// Modification (UPDATE) :
$result = $pdo->exec("UPDATE employes SET salaire = 2500 WHERE id_employees = 802");
echo $result . ' enregistrement(s) affecté(s) par la requête UPDATE<br>';
//-----
// Suppression (DELETE) :
$result = $pdo->exec("DELETE FROM employes WHERE id_employees = 388");
echo $result . ' enregistrement(s) affecté(s) par la requête DELETE<br>';
//-----
// Affichage (SELECT) :
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch(PDO::FETCH_ASSOC);
echo "<pre>"; print_r($employe); echo "</pre>";
echo "Bonjour je suis $employe[prenom] $employe[nom] du service $employe[service]<br>";
//-----
// Affichage (SELECT) :
$result = $pdo->query("SELECT * FROM employes");
while($employe = $result->fetch(PDO::FETCH_ASSOC))
{
    echo "<pre>"; print_r($employe); echo "</pre>";
    echo "Bonjour je suis $employe[prenom] $employe[nom] du service $employe[service]";
}
echo $result->rowCount() . ' enregistrement(s) récupéré(s) par la requête SELECT<br>';
//-----
```

## Query

Nous sélectionnons tous les employés.

Nous obtenons les enregistrements demandés dans la variable \$result prévue pour récupérer le retour de la requête SQL.

## \$result

Nous avons prévu une variable \$result juste avant la requête pour obtenir un retour.

Par défaut, le retour de la requête SQL crée un nouvel objet, issu de la classe PDOSTATEMENT

- Dans le cas d'une erreur de requête SQL, \$result contiendra : (boolean) false
- Dans le cas d'une bonne requête SQL, \$result contiendra : (object) PDOSTATEMENT *Si la requête est bonne, vous obtiendrez un autre objet issu d'une autre classe (PDOSTATEMENT) !*

## While

while est une boucle permettant d'effectuer une répétition.

Pourquoi effectuer une répétition ?

Parce que nous avons plusieurs lignes d'enregistrements à récupérer. Il est donc nécessaire de répéter le traitement fetch(PDO::FETCH\_ASSOC) afin de rendre le résultat exploitable sous forme d'array.

La boucle While permet d'afficher chaque ligne de la table (tant que l'on possède des enregistrements, on les affiche).

votre requête sort plusieurs résultats ? : une boucle !

votre requête ne doit sortir qu'un seul et unique résultat ? : Pas de boucle !

votre requête ne sort qu'un seul résultat à l'instant T mais peut potentiellement en sortir plusieurs plus tard ? : une boucle !

## fetch(PDO::FETCH\_ASSOC)

Le résultat (ressource) sous forme d'objet PDOSTATEMENT n'est pas exploitable en l'état.

Pour accéder aux données que nous avons sélectionnées précédemment, nous avons besoin d'utiliser la méthode (fonction) fetch(PDO::FETCH\_ASSOC) de la classe PDOSTATEMENT (`fetch(PDO::FETCH_ASSOC)`). La méthode fetch(PDO::FETCH\_ASSOC) permet de rendre un résultat exploitable sous forme de tableau ARRAY associatif.

Avec la présence de la boucle while, nous obtiendrons un nouveau tableau ARRAY dans la variable \$employe, à chaque tour de boucle.

! Il y aura pas un tableau ARRAY avec tous les enregistrements à l'intérieur mais bien un nouveau tableau ARRAY par enregistrement, un array par employé !

## \$employe

\$employe est donc un tableau ARRAY associatif (associatif = avec des clés nominatives).

Nous pouvons donc afficher notre tableau ARRAY par l'intermédiaire d'un `print_r` ou d'un `echo`

`rowCount()` est une méthode de l'objet PDOSTATEMENT permettant d'observer le nombre d'enregistrements récupérés et affichés par une requête.

## résultat

```
Array
(
    [id_employes] => 7350
    [prenom] => jean-pierre
    [nom] => laborde
    [sexe] => m
    [service] => direction
    [date_embauche] => 1999-12-09
    [salaire] => 5000
)
```

Bonjour je suis jean-pierre laborde du service direction

```
Array
(
    [id_employes] => 7388
    [prenom] => clement
    [nom] => gallot
```

```

[sexe] => m
[service] => commercial
[date_embauche] => 2015-05-15
[salaire] => 2300
)

```

Bonjour je suis clement gallet du service commercial

...etc...

Avec cette requête de SELECTION, nous avons pu afficher tous les enregistrements de la table employes.

### PDO : Une table SQL sous forme de table HTML

Voici le code qui permettrait de reproduire la table sql (de la base de données) dans une table html (dans la page web).

Table SQL						
<pre>&lt;?php \$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=&gt;PDO::ERRMODE_EXCEPTION));  \$résultat = \$pdo-&gt;query("SELECT * FROM employes");  echo "&lt;table border='1'&gt; &lt;tr&gt;"; for(\$i=0; \$i&lt;\$résultat-&gt;columnCount(); \$i++) {     \$colonne = \$résultat-&gt;getColumnMeta(\$i);     echo '&lt;th&gt;' . \$colonne['name'] . '&lt;/th&gt;'; } echo "&lt;/tr&gt;";  while (\$ligne = \$résultat-&gt;fetch(PDO::FETCH_ASSOC)) {     echo '&lt;tr&gt;';     foreach (\$ligne as \$indice =&gt; \$information)     {         echo '&lt;td&gt;' . \$information . '&lt;/td&gt;';     }     echo '&lt;/tr&gt;'; } echo '&lt;/table&gt;'; ?&gt;</pre>						

### Résultat

id_employes	prenom	nom	sexe	service	date_embauche	salaire
350	Jean-pierre	Laborde	m	direction	1999-12-09	5000
388	Clement	Gallet	m	commercial	2000-01-15	2300
415	Thomas	Winter	m	commercial	2000-05-03	3550
417	Chloe	Dubar	f	production	2001-09-05	1900
491	Elodie	Fellier	f	secretariat	2002-02-22	1600
509	Fabrice	Grand	m	comptabilite	2003-02-20	1900
547	Melanie	Collier	f	commercial	2004-09-08	3100
592	Laura	Blanchet	f	direction	2005-06-09	4500
627	Guillaume	Miller	m	commercial	2006-07-02	1900
655	Celine	Perrin	f	commercial	2006-09-10	2700
699	Julien	Cottet	m	secretariat	2007-01-18	1390
701	Mathieu	Vignal	m	informatique	2008-12-03	2000
739	Thierry	Desprez	m	secretariat	2009-11-17	1500
780	Amandine	Thoyer	f	communication	2010-01-23	1500
802	Damien	Durand	m	informatique	2010-07-05	2250
854	Daniel	Chevel	m	informatique	2011-09-28	1700

876	Nathalie	Martin	f	juridique	2012-01-12	3200
900	Benoit	Lagarde	m	production	2013-01-03	2550
933	Emilie	Sennard	f	commercial	2014-09-11	1800
990	Stephanie	Lafaye	f	assistant	2015-06-02	1775

### Toutes les explications

Nous allons tâcher d'expliquer chaque ligne pour une compréhension optimale.

```
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', "", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
```

Cette ligne représente la connexion à notre base de données avec les informations suivantes : nom du serveur, pseudo, mot de passe, nom de la base de données.

Humainement, \$pdo représente la variable permettant de communiquer et de gérer des échanges avec la base de données.

Techniquement, \$pdo est un objet issu de la classe Pdo.

```
$résultat = $pdo->query("SELECT * FROM employes");
```

Cette ligne permet de formuler une requête SQL afin de sélectionner tous les employés de la table via la méthode query() de l'objet \$pdo.

Nous récupérons le résultat de la requête dans la variable \$résultat.

```
echo '<table border="1"> <tr>';
```

Nous créons une table html avec une première ligne (<tr>)

```
for($i=0; $i < $résultat->columnCount(); $i++)
```

Nous aurons besoin d'enclencher des tours de boucle, pour savoir combien de tours nous devons effectuer, la méthode columnCount() nous permettra de savoir combien il y a de champs/colonnes au total.

La boucle for est donc présente pour répéter l'action d'affichage et le traitement puisqu'il y a plusieurs champs/colonnes à afficher

```
$meta = $résultat->getColumnMeta($i);
```

Pour consulter le nom des champs/colonnes de la table SQL, nous utiliserons la méthode getColumnMeta() qui nous permettra de récolter des informations sur les champs/colonnes

\$colonne est 1 array car la méthode getColumnMeta() que nous utilisons est prévue pour récolter des informations sur les champs/colonnes d'une table SQL et de faire un retour sous forme d'array.

En 1 mot : getColumnMeta() est une méthode de l'objet PDOStatement qui nous permet de récupérer des informations sur les champs sous forme d'array.

```
echo '<th>' . $colonne['name'] . '</th>';
```

Nous inscrivons des <th> (en html) pour commencer l'affichage des entêtes de la table.

A l'intérieur des balises html <th>, nous avons besoin d'inscrire les noms des champs.

Nous passons donc par la variable \$colonne (remplie précédemment avec getColumnMeta()) et nous demandons le name du champ.

Pour voir les autres informations auxquelles nous pouvons avoir accès, il est possible d'effectuer un print\_r(\$colonne); ou de [consulter la documentation officielle](#) à ce sujet !

A chaque tour de boucle, le nom du champ sera affiché.

```
echo "</tr>";
```

Nous terminons l'affichage de la 1ère ligne en fermant la balise </tr>.

```
while ($ligne = $resultat->fetch(PDO::FETCH_ASSOC))
```

fetch\_assoc permet de traiter le résultat et de le rendre exploitable sous forme de tableau ARRAY (nous utilisons la méthode fetch\_assoc de l'objet mysqli\_result).

\$ligne représente un tableau array associatif contenant le résultat.

while permet de répéter cette action tant qu'il y a des résultats et de passer à la ligne d'enregistrement suivante pour faire avancer les traitements.

```
echo "<tr>";
```

Nous préparons l'affichage d'une prochaine ligne en ouvrant la balise <tr>.

```
foreach($ligne as $indice => $information)
```

La boucle foreach va nous permettre de parcourir notre tableau array représenté par la variable \$ligne et d'afficher chacune des informations contenues à l'intérieur.

Chaque employé possède 7 informations : id\_employes, prenom, nom, sexe, service, date\_embauche, salaire.

La boucle while est présente pour traiter chaque employé (et avancer sur l'employé suivant), tandis que, la boucle foreach est présente pour traiter et afficher chaque information pour chaque employé.

Lorsque nous faisons 1 tour dans la boucle while, cela entraîne 7 tours dans la boucle foreach.

S'il y a 20 employés, la boucle while fera 20 tours et la boucle foreach 140 tours (20 employés x 7 informations).

```
echo '<td>' . $information . '</td>';
```

Cette ligne (présente à l'intérieur de la boucle foreach) permet d'afficher chaque information d'un employé via \$information dans une nouvelle balise <td>.

```
echo "</tr>";
```

Nous terminons l'affichage de la ligne en fermant la balise </tr>.

```
echo "</table>";
```

Nous terminons l'affichage de notre table en fermant la balise </table>.

Vous pouvez garder ce code en référence pour vos besoins. Il suffira juste d'adapter le nom de la table sur laquelle vous effectuez une requête SQL. C'est une première approche, ne prenez pas peur, nous reviendrons dessus jusqu'à ce que chaque mot, chaque détail, soit compris !

Pour résumé :

#### PDO et PDOSTATEMENT

\$pdo représente un objet [1] issu de la classe prédefinie Pdo

Lorsqu'on exécute une requête de SELECTION via la méthode QUERY() sur l'objet Pdo :

- En cas de succès : On obtient un autre objet[2] issu de la classe PDOSTATEMENT. Cet objet à donc des méthodes et des propriétés différentes !
- En cas d'échec : On obtient un boolean False

Lorsqu'on exécute une requête INSERT/UPDATE/DELETE via la méthode QUERY() sur l'objet Pdo, c'est plus simple :

- En cas de succès : On obtient un boolean TRUE.
- En cas d'échec : On obtient un boolean FALSE

Pour comprendre techniquement dans la précision, n'hésitez pas à lancer var\_dump(\$pdo) ou var\_dump(\$result) ;

#### While : boucle ou pas boucle ?

- S'il n'y a qu'une seule ligne d'enregistrement à récupérer dans le résultat, nous ne ferons pas de boucle while.
- S'il y a plusieurs lignes d'enregistrements à récupérer dans le résultat, il faut prévoir une boucle while.

Techniquement, la boucle while permet de faire avancer le curseur sur la ligne suivante dans la table.

#### rowCount()

Dans le cas d'une requête de SELECTION, nous utiliserons rowCount() pour voir le nombre d'enregistrements récupérés (et affichés) par une requête.

#### Fetch

La méthode fetch(PDO::FETCH\_ASSOC) permet de traiter les résultats afin d'obtenir un tableau ARRAY.

Cette étape est indispensable si l'on souhaite produire un affichage par la suite.

D'autres méthodes de traitement existent.

#### PDO::FETCH\_ASSOC

##### PDO::FETCH\_ASSOC

```
<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
?>
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch(PDO::FETCH_ASSOC);
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe[prenom] . '<br>';
?>
```

#### Résultat

Avec print\_r :

```
Array
(
    [id_employes] => 699
    [prenom] => Julien
    [nom] => Cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 1390
)
```

Avec une représentation plus compréhensible :

clé	valeur
id_employees	699
prenom	Julien
nom	Cottet
sexe	m
service	secretariat
date_embauche	2007-01-18
salaire	1390

Nous pouvons dire que PDO::FETCH\_ASSOC a l'avantage de donner des clés associatives/nominatives (correspondant au nom des champs/colonnes dans la table SQL) au tableau ARRAY.

#### PDO::FETCH\_ROW

PDO::FETCH\_ROW

```
<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', " ", array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch(PDO::FETCH_ROW);
echo $employe[0] . '<br>';
echo "<pre>"; print_r($employe); echo "</pre>";
?>
```

#### Résultat

Avec `print_r` :

```
Array
(
    [0] => 699
    [1] => Julien
    [2] => Cottet
    [3] => m
    [4] => secretariat
    [5] => 2007-01-18
    [6] => 1390
)
```

Avec une représentation plus compréhensible :

clé	valeur
0	699
1	Julien
2	Cottet
3	m
4	secretariat
5	2007-01-18
6	1390

Nous pouvons dire que PDO::FETCH\_ROW donne des clés numériques (correspondant à l'ordre des champs/colonnes dans la table SQL) au tableau

ARRAY.

#### PDO::FETCH\_ARRAY

```
PDO::FETCH_ARRAY  
?  
<?php  
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', " , array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));  
//---  
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");  
$employe = $result->fetch(PDO::FETCH_ARRAY);  
echo "<pre>"; print_r($employe); echo "</pre>";  
echo $employe['prenom'] . $employe[0] . '<br>';  
?>
```

#### Résultat

Avec `print_r` :

```
Array  
(  
    [0] => 699  
    [id_employes] => 699  
    [1] => Julien  
    [prenom] => Julien  
    [2] => Cottet  
    [nom] => Cottet  
    [3] => m  
    [sexe] => m  
    [4] => secretariat  
    [service] => secretariat  
    [5] => 2007-01-18  
    [date_embauche] => 2007-01-18  
    [6] => 1390  
    [salaire] => 1390  
)
```

Avec une représentation plus compréhensible :

clé	valeur
0	699
id_employes	699
1	Julien
prenom	Julien
2	Cottet
nom	Cottet
3	m
sexe	m
4	secretariat
service	secretariat
5	2007-01-18
date_embauche	2007-01-18
6	1390
salaire	1390

Nous pouvons dire que PDO::FETCH\_ARRAY est une combinaison du PDO::FETCH\_ASSOC et du PDO::FETCH\_ROW, cela donne des clés numériques (correspondant à l'ordre des champs/colonnes dans la table SQL) et des clés associatives/nominatives (correspondant au nom des champs/colonnes dans la table SQL) au tableau ARRAY.

#### PDO::FETCH\_OBJECT

#### PDO::FETCH\_OBJECT

```
<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
$result = $pdo->query("SELECT * FROM employes WHERE prenom='julien'");
$employe = $result->fetch(PDO::FETCH_OBJECT);
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe->prenom . '<br>';
?>
```

#### Résultat

```
stdClass Object
(
    [id_employes] => 7699
    [prenom] => julien
    [nom] => cottet
    [sexe] => m
    [service] => secretariat
    [date_embauche] => 2007-01-18
    [salaire] => 2500
)
```

PDO::FETCH\_OBJECT ne produit pas un tableau mais un objet (de la StdClass, class standard en PHP). Les champs/colonnes de la tables SQL deviennent le nom des propriétés de l'objet.

Pour accéder à l'une d'entre-elles, il sera nécessaire d'utiliser la syntaxe suivante : \$employe->prenom.

#### PDO::FETCH\_ALL

#### PDO::FETCH\_ALL

```
<?php
$pdo = new PDO('mysql:host=localhost;dbname=entreprise', 'root', '', array(PDO::ATTR_ERRMODE=>PDO::ERRMODE_EXCEPTION));
//-----
$result = $pdo->query("SELECT * FROM employes");
$employe = $result->fetch(PDO::FETCH_ALL);
echo "<pre>"; print_r($employe); echo "</pre>";
echo $employe->prenom . '<br>';
?>
```

#### Résultat

```
Array
(
    [0] => Array
        (
            [0] => 7350
            [1] => jean-pierre
            [2] => laborde
            [3] => m
            [4] => direction
            [5] => 1999-12-09
            [6] => 2500
        )
)
```

```

[1] => Array
(
    [0] => 7388
    [1] => clement
    [2] => gallet
    [3] => m
    [4] => commercial
    [5] => 2015-05-15
    [6] => 2500
)
)
etc...

```

Cette fois-ci nous avons modifié la requête SQL de départ pour sélectionner tous les enregistrements (tous les employés).

Habituellement nous avons un nouveau tableau ARRAY représentant chaque ligne d'enregistrement.

Avec PDO::FETCH\_ALL cela sera exactement pareil sauf que nous aurons un tableau supplémentaire englobant tous les autres tableaux.

#### CHOIX DU CONNECTEUR DE BASE DE DONNEES

Nous pouvons utiliser librement la classe Mysqli ou la classe PDO pour échanger des données avec la base.

## Cas concret : Création d'un espace de dialogue

### Cas concret : Réalisation d'un espace de dialogue

Maintenant que nous avons pris des forces en PHP, nous allons pouvoir créer des pages web utiles et concrètes.

Premier exemple concret, 1 espace de dialogue / commentaire (ex livre d'or et pouvant aussi être apparenté à 1 tchat).

Nous pouvons retrouver ce type de fonctionnalité sur FaceBook, YouTube et de nombreux autres sites permettant à leurs internautes de dialoguer en direct.

Quelles sont les différentes étapes afin de pouvoir créer cela ?

Tout d'abord, nous aurons besoin d'une base de données afin que les commentaires puissent être enregistrés dans une table.

Nous aurons également besoin d'une page web avec un formulaire afin de déposer des commentaires.

121

### Etape 1 : Modélisation et création de la base de données, table et champs.



Base de données : dialogue

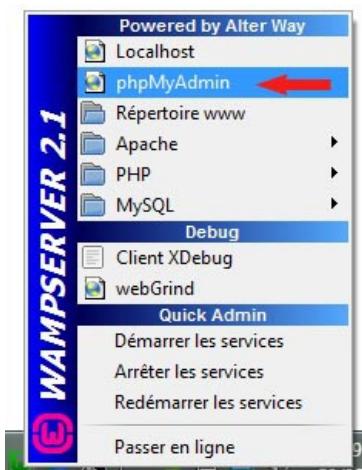


Table : commentaire

Champ	Type	Taille	Spécificité
id_commentaire	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)
pseudo	VARCHAR	20	-
message	TEXT	-	-
date_enregistrement	DATETIME	-	-

Pour créer cette base de données, vous pouvez également passer par [le gestionnaire de base de données PhpMyAdmin](#) :

Accès à PhpMyAdmin :



Création d'une nouvelle base de données :

A screenshot of the phpMyAdmin interface. At the top, there's a navigation bar with tabs: 'localhost' (selected), 'Bases de données' (highlighted with a red arrow), 'SQL', 'État', 'Variables', and 'Jeux de caractères'. Below the navigation bar, there's a toolbar with icons for 'Home', 'SQL', 'Search', and 'Help'. The main area has a title 'Actions' and a sub-section 'MySQL localhost'. It shows a 'Créer une base de données' button with a red arrow pointing to it. The 'dialogue' database is selected. The 'Interface' section includes settings for 'Langue - Language' (French), 'Thème / Style' (Original), 'Couleur au choix' (with a color palette icon), and 'Taille du texte' (82%).

Création d'une nouvelle table :

A screenshot of the phpMyAdmin interface. The left sidebar shows 'localhost' and 'dialogue' (0) as the current database. The main area shows a message: 'Aucune table n'a été trouvée dans cette base.' Below this, there's a form titled 'Créer une nouvelle table sur la base dialogue'. It has two input fields: 'Nom:' containing 'commentaire' and 'Nombre de colonnes:' containing '4', both of which are highlighted with red arrows.

Création de la structure de la table (champs/colonnes) :

localhost > dialogue > commentaire

Colonne	Type <small>②</small>	Taille/Valeurs <sup>*1</sup>	Défaut <sup>2</sup>	Interclassement	Attributs	Null	Index	A.
id_commentaire	INT	3	Aucun				PRIMARY	
pseudo	VARCHAR	20	Aucun					
message	TEXT		Aucun					
date_enregistrement	DATETIME		Aucun					

Commentaires sur la table:

Moteur de stockage: ③

InnoDB

Interclassement:

Définition de PARTITION: ④

**Structure de la table (relecture) :**

phpMyAdmin

localhost > dialogue > commentaire

Afficher Structure SQL Rechercher Insérer Exporter Importer Opérations Vider Supprimer

Base de données dialogue (1)

La table 'dialogue'.'commentaire' a été créée.

```
CREATE TABLE `dialogue`.`commentaire` (
  `id_commentaire` INT ( 3 ) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `pseudo` VARCHAR ( 20 ) NOT NULL ,
  `message` TEXT NOT NULL ,
  `date_enregistrement` DATETIME NOT NULL
) ENGINE = INNODB;
```

commentaire

Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<u><a href="#">id_commentaire</a></u>	int(3)			Non	Aucun	AUTO_INCREMENT	     
<u><a href="#">pseudo</a></u>	varchar(20)	latin1_swedish_ci		Non	Aucun		     
<u><a href="#">message</a></u>	text	latin1_swedish_ci		Non	Aucun		     
<u><a href="#">date_enregistrement</a></u>	datetime			Non	Aucun		     

Tout cocher / Tout décocher Pour la sélection :      

Pour créer cette base de données, vous pouvez également passer par la console Mysql :

A screenshot of the WAMPSEVER 2.1 application window. On the left, there's a sidebar with icons for Version, Service, Console MySQL (which is selected and highlighted in blue), my.ini, and MySQL log. The main area has a large blue header "WAMPSEVER 2.1". To the right of the header is a vertical menu with the following items: Localhost, phpMyAdmin, Répertoire www, Apache, PHP, MySQL (which is highlighted with a red arrow), Client XDebug, webGrind, Quick Admin, Démarrer les services, Arrêter les services, Redémarrer les services, and Passer en ligne. At the bottom of the screen, there's a taskbar with icons for various applications.

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe  
Enter password:  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 1  
Server version: 5.5.8-log MySQL Community Server <GPL>  
  
Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> CREATE DATABASE dialogue ;_
```

Voici le code à insérer :

Base de données dialogue - Table commentaire

```
CREATE DATABASE dialogue ;
USE dialogue ;

CREATE TABLE commentaire (
id_commentaire INT( 3 ) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
pseudo VARCHAR( 20 ) NOT NULL ,
message TEXT NOT NULL ,
date_enregistrement DATETIME NOT NULL
) ENGINE = InnoDB;
```

## 122 Etape 2. Crédation d'un dossier et d'un fichier dialogue.php avec 1 ligne de code permettant la connexion à la base de données.

Une fois que la base de données a été créée, vous pouvez commencer à développer votre script.

 /dialogue/
 dialogue.php

```
dialogue.php
```

```
<?php
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
```

Cette ligne de code nous permet de connecter notre page web (script) à la base de données.

Pour rappel, Mysqli est une classe permettant de se connecter à 1 base de données. Pour cela nous indiquons le nom du serveur, le pseudo, le mot de passe, le nom de la base de données.

## 123 Etape 3. Crédation d'un formulaire HTML (pour l'ajout de commentaire)

Nous allons créer un formulaire (au format HTML) afin que les internautes puissent s'exprimer (c'est bien le but de l'exemple !).

```
dialogue.php
```

```
<?php
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
?>

<hr>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
<label for="pseudo">Pseudo</label><br>
<input type="text" id="pseudo" name="pseudo"><br>

<label for="message">Message</label><br>
<textarea id="message" name="message" cols="50" rows="7"></textarea><br>

<input type="submit" value="Envoyer le message">
</form>
```

Nous prévoyons 2 champs dans notre formulaire : **pseudo & message**.

Dans la table commentaire de notre base de données dialogue, nous avions 4 champs : **id\_commentaire , pseudo , message , date\_enregistrement**.

Mais, nous n'allons pas demander à l'internaute de rentrer un **id\_commentaire**, puisque notre système est prévu pour le faire automatiquement (clé primaire + auto increment) et ça ne se fait pas !

De la même manière, ce n'est pas à l'internaute de rentrer la date et l'heure de son propre message, nous tacherons d'utiliser une fonction prédéfinie permettant de le faire de manière automatisée.

## Etape 4. Récupération et affichage des saisies en PHP (POST) sur la même page.

Maintenant que nous avons notre formulaire permettant la saisie de commentaires, il nous faut prévoir du code de récupération afin de "capter" les saisies de l'internaute.

```
dialogue.php
```

```
<?php
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
if($_POST)
{
    echo "pseudo posté: $_POST[pseudo] <br>";
    echo "message posté: $_POST[message] <br>";
}
?>

<hr>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo"><br>

    <label for="message">Message</label><br>
    <textarea id="message" name="message" cols="50" rows="7"></textarea><br>

    <input type="submit" value="Envoyer le message">
</form>
```

Nous ajoutons une condition IF pour dire que si l'internaute nous poste quelque chose (cette action est liée au clic sur le bouton submit), et bien nous affichons les saisies de l'internaute pour être certain de les "capter", avant de les enregistrer dans notre base de données.

L'affichage des saisies de l'internaute se fait grâce à la superglobale `$_POST`.

## Etape 5. Requête SQL d'enregistrement (INSERT)

Maintenant que nous avons un système qui nous permet de recevoir des données sur la page web, il serait utile d'enregistrer ces informations dans notre base afin de les garder en mémoire.

```
dialogue.php
```

```
<?php
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
if($_POST)
{
    //echo "pseudo posté: $_POST[pseudo] <br>";
    //echo "message posté: $_POST[message] <br>";
    $mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('$_POST[pseudo]', '$_POST[message]', NOW())");
    OR DIE ($mysqli->error);
    echo '<div class="validation">Votre message a bien été enregistré.</div>';
}
?>

<hr>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo"><br>

    <label for="message">Message</label><br>
    <textarea id="message" name="message" cols="50" rows="7"></textarea><br>

    <input type="submit" value="Envoyer le message">
</form>
```

Nous utilisons la méthode (fonction) `query()` de l'objet `$mysqli` afin de formuler une requête (de type INSERT, insertion) qui va nous permettre d'enregistrer les informations dans notre base de données.

Voici ce que ça donne au niveau de nos champs :

- id\_commentaire : nous ne rentrons pas d'informations pour ce champ, AUTO\_INCREMENT
- pseudo : nous envoyons le pseudo posté, \$\_POST, par l'internaute
- message : nous envoyons le message posté, \$\_POST, par l'internaute
- date\_enregistrement : nous utilisons la fonction SQL prédéfinie, NOW(), afin d'obtenir la date et l'heure du serveur au moment de l'enregistrement

Faites le test !

#### Résultat

Votre message a bien été enregistré.

#### Pseudo

Joker

#### Message

Mon premier message !

**Envoyer le message**

Pour être sûr que l'enregistrement a bien été pris en compte, vous pouvez retourner dans PhpMyAdmin ou dans la console SQL pour voir le contenu :

id_commentaire	pseudo	message	date_enregistrement
1	Joker	Mon premier message !	2015-07-30 10:30:11

← T →	id_commentaire	pseudo	message	date_enregistrement
□	✎	X		
	1	Joker	Mon premier message !	2015-07-30 10:30:11

#### 126 Etape 6. Améliorer l'étape 5 d'enregistrement avec la gestion des apostrophes et réaliser quelques contrôles de saisie.

Notre espace de dialogue est fonctionnel et il se présente bien puisqu'il enregistre des commentaires, cependant on peut nous envoyer tout et n'importe quoi, il serait donc utile de contrôler les saisies qui rentrent dans notre formulaire.

Exemple : L'internaute peut envoyer un pseudo et un message complètement vide. Cela s'enregistrerait aussi (faites le test !).

Pour y remédier nous pouvons ajouter des conditions (avec l'aide de la fonction prédéfinie empty, pour vérifier si cela est vide ou n'est pas vide).

Autre Problématique : Actuellement, nous ne pouvons pas déposer de message avec une apostrophe car cela ferait dysfonctionner la requête SQL.

Voici ce que cela donnait lors de la 1ère insertion :

```
$mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('joker', 'Mon premier message !', NOW())" OR DIE
($mysqli->error);
```

Imaginons maintenant que nous souhaitions déposer le message suivant : Aujourd'hui il fait beau.

Voici la requête SQL :

```
$mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('joker', 'Aujourd'hui il fait beau', NOW())" OR DIE
($mysqli->error);
```

Cela pose problème puisqu'il y aura une apostrophe de trop dans la requête ! Regardez bien. Le dysfonctionnement ne nous permettra pas d'insérer l'enregistrement !

Voici donc quelques améliorations dans les contrôles et traitements des données :

```
dialogue.php

<?php
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
if($_POST)
{
    $_POST['pseudo'] = addslashes($_POST['pseudo']);
    $_POST['message'] = addslashes($_POST['message']);
    if(empty($_POST['pseudo']) && empty($_POST['message']))
    {
        $mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('$_POST[pseudo]', '$_POST[message]', NOW())"
OR DIE ($mysqli->error);
        echo <div class="validation">Votre message a bien été enregistré.</div>';
    }
    else
    {
        echo '<div class="erreur">Afin de déposer un commentaire, veuillez svp remplir tous les champs du formulaire.</div>';
    }
}
?>

<hr>

<form method="post" action="">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo" maxlength="20" pattern="[a-zA-Z0-9-_]+" title="caractère autorisés : a-zA-Z0-9-_+"><br>

    <label for="message">Message</label><br>
    <textarea id="message" name="message" cols="50" rows="7"></textarea><br>

    <input type="submit" value="Envoyer le message">
</form>
```

Nous avons ajouté l'attribut pattern en html pour accepter seulement certains caractères pour le pseudo de l'internaute mais il s'agit du code exécuté côté client qui pourrait être retiré par l'internaute (puisque'il a accès au code source) et aussi non compatible avec d'anciens navigateurs. Il est donc vivement recommandé de renforcer ces contrôles dans la partie exécutée côté serveur, c'est à dire dans le code PHP.

La condition appuyée par la fonction prédéfinie `!empty()` permet de vérifier, avant d'enregistrer, si les saisies ne sont pas vides.

La fonction prédéfinie `addslashes()` permet d'ajouter automatiquement des anti-slash "\\" avant chaque apostrophe.

Nous pouvons maintenant déposer des messages avec des apostrophes, voici ce que cela donnerait avec le message suivant : Aujourd'hui il fait beau

```
$mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('joker', 'Aujourd'hui il fait beau', NOW())" OR DIE
($mysqli->error);
```

Vous pouvez remarquer la présence de l'anti-slash "\\" générée automatiquement par la fonction prédéfinie `addslashes()`, cela permet de ne pas faire dysfonctionner la requête SQL.

Plusieurs points de sécurité seraient également à améliorer. Il reste faillible à diverses attaques. Notre exemple est donc fonctionnel mais non sécurisé.

Notre exemple n'est pas totalement terminé, il nous faut absolument prévoir une partie affichage des commentaires, sinon les internautes auront du mal à se répondre les uns les autres

```
dialogue.php

<?php
// Partie connexion à la BDD
$mysqli = new mysqli('localhost', 'root', '', 'dialogue');
//-----
// Partie enregistrement
if($_POST)
```

```

{
    $_POST['pseudo'] = addslashes($_POST['pseudo']);
    $_POST['message'] = addslashes($_POST['message']);
    if(empty($_POST['pseudo']) && empty($_POST['message']))
    {
        $mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('$_POST[pseudo]', '$_POST[message]', NOW())");
        OR DIE ($mysqli->error);
        echo '<div class="validation">Votre message a bien été enregistré.</div>';
    }
    else
    {
        echo '<div class="erreur">Afin de déposer un commentaire, veuillez svp remplir tous les champs du formulaire.</div>';
    }
}
// Partie affichage des commentaires
$résultat = $mysqli->query("SELECT * FROM commentaire");
while($commentaire = $résultat->fetch_assoc())
{
    echo '<div class="message">';
    echo '<div class="titre">Par: ' . $commentaire['pseudo'] . ' ' . $commentaire['date_enregistrement'] . '</div>';
    echo '<div class="contenu">' . $commentaire['message'] . '</div>';
    echo '</div>';
}
// Partie formulaire d'envoi de commentaire
?>

<hr>
<form method="post" action="">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo" maxlength="20" pattern="[a-zA-Z0-9.-]+" title="caractère autorisés : a-zA-Z0-9.-"><br>

    <label for="message">Message</label><br>
    <textarea id="message" name="message" cols="50" rows="7"></textarea><br>

    <input type="submit" value="Envoyer le message">
</form>

```

#### Explications pour obtenir de l'affichage :

Nous utilisons la méthode (fonction) query() de l'objet \$mysqli afin de formuler une requête (de type SELECT, selection) qui va nous permettre de récupérer et d'afficher sur la page web les informations contenues dans notre base de données.

Nous récupérons les enregistrements dans la variable \$résultat.

\$résultat représente les enregistrements, techniquement c'est un objet Mysqli\_result (c'est ce que la méthode query() renvoie lors d'une requête de selection).

La méthode fetch\_assoc() utilisé sur l'objet Mysqli\_result (par l'intermédiaire de la variable \$résultat) permet de traiter les enregistrements (cela génère un tableau ARRAY) afin de pouvoir les exploiter. Combiné à la boucle while, cela permet aussi d'avancer de ligne en ligne dans les enregistrements de la table.

La boucle while est présente pour répéter le traitement autant de fois que nécessaire (s'il y a 10 enregistrements à afficher, nous ferons 10 tours de boucle afin d'exécuter le code permettant de faire un affichage 10 fois).

\$commentaire est, à chaque tour de boucle, un nouveau tableau ARRAY contenant les enregistrements.

Nous "piochons" à l'intérieur du tableau array \$commentaire avec l'utilisation des crochets [].

#### Résultat

Par: Joker, 2015-07-30 10:30:11

Mon premier message !

Par: Tomato, 2015-07-30 11:58:48

Et voici le second message !

Pseudo

Message

**Envoyer le message**

128

## Etape 8. Améliorer l'affichage : Ordonner et mettre les derniers commentaires en tête de liste, afficher la date au format Français, Afficher le nombre total de commentaires.

Voici le code de la partie affichage avec quelques améliorations (ordre, date, et nombre de commentaires)

dialogue.php

```
<?php
// Partie connexion à la BDD
$mysqli = new mysqli('localhost', 'root', "", 'dialogue');
//-----
// Partie enregistrement
if($_POST)
{
    $_POST['pseudo'] = addslashes($_POST['pseudo']);
    $_POST['message'] = addslashes($_POST['message']);
    if(empty($_POST['pseudo']) && empty($_POST['message']))
    {
        $mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('$_POST[pseudo]', '$_POST[message]', NOW())");
        OR DIE ($mysqli->error);
        echo '<div class="validation">Votre message a bien été enregistré.</div>';
    }
    else
    {
        echo '<div class="erreur">Afin de déposer un commentaire, veuillez svp remplir tous les champs du formulaire.</div>';
    }
}
// Partie affichage des commentaires
résultat = $mysqli->query("SELECT pseudo, message, DATE_FORMAT(date_enregistrement, '%d/%m/%Y') AS datefr,
DATE_FORMAT(date_enregistrement, '%H:%i:%s') AS heurefr FROM commentaire ORDER BY date_enregistrement DESC");
echo '<h2>' . $résultat->num_rows . ' commentaire(s)</h2>';
while($commentaire = $résultat->fetch_assoc())
{
    echo '<div class="message">';
    echo '<div class="titre">Par: ' . $commentaire['pseudo'] . ' le ' . $commentaire['datefr'] . ' à ' . $commentaire['heurefr'] . '</div>';
    echo '<div class="contenu">' . $commentaire['message'] . '</div>';
    echo '</div>';
}
// Partie formulaire d'envoi de commentaire
?>
<hr>
<form method="post" action="">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo" maxlength="20" pattern="[a-zA-Z0-9-_+]" title="caractère autorisés : a-zA-Z0-9-_+"><br>
    <label for="message">Message</label><br>
    <textarea id="message" name="message" cols="50" rows="7"></textarea><br>
    <input type="submit" value="Envoyer le message">
```

La requête SQL a été légèrement modifiée :

- Nous utilisons `DATE_FORMAT` afin de pouvoir transformer l'affichage de la date au format Français.
- Nous utilisons `ORDER BY date_enregistrement DESC` afin de pouvoir ré-ordonner l'affichage des enregistrements.

`$résultat->num_rows` permet d'afficher le nombre de commentaires déposés sur notre page web.

129

## Etape 9. Effectuer un retour visuel de meilleure qualité (css)

Nous pouvons ajouter les entêtes habituelles (`doctype`, `html`, `head`, `body`...) afin d'assurer la liaison avec une feuille de style CSS.

Voici une autre version du code possible :

```
dialogue.php

<?php
// Partie connexion à la BDD et initialisation
$mysqli = new mysqli('localhost', 'root', "", 'dialogue');
$contenu = "";
//-----
// Partie enregistrement
if($_POST)
{
    $_POST['pseudo'] = addslashes($_POST['pseudo']);
    $_POST['message'] = addslashes($_POST['message']);
    if(empty($_POST['pseudo']) && empty($_POST['message']))
    {
        $mysqli->query("INSERT INTO commentaire (pseudo, message, date_enregistrement) VALUES ('$_POST[pseudo]', '$_POST[message]', NOW())");
        OR DIE ($mysqli->error);
        $contenu .= '<div class="validation">Votre message a bien été enregistré.</div>';
    }
    else
    {
        $contenu .= '<div class="erreur">Afin de déposer un commentaire, veuillez svp remplir tous les champs du formulaire.</div>';
    }
}
//-----
// Partie affichage des commentaires
$résultat = $mysqli->query("SELECT pseudo, message, DATE_FORMAT(date_enregistrement, '%d/%m/%Y') AS datefr,
DATE_FORMAT(date_enregistrement, '%H:%i:%s') AS heurefr FROM commentaire ORDER BY date_enregistrement DESC");
$contenu .= '<h2>' . $résultat->num_rows . ' commentaire(s)';
while($commentaire = $résultat->fetch_assoc())
{
    $contenu .= '<div class="message">';
    $contenu .= '<div class="titre">Par: ' . $commentaire['pseudo'] . ', le ' . $commentaire['datefr'] . ' à ' . $commentaire['heurefr'] . '</div>';
    $contenu .= '<div class="contenu">' . $commentaire['message'] . '</div>';
    $contenu .= '</div>';
}
//-----
// Partie formulaire d'envoi de commentaire
?>

<!Doctype html>
<html>
    <head>
        <link rel="stylesheet" href="style.css">
    </head>
    <body>
        <div class="commentaire"><?php echo $contenu; ?></div>
        <form method="post" action="">
            <label for="pseudo">Pseudo</label><br>
            <input type="text" id="pseudo" name="pseudo" maxlength="20" pattern="[a-zA-Z0-9.-_]+" title="caractère autorisés : a-zA-Z0-9.-_"><br>
            <label for="message">Message</label><br>
            <textarea id="message" name="message" cols="50" rows="7"></textarea><br>
            <input type="submit" value="Envoyer le message">
        </form>
    </body>
</html>
```

Nous avons prévu une variable `$contenu` qui est déclarée vide mais que l'on va remplir à chaque fois que nous devrons réaliser un affichage.

Cela aura pour but de retenir l'affichage (puisque il s'agira d'une affectation de variable) afin qu'il n'y est pas d'affichage avant les balises principales (`doctype`, `html`, `head`, `body`).

L'avantage c'est que notre code sera bien valide w3c et que nous pourrons choisir l'emplacement précis de l'affichage.

A vous de créer votre fichier `style.css` et de mettre un peu de code à l'intérieur avec les classes existantes sur notre page web :

```
style.css
```

```
.erreur{ background: red; padding: 5px; }
.validation{ background: green; padding: 5px; }
.message{
background: #d9e8fd; : 500px; padding: 15px; margin: 10px auto; border: 2px solid #6eaafb; }
.titre{ text-align: center; : 500px; }
.contenu{ : 500px; font-style: italic; }
.commentaire{}
```

130

## Etape 10. Mise en ligne et Améliorations possibles.

### Mise en ligne :

A vous de faire la mise en ligne ! c'est un bon exercice pour voir si tout fonctionne bien.

Pour cela, il vous faut un hébergement type ovh, 1&1, planethoster, etc.

### Améliorations possible :

Enormément d'améliorations sont possibles !

- Nous pourrions par exemple ajouter une partie ajax (mélange php+js) afin de mettre à jour automatiquement les messages (sans recharge de page) lorsque plusieurs internautes communiquent.
- Nous pourrions aussi sécuriser notre page web.
- Nous pourrions améliorer le graphisme avec l'aide du code html/css

## Approche de la sécurité

131

## Espace membre

La Sécurité en PHP et la sécurité des sites web en général est un sujet très vaste, cela pourrait faire l'objet d'une formation entière.

Pour le moment, nous allons donc faire uniquement une sensibilisation par l'approche d'une des failles les plus connues : L'injection SQL.

Nous allons créer une nouvelle base de données que nous nommerons  **securite** (pour l'occasion).

Modélisation et création de la base de données, table et champs.

Base de données : **securite** 

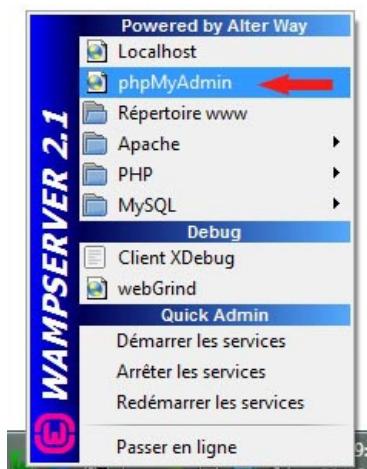
Table : **membre** 

Champ	Type	Taille	Spécificité
id_membre	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)
pseudo	VARCHAR	20	UNIQUE
mdp	VARCHAR	20	-
nom	VARCHAR	20	-
prenom	VARCHAR	20	-

email	VARCHAR	20	
-------	---------	----	--

Pour créer cette base de données, vous pouvez également passer par le gestionnaire de base de données **PhpMyAdmin** :

Accès à PhpMyAdmin :



Création d'une nouvelle base de données :

The screenshot shows the phpMyAdmin interface for MySQL localhost. The top navigation bar has tabs for 'Bases de données' (selected), 'SQL', 'Etat', 'Variables', and 'Jeux de caractères'. Below the tabs, there's an 'Actions' section. In the 'MySQL localhost' section, there's a 'Créer une base de données' button with a red arrow pointing to it. The input field next to it contains 'securite'. There's also a dropdown for 'Interclassement' set to 'Interclassement' and a 'Créer' button. The 'Interface' section includes settings for 'Langue - Language' (French), 'Thème / Style' (Original), 'Couleur au choix' (with a rainbow icon and 'Réinitialiser' button), and 'Taille du texte' (82%).

Création d'une nouvelle table :

The screenshot shows the phpMyAdmin interface for the 'securite' database. The top navigation bar has tabs for 'Structure' (selected), 'SQL', 'Rechercher', 'Requête', 'Exporter', 'Importer', 'Opérations', 'Privilèges', and 'Supprimer'. Below the tabs, it says 'Aucune table n'a été trouvée dans cette base.' In the 'Créer une nouvelle table sur la base securite' section, there's a form with 'Nom:' set to 'membre' and 'Nombre de colonnes:' set to '6'.

Création de la structure de la table (champs/colonnes) :

Colonne	Type	Taille/Valeurs <sup>1</sup>	Défaut <sup>2</sup>	Interclassement	Attributs	Null	Index	A.I.
id_membre	INT	3	Aucun				PRIMARY	
pseudo	VARCHAR	20	Aucun				UNIQUE	
mdp	VARCHAR	20	Aucun					
nom	VARCHAR	20	Aucun					
prenom	VARCHAR	20	Aucun					
email	VARCHAR	50	Aucun					

Structure de la table (relecture) :

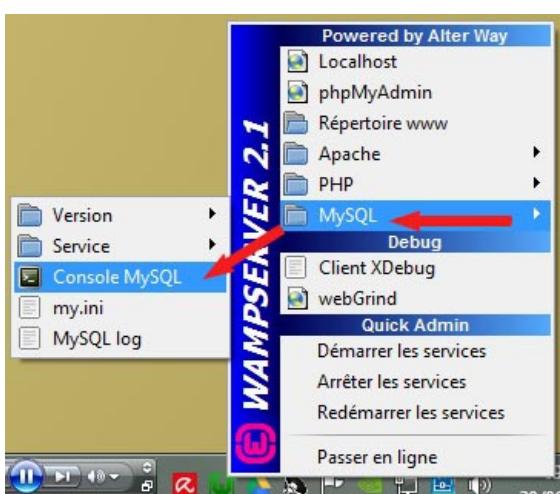
Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
id_membre	int(3)			Non	Aucun	AUTO_INCREMENT	
pseudo	varchar(20)	latin1_swedish_ci		Non	Aucun		
mdp	varchar(20)	latin1_swedish_ci		Non	Aucun		
nom	varchar(20)	latin1_swedish_ci		Non	Aucun		
prenom	varchar(20)	latin1_swedish_ci		Non	Aucun		
email	varchar(50)	latin1_swedish_ci		Non	Aucun		

Insertion d'enregistrement (remplissage de la table) :

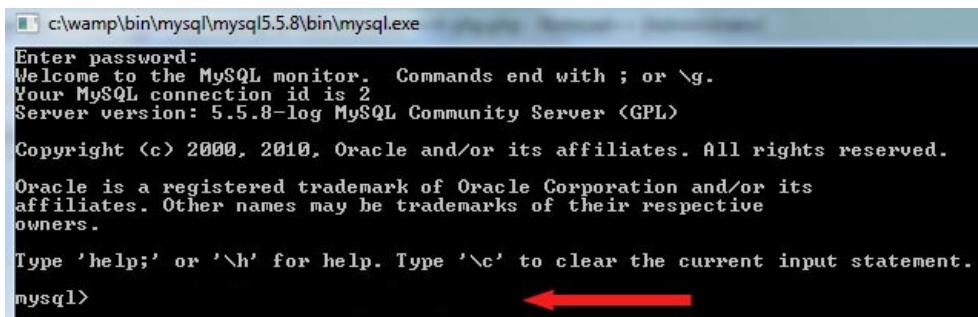
id_membre	pseudo	mdp	nom	prenom	email
1	Juju	soleil	Cottet	Julien	contact@monsite.com
2	LaMarie	planete	Thoyer	Marie	marie.thoyer@gmail.com
3	Laurence75	mars1980	Winter	Laurence	laurence75@hotmail.fr

Pour créer cette base de données, vous pouvez également passer par [la console Mysql](#) :

Accès à la console Mysql :



Requête dans la console Mysql :



```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

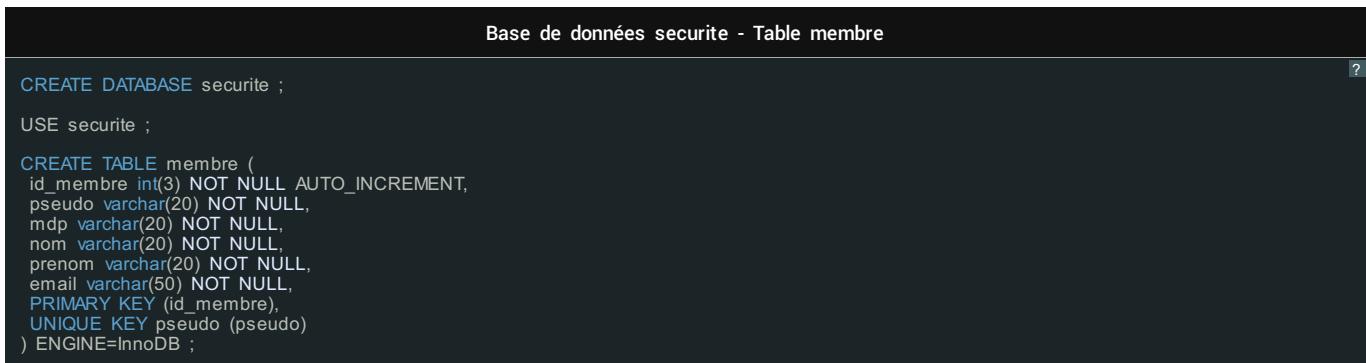
Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ↙
```

Voici le code à insérer :

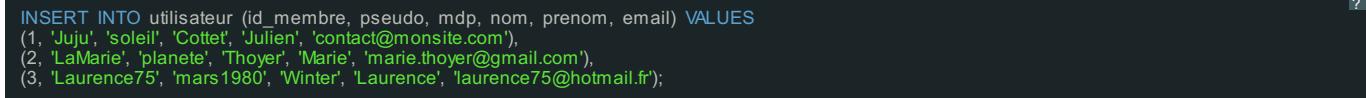


Base de données securite - Table membre

```
CREATE DATABASE securite ;
USE securite ;

CREATE TABLE membre (
    id_membre int(3) NOT NULL AUTO_INCREMENT,
    pseudo varchar(20) NOT NULL,
    mdp varchar(20) NOT NULL,
    nom varchar(20) NOT NULL,
    prenom varchar(20) NOT NULL,
    email varchar(50) NOT NULL,
    PRIMARY KEY (id_membre),
    UNIQUE KEY pseudo (pseudo)
) ENGINE=InnoDB ;
```

Nous allons insérer plusieurs enregistrement d'emblé :



```
INSERT INTO utilisateur (id_membre, pseudo, mdp, nom, prenom, email) VALUES
(1, 'Juju', 'soleil', 'Cottef', 'Julien', 'contact@monsite.com'),
(2, 'LaMarie', 'planete', 'Thoyer', 'Marie', 'marie.thoyer@gmail.com'),
(3, 'Laurence75', 'mars1980', 'Winter', 'Laurence', 'laurence75@hotmail.fr');
```

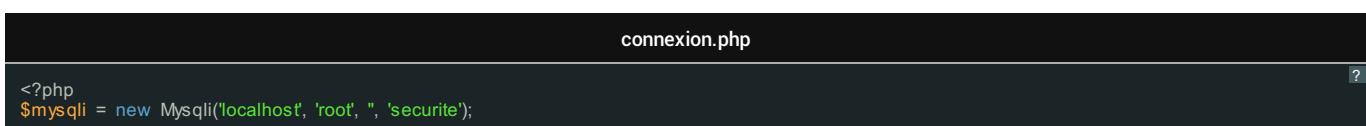
*Le pseudo est déclaré comme étant unique afin que 2 internautes ne puissent pas prendre le même pseudo.*

Création d'un dossier et d'un fichier connexion.php avec 1 ligne de code permettant la connexion à la base de données.

Une fois que la base de données a été créée, vous pouvez commencer à développer votre script.

/securite/  
  connexion.php

Dans cet exemple, nous ne ferons pas de page d'inscription afin de gagner du temps. Nous nous concentrerons directement sur la page de connexion :



connexion.php

```
<?php
$mysqli = new mysqli("localhost", "root", "", "securite");
```

Cette ligne de code nous permet de connecter notre page web (script) à la base de données.

Pour rappel, Mysqli est une classe permettant de se connecter à 1 base de données. Pour cela nous indiquons le nom du serveur, le pseudo, le mot de passe, le nom de la base de données.

Création d'un formulaire HTML (permettant la connexion des membres déjà inscrits)

Nous allons créer un formulaire (au format HTML) afin que les internautes puissent se connecter.



connexion.php

```
<?php
$mysqli = new mysqli("localhost", "root", "", "securite");
?>

<hr>

<form method="post" action=">
  <label for="pseudo">Pseudo</label><br>
```

```

<input type="text" id="pseudo" name="pseudo"><br>
<label for="mdp">mdp</label><br>
<input type="text" id="mdp" name="mdp"><br>
<input type="submit" value="Se connecter">
</form>

```

Nous prévoyons 2 champs dans notre formulaire : **pseudo** & **mdp**.

**Récupération et affichage des saisies en PHP (POST) sur la même page.**

Maintenant que nous avons notre formulaire permettant la saisie d'identifiants de connexion, il nous faut prévoir du code de récupération afin de "capter" les saisies de l'internaute.

connexion.php

```

<?php
$mysqli = new mysqli('localhost', 'root', '', 'securite');
if($_POST)
{
    echo "pseudo posté: $_POST[pseudo] <br>";
    echo "mdp posté: $_POST[mdp] <br>";
}
?>

<hr>

<form method="post" action="">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo"><br>

    <label for="mdp">mdp</label><br>
    <input type="text" id="mdp" name="mdp"><br>

    <input type="submit" value="Se connecter">
</form>

```

Nous ajoutons une condition IF pour dire que si l'internaute nous poste quelque chose (cette action est liée au clic sur le bouton submit), et bien nous affichons les saisies de l'internaute pour être certain de les "capter", avant de tenter de connecter le membre.

L'affichage des saisies de l'internaute se fait grâce à la superglobale **\$\_POST**.

#### Requête SQL de sélection (SELECT)

Maintenant que nous avons un système qui nous permet de recevoir des données sur la page web, il serait utile de faire une sélection dans notre base afin de voir si ces informations sont reconnues (le membre existe-t'il ?).

connexion.php

```

<?php
$mysqli = new mysqli('localhost', 'root', '', 'securite');
if($_POST)
{
    //echo "pseudo posté: $_POST[pseudo] <br>";
    //echo "mdp posté: $_POST[mdp] <br>";
    $req = "SELECT * FROM membre WHERE pseudo='".$_POST[pseudo]' AND mdp='".$_POST[mdp]'";
    $résultat = $mysqli->query($req);
    echo 'requête debug : ' . $req . ':';
    $membre = $résultat->fetch_assoc();
    if(empty($membre))
    {
        echo '<div class="validation"><h1>Vous êtes bien reconnu par le site web pour vous connecter...</h1></div>';
        echo 'votre id est : ' . $membre[id_membre] . "<br>";
        echo 'votre pseudo est : ' . $membre[pseudo] . "<br>";
        echo 'votre mdp est : ' . $membre[mdp] . "<br>";
        echo 'votre nom est : ' . $membre[nom] . "<br>";
        echo 'votre prenom est : ' . $membre[mdp] . "<br>";
        echo 'votre email est : ' . $membre[email] . "<br>";
    }
    else
    {
        echo '<div class="erreur"><h1>Erreur d\'identification</h1></div>';
    }
}
?>

<hr>

<form method="post" action="">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo"><br>
    <label for="mdp">mdp</label><br>

```

```
<input type="text" id="mdp" name="mdp"><br>
<input type="submit" value="Se connecter">
</form>
```

#### Explications :

Une fois que l'internaute saisit un pseudo et un mot de passe, nous souhaitons vérifier dans notre base de données si les informations existent afin de pouvoir éventuellement le connecter et le faire accéder à son compte.

Pour cela, nous utilisons la méthode (fonction) `query()` de l'objet `$mysql` afin de formuler une requête (de type `SELECT, selection`) qui va nous permettre de récupérer et d'afficher sur la page web les informations contenues dans notre base de données.

Nous récupérons les enregistrements dans la variable `$résultat`.

`$résultat` représente les enregistrements, techniquement c'est un objet `Mysqli_result` (c'est ce que la méthode `query()` renvoie lors d'une requête de `selection`).

La méthode `fetch_assoc()` utilisée sur l'objet `Mysqli_result` (par l'intermédiaire de la variable `$résultat`) permet de traiter les enregistrements (cela génère un tableau `ARRAY`) afin de pouvoir les exploiter.

Il y a qu'un seul internaute à récupérer à la fois, nous ne prévoyons donc pas de boucle `while`.

Nous prévoyons une condition `IF` supplémentaire pour demander si `$membre` n'est pas vide, c'est que l'on a rapatrié des informations sur un membre avec un pseudo et mdp existants.

Nous "piochons" à l'intérieur du tableau array `$membre` avec l'utilisation des crochets `[]`.

#### Résultat

```
requete debug : SELECT * FROM membre WHERE pseudo='juju' AND mdp='soleil'
```

## Vous êtes bien reconnu par le site web pour vous connecter...

```
votre id est : 1
votre pseudo est : Juju
votre mdp est : soleil
votre nom est : Cottet
votre prenom est : soleil
votre email est : contact@monsite.com
```

Pseudo  
  
mdp

Nous allons vous présenter une attaque (hacking) d'injection SQL pour détourner le comportement initialement prévu du site web, et bien entendu nous verrons comment s'en protéger.

Pour attaquer, tenter d'inscrire les informations suivantes (uniquement dans la case `pseudo`) :

```
pseudo : juju' #
```

```
pseudo : lamarie' #
```

pseudo : laurence75' #

#### Résultat

requete debug : SELECT \* FROM membre WHERE pseudo='Lamarie' # AND mdp=" 

## Vous êtes bien reconnu par le site web pour vous connecter...

votre id est : 2  
votre pseudo est : LaMarie  
votre mdp est : planete  
votre nom est : Thoyer  
votre prenom est : planete  
votre email est : marie.thoyer@gmail.com

Pseudo  
   
mdp

Nous inscrivons le pseudo de l'internaute suivi d'une quotes (symbole apostrophe) pour fermer la valeur à l'intérieur de la requête, nous plaçons ensuite un signe dieze pour que le reste de la requête soit mis en commentaire.

De cette manière la bonne apostrophe (quotes) sera ignorée, ainsi que le mot de passe. Cela nous permettra d'accéder à tous les comptes uniquement avec 1 pseudo.

Avec ce détournement, voici la requête SQL formulée par le serveur vers notre SGBD :

```
S E L E C T      *      F R O M      m i e m b r e _ i d e i e @ ' H I D # R E
```

*La partie inscrite en rouge correspond à notre texte saisi*

Le dieze # permet de mettre la suite de la requête en commentaire, on ne se préoccupe plus du mdp donc (^\_\*) !!!

Si nous ne connaissons pas le pseudo du membre, nous pouvons toujours tenter avec son id\_membre :

Pour attaquer, tenter d'inscrire les informations suivantes (uniquement dans la case mdp) :

mdp : ' OR id\_membre = '1

mdp : ' OR id\_membre = '2

mdp : ' OR id\_membre = '3

#### Résultat

requete debug : SELECT \* FROM membre WHERE pseudo=" AND mdp=" OR id\_membre = '1'

## Vous êtes bien reconnu par le site web pour vous connecter...

votre id est : 1  
votre pseudo est : Juju  
votre mdp est : soleil  
votre nom est : Cottet  
votre prenom est : soleil  
votre email est : contact@monsite.com

Pseudo

mdp

Avec ce nouveau détournement, voici la requête SQL formulée par le serveur vers notre SGBD :

```
S E L E C T * F R O M m e m b r e ' W O H RE R E d _p n's ee m b r e = ' 1
```

La partie inscrite en rouge correspond à notre texte saisi

Nous fermons l'apostrophe du champ mot de passe de la requête et nous en servons plus loin pour demander un id\_membre en particulier.

Cela donne "selectionne moi le membre qui a un pseudo et un mot de passe vide OU celui qui a l'id\_membre 1".

Personne ne possède de compte avec 1 pseudo et 1 mdp vide, mais il y a bien 1 membre qui a le numéro de membre 1 (^\_~) !!!

Notons tout de même que nous avons orienté le code de manière à laisser la porte très ouverte à ces attaques afin de présenter les injections SQL. En règle général, les sites web se protègent contre ce type d'attaque.

En l'état, ces injections permettent de rentrer sur le compte d'un internaute ou avoir des informations sans être le propriétaire du compte.

Nous aurions pu aller plus loin et détruire une Base De Données, stopper le serveur, etc. mais nous allons nous arrêter là car il ne s'agit pas d'une introduction au piratage mais une sensibilisation aux failles de sécurité pour montrer de quelle manière cela peut exister.

133

### Moyen de contre

Si vous souhaitez vous protéger de ce type d'attaque par injection SQL, vous pouvez faire appel à la fonction prédéfinie [htmlspecialchars\(\)](#) ou encore [htmlentities\(\)](#).

connexion.php

```
<?php
$mysqli = new mysqli('localhost', 'root', '', 'securite');
if($_POST)
{
    $_POST['pseudo'] = htmlentities($_POST['pseudo'], ENT_QUOTES);
    $_POST['mdp'] = htmlentities($_POST['mdp'], ENT_QUOTES);
    $req = "SELECT * FROM membre WHERE pseudo=$_POST[pseudo] AND mdp=$_POST[mdp]";
    $resultat = $mysqli->query($req);
    echo 'requete debug : ' . $req . ';
    $membre = $resultat->fetch_assoc();
    if(empty($membre))
    {
        echo '

<h1>Vous êtes bien reconnu par le site web pour vous connecter...</h1></div>';
        echo "votre id est : " . $membre[id_membre] . "<br>";
        echo "votre pseudo est : " . $membre[pseudo] . "<br>";
        echo "votre mdp est : " . $membre[mdp] . "<br>";
        echo "votre nom est : " . $membre[nom] . "<br>";
        echo "votre prenom est : " . $membre[mdp] . "<br>";
    }
}


```

```

        echo "votre email est : " . $membre['email'] . "<br>";
    }
} else
{
    echo '<div class="erreur"><h1>Erreur d\'identification</h1></div>';
}
?>

<hr>

<form method="post" action=""">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo"><br>

    <label for="mdp">mdp</label><br>
    <input type="text" id="mdp" name="mdp"><br>

    <input type="submit" value="Se connecter">
</form>

```

Comme le dit la documentation officielle de PHP, la fonction préédéfinie `htmlentities()` convertit tous les caractères éligibles en entités HTML.

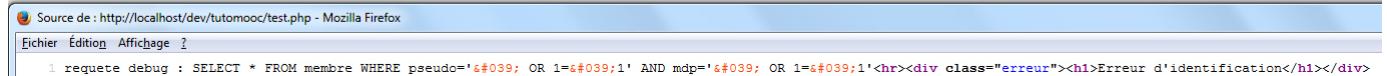
Pour le comprendre, il faut se rendre au niveau du code-source juste après l'attaque.

## Résultat

requete debug : `SELECT * FROM membre WHERE pseudo=' OR 1=1' AND mdp=' OR 1=1'`

## Erreur d'identification

Pseudo  
  
 mdp



Nous pouvons voir notamment que les chevrons ont été convertis.

Bien entendu, cela n'est qu'une introduction mais elle n'est pas suffisante pour protéger 1 site web complet, beaucoup d'autres types d'attaques existent.

## Réalisation d'un site web complet

### Objectifs

- ✓ Créer un site web « 2en1 » avec une partie cliente (front) et une interface de gestion (back).
- ✓ Développement de fonctionnalités. (CRUD orienté e-commerce)

### Réalisation d'un site web (CRUD), tendance ecommerce

Pour le prochain exemple, nous allons réaliser la base d'un site web procédural avec une tendance ecommerce.

Ce site aura le mérite de nous faire pratiquer le CRUD, Create Read Update Delete.

Le CRUD représente la base en PHP. Il s'agit de savoir librement consulter, insérer, modifier ou supprimer des données.

Sans ça, ne pensez pas mettre le mot "PHP" sur votre CV. (*ou alors il faudra éventuellement écrire « notions php »*).



Base de données : site

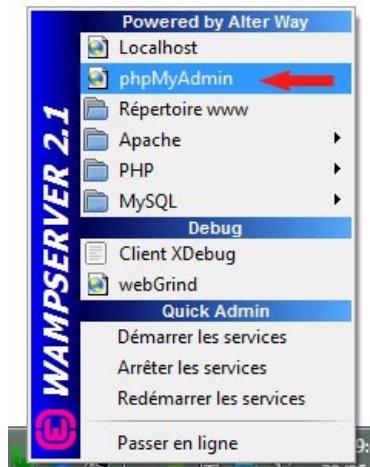


Table : membre

Champ	Type	Taille	Spécificité	Description
id_membre	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)	Ce champ correspond au numéro du membre qui sera auto-généré et incrémenté
pseudo	VARCHAR	20	UNIQUE	Ce champ correspond au pseudo du membre. Il sera unique et par conséquent 2 membres ne pourront pas avoir le même pseudo.
mdp	VARCHAR	32	-	Ce champ correspond au mot de passe du membre. La taille fait 32 caractères car il sera crypté par la suite.
nom	VARCHAR	20	-	Ce champ correspond au nom de famille du membre.
prenom	VARCHAR	20	-	Ce champ correspond au prénom du membre.
email	VARCHAR	50	-	Ce champ correspond à l'email du membre.
civilite	ENUM	'm';'f'	-	Ce champ correspond à la civilité du membre. Le membre sera soit Homme (M) soit Femme (F). Logiquement il n'y a que 2 choix possibles :p
ville	VARCHAR	20	-	Ce champ correspond à la ville du membre.
code_postal	INT	5	UNSIGNED ZEROFILL	Ce champ correspond au code postal du membre.
adresse	VARCHAR	50	-	Ce champ correspond à l'adresse du membre.
statut	INT	1	DEFAULT 0	Ce champ correspond au statut du membre. Par défaut il sera à zéro (ce qui correspondra à 1 membre). Nous pourrons mettre le chiffre 1 pour donner des droits d'administration à certains membres (1 admin est aussi 1 membre).

Pour créer cette base de données, vous pouvez également passer par [le gestionnaire de base de données PhpMyAdmin](#) :

Accès à PhpMyAdmin :



Création d'une nouvelle base de données :

The screenshot shows the phpMyAdmin interface for MySQL localhost. A red arrow points to the search bar where 'site' is typed. Another red arrow points to the 'Créer' (Create) button. The interface includes language settings (Français - French), theme (Original), color palette, and font size controls.

Création d'une nouvelle table :

The screenshot shows the phpMyAdmin interface for the 'site' database. It displays a success message: 'La base de données site a été créée.' Below it, the SQL command 'CREATE DATABASE `site` ;' is shown. The 'Structure' tab is selected. A red arrow points to the 'Nom:' field containing 'membre'. Another red arrow points to the 'Nombre de colonnes:' field containing '11'.

Création de la structure de la table (champs/colonnes) :

The screenshot shows the detailed structure of the 'membre' table in the 'site' database. The table has 11 columns: id\_membre, pseudo, mdp, nom, prenom, email, civilite, ville, code\_postal, adresse, and statut. Red arrows highlight various fields across the row: the first two columns (id\_membre and pseudo), the 'Type' column for pseudo and mdp, the 'Taille/Valeurs' column for pseudo, mdp, nom, prenom, email, and ville, the 'Défaut' column for pseudo, mdp, nom, prenom, email, and ville, the 'Interclassement' column for pseudo, mdp, nom, prenom, email, and ville, the 'Attributs' column for pseudo, mdp, nom, prenom, email, and ville, the 'Null' column for pseudo, mdp, nom, prenom, email, and ville, the 'Index' column for pseudo, mdp, nom, prenom, email, and ville, the 'A.I.' column for id\_membre, and the 'Commentaires' column for id\_membre.

Structure de la table (relecture) :

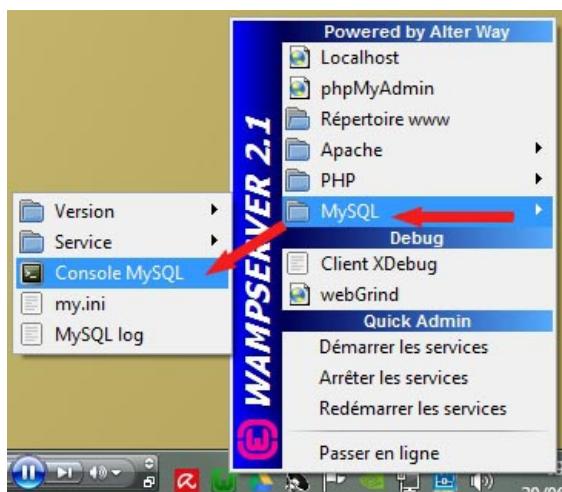
**phpMyAdmin**

localhost > site > membre

**Afficher** **Structure** **SQL** **Rechercher** **Insérer** **Exporter** **Importer** **Opérations** **Vider** **Supprimer**

Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<code>id_membre</code>	int(3)			Non	Aucun	AUTO_INCREMENT	
<code>pseudo</code>	varchar(20)	latin1_swedish_ci		Non	Aucun		
<code>mdp</code>	varchar(32)	latin1_swedish_ci		Non	Aucun		
<code>nom</code>	varchar(20)	latin1_swedish_ci		Non	Aucun		
<code>prenom</code>	varchar(20)	latin1_swedish_ci		Non	Aucun		
<code>email</code>	varchar(50)	latin1_swedish_ci		Non	Aucun		
<code>civilite</code>	enum('m','f')	latin1_swedish_ci		Non	Aucun		
<code>ville</code>	varchar(20)	latin1_swedish_ci		Non	Aucun		
<code>code_postal</code>	int(5)			UNSIGNED ZEROFILL	Non	Aucun	
<code>adresse</code>	varchar(50)	latin1_swedish_ci			Non	Aucun	
<code>statut</code>	int(1)				Non	0	

Pour créer cette base de données, vous pouvez également passer par la [console Mysql](#) :



```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

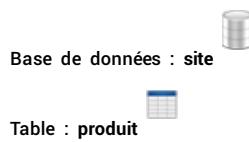
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
```

Voici le code à insérer :

```
CREATE DATABASE site ;
USE site ;

CREATE TABLE membre (
    id_membre INT(3) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    pseudo VARCHAR(20) NOT NULL ,
    mdp VARCHAR(32) NOT NULL ,
    nom VARCHAR(20) NOT NULL ,
    prenom VARCHAR(20) NOT NULL ,
    email VARCHAR(50) NOT NULL ,
    civilite ENUM('m', 'f') NOT NULL ,
    ville VARCHAR(20) NOT NULL ,
    code_postal INT(5) UNSIGNED ZEROFILL NOT NULL ,
    adresse VARCHAR(50) NOT NULL ,
    statut INT(1) NOT NULL DEFAULT 0,
    UNIQUE (pseudo)
) ENGINE = InnoDB;
```

## La table produit



Champ	Type	Taille	Spécificité	Description
id_produit	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)	Ce champ correspond au numéro du produit qui sera auto-généré et incrémenté
reference	VARCHAR	20	UNIQUE	Ce champ correspond à la référence du produit. Il sera unique et par conséquent 2 produits ne pourront pas avoir la même référence.
categorie	VARCHAR	20	-	Ce champ correspond à la catégorie du produit
titre	VARCHAR	100	-	Ce champ correspond au titre du produit.
description	TEXT	-	-	Ce champ correspond à la description du produit.
couleur	VARCHAR	20	-	Ce champ correspond à la couleur du produit.
taille	VARCHAR	5	-	Ce champ correspond à la taille du produit.
public	ENUM	'm','f', 'mixte'	-	Ce champ permettra de déterminer à quel public est destiné ce produit. Les choix possibles sont Homme (M), soit Femme (F) ou mixte (mixte).
photo	VARCHAR	250	-	Ce champ correspond au chemin de la photo qui sera enregistré pour représenter le produit. Ce ne sera pas le fichier image directement mais bien son chemin qui sera enregistré.
prix	INT	3	-	Ce champ correspond au prix du produit.
stock	INT	3	-	Ce champ correspond au stock restant du produit.

Pour créer cette base de données, vous pouvez également passer par [le gestionnaire de base de données PhpMyAdmin](#) :

Création d'une nouvelle table :

The screenshot shows the phpMyAdmin interface. In the left sidebar, under the 'site' database, a table named 'membre' is listed. On the right, a form for creating a new table is open. The table name is set to 'produit'. Two red arrows point to the input fields for 'Nom:' and 'Nombre de colonnes:', both of which contain the value '11'.

Création de la structure de la table (champs/columnes) :

Colonne	Type	Taille/Valeurs <sup>*1</sup>	Défaut <sup>*2</sup>	Interclassement	Attributs	Null	Index	A.I	Commentaires
id_produit	INT	3	Aucun				PRIMARY	<input checked="" type="checkbox"/>	
reference	VARCHAR	20	Aucun				UNIQUE	<input type="checkbox"/>	
categorie	VARCHAR	20	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
titre	VARCHAR	100	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
description	TEXT		Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
couleur	VARCHAR	20	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
taille	VARCHAR	5	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
public	ENUM	'm','f','mixte'	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
photo	VARCHAR	250	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
prix	INT	3	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
stock	INT	3	Aucun			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Structure de la table (relecture) :

	Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action			
<input type="checkbox"/>	<b>id_produit</b>	int(3)			Non	Aucun	AUTO_INCREMENT				
<input type="checkbox"/>	<b>reference</b>	varchar(20)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>categorie</b>	varchar(20)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>titre</b>	varchar(100)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>description</b>	text	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>couleur</b>	varchar(20)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>taille</b>	varchar(5)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>public</b>	enum('m','f','mixte')	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>photo</b>	varchar(250)	latin1_swedish_ci		Non	Aucun					
<input type="checkbox"/>	<b>prix</b>	int(3)			Non	Aucun					
<input type="checkbox"/>	<b>stock</b>	int(3)			Non	Aucun					

Pour créer cette base de données, vous pouvez également passer par la [console Mysql](#) :

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> 
```

Voici le code à insérer :

Base de données site - Table produit	
<pre>CREATE TABLE produit (     id_produit INT(3) NOT NULL AUTO_INCREMENT PRIMARY KEY ,     reference VARCHAR(20) NOT NULL ,     categorie VARCHAR(20) NOT NULL ,     titre VARCHAR(100) NOT NULL ,     description TEXT NOT NULL ,     couleur VARCHAR(20) NOT NULL ,     taille VARCHAR(5) NOT NULL ,     public ENUM('m','f','mixte') NOT NULL ,     photo VARCHAR(250) NOT NULL ,     prix INT(3) NOT NULL ,     stock INT(3) NOT NULL ,</pre>	

UNIQUE (reference)  
ENGINE = InnoDB;

#### La table commande

Base de données : site



Table : commande



Champ	Type	Taille	Spécificité	Description
id_commande	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)	Ce champ correspond au numéro de commande qui sera auto-généré et incrémenté
id_membre	INT	3	Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL	Ce champ correspond à l'id_membre qui aura commandé.
montant	INT	3	-	Ce champ correspond au montant total (en euros) de la commande
date_enregistrement	DATETIME	-	-	Ce champ correspond à la date et heure d'enregistrement de la commande.
etat	Enum	'en cours de traitement','envoyé','livré'	DEFAULT 'en cours de traitement'	Ce champ correspond a l'état de commande.

Pour créer cette base de données, vous pouvez également passer par [le gestionnaire de base de données PhpMyAdmin](#) :

Création d'une nouvelle table :

**Créer une nouvelle table sur la base site**

Nom: commande	Nombre de colonnes: 5
---------------	-----------------------

Création de la structure de la table (champs/colonnes) :

Colonne	Type	Taille/Valeurs <sup>1</sup>	Défaut <sup>2</sup>	Interclassement	Attributs	Null	Index	A_I	Commentaires
id_commande	INT	3	Aucun				PRIMARY	<input checked="" type="checkbox"/>	
id_membre	INT	3	NULL			<input checked="" type="checkbox"/>			
montant	INT	3	Aucun						
date_enregistrement	DATETIME		Aucun						
etat	ENUM	'en cours de traitement','envoyé','livré'	'en cours de traitement'						

Structure de la table (relecture) :

	Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action				
<input type="checkbox"/>	<u>id_commande</u>	int(3)			Non	Aucun	AUTO_INCREMENT					
<input type="checkbox"/>	<u>id_membre</u>	int(3)			Oui	NULL						
<input type="checkbox"/>	<u>montant</u>	int(3)			Non	Aucun						
<input type="checkbox"/>	<u>date_enregistrement</u>	datetime			Non	Aucun						
<input type="checkbox"/>	<u>etat</u>	enum('en cours de traitement','envoyé','livré')	latin1_swedish_ci		Non	Aucun						

Pour créer cette base de données, vous pouvez également passer par [la console Mysql](#) :

```

c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ↙

```

Voici le code à insérer :

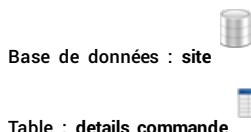
Base de données site - Table commande

```

CREATE TABLE commande (
    id_commande INT(3) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    id_membre INT(3) NULL DEFAULT NULL,
    montant INT(3) NOT NULL,
    date_enregistrement DATETIME NOT NULL,
    etat ENUM('en cours de traitement', 'envoyé', 'livré') NOT NULL
) ENGINE = InnoDB;

```

#### La table details\_commande



Champ	Type	Taille	Spécificité	Description
id_details_commande	INT	3	Clé primaire (PK - Primary Key), AUTO_INCREMENT (AI)	Ce champ correspond au numéro du détail de la commande qui sera auto-généré et incrémentée
id_commande	INT	3	Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL	Ce champ correspond à l'id_commande à laquelle le détail est rattaché.
id_produit	INT	3	Clé étrangère (FK - Foreign Key), NULL, DEFAULT NULL	Ce champ correspond à l'id_produit qui aura été commandé.
quantite	INT	3	-	Ce champ correspond à la quantité demandée par produit.
prix	INT	3	-	Ce champ correspond au prix du produit

Pour créer cette base de données, vous pouvez également passer par [le gestionnaire de base de données PhpMyAdmin](#) :

Création d'une nouvelle table :

Créer une nouvelle table sur la base site

Nom: details\_commande Nombre de colonnes: 5

Création de la structure de la table (champs/colonnes) :

Colonne	Type	Taille/Valeurs <sup>1</sup>	Défaut <sup>2</sup>	Interclassement	Attributs	Null	Index	A.I
id_details_commande	INT	3	Aucun				PRIMARY	
id_commande	INT	3	NULL					
id_produit	INT	3	NULL					
quantite	INT	3	Aucun					
prix	INT	3	Aucun					

Structure de la table (relecture) :

	Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/>	<u><a href="#">id_details_commande</a></u>	int(3)			Non	Aucun	AUTO_INCREMENT	
<input type="checkbox"/>	<u><a href="#">id_commande</a></u>	int(3)			Oui	NULL		
<input type="checkbox"/>	<u><a href="#">id_produit</a></u>	int(3)			Oui	NULL		
<input type="checkbox"/>	<u><a href="#">quantite</a></u>	int(3)			Non	Aucun		
<input type="checkbox"/>	<u><a href="#">prix</a></u>	int(3)			Non	Aucun		

Pour créer cette base de données, vous pouvez également passer par la [console Mysql](#) :

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> 
```

Voici le code à insérer :

### Base de données site - Table details\_commande

```
CREATE TABLE details_commande (
    id_details_commande INT(3) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    id_commande INT(3) NULL DEFAULT NULL,
    id_produit INT(3) NULL DEFAULT NULL,
    quantite INT(3) NOT NULL,
    prix INT(3) NOT NULL
) ENGINE = InnoDB;
```

Avec cette table nous pourrons avoir 1 commande (dans la table commande) comprenant par exemple 5 produits (dans la table details\_commande, pour cette même commande).

Comme il s'agit d'un entraînement, nous reviendrons sur les clés étrangères et les contraintes d'intégrité un peu plus tard afin d'en parler en détail.

Bien entendu, toutes les tables sont simplifiées, dans le cadre d'un vrai site ecommerce nous pourrions avoir des dizaines de tables.

135

## Etape 2. Création d'une arborescence

Nous allons créer notre arborescence de site web :

```
- /site/
---- /photo/
---- /admin/
----- /gestion_boutique.php
----- /gestion_commande.php
----- /gestion_membre.php
---- /inc/
----- /img/
----- /js/
----- /css/
----- /style.php
----- /init.inc.php
----- /fonction.inc.php
----- /haut.inc.php
```

```
----- ----- bas.inc.php
-  inscription.php
-  connexion.php
-  profil.php
-  boutique.php
-  fiche_produit.php
-  panier.php
-  .htaccess
```

Notre site web se trouvera à l'intérieur du dossier  /site/.

1 dossier  /photo/ sera présent pour contenir les photos de nos produits.

1 dossier  /admin/ sera présent pour contenir les pages d'administration (BackOffice).

1 dossier  /inc/ sera présent pour contenir les fichiers n'étant pas des pages web à part entière (bien souvent il s'agit de fichiers inclus dans des pages web).

Jusque là, comprenez-vous notre arborescence ? elle est plutôt simple, non ? Et bien entendu, il y a aussi les pages web côté FRONT :

- inscription.php : page d'inscription pour les visiteurs
- connexion.php : page de connexion pour les membres
- profil.php : page de profil pour les membres connectés
- boutique.php : catalogue des différents produits séparés par catégorie
- fiche\_produit.php : fiche d'un produit en particulier
- panier.php : panier
- .htaccess : fichier permettant de faire des réglages, notamment sur les URLs pour améliorer le référencement/li>

136

### Etape 3. Ecriture des fichiers en inclusion

#### Etape 3. Ecriture du fichier inc/init.inc.php

Le fichier init.inc.php (nommé .inc car destiné à l'inclusion et non pas à l'affichage), va nous permettre d'initialiser plusieurs choses sur notre site web.

Le fichier init.inc.php sera donc inclus par toutes nos pages web afin de profiter de l'initialisation.

Pourquoi initialiser ? qu'est-ce qu'on pourrait retrouver dedans ? la connexion à la base de données par exemple ! le réglage de l'encodage, et différentes variables...

Et oui, sans ça, pas de site dynamique, nous en aurons donc besoin sur toutes les pages.

```
inc/init.inc.php

<?php
//----- BDD
$mysqli = new mysqli("localhost", "root", "", "site");
if ($mysqli->connect_error) die('Un problème est survenu lors de la tentative de connexion à la BDD : ' . $mysqli->connect_error);
// $mysqli->set_charset("utf8");

//----- SESSION
session_start();

//----- CHEMIN
define("RACINE_SITE","/site/");

//----- VARIABLES
$content = "";

//----- AUTRES INCLUSIONS
require_once("fonction.inc.php");
```

#### Quelques explications

Mysqli est une classe prédéfinie en PHP me permettant de me connecter à la base de données.

Pour cela il est nécessaire de lui annoncer le nom du serveur, le pseudo, le mot de passe, et la base de données à laquelle nous souhaitons nous connecter.

1 condition est présente `$mysqli->connect_error` pour afficher un message d'erreur en Français si jamais la connexion ne peut pas se faire (l'erreur est souvent due à une mauvaise information dans la chaîne de connexion).

`$mysqli->set_charset("utf8");` permet de régler l'encodage de la base de données.

`session_start()` permet de créer (ou lire) 1 fichier de session sur le serveur. Sans cette ligne, nous ne pourrons pas connecter d'internautes à leurs espaces membres plus tard.

`Session_start()` permettra en effet de maintenir (et ne pas perdre) l'internaute connecté au site web même s'il navigue de page en page.

`define("RACINE_SITE","/site/");` permettra de gérer notre site web en chemin absolu et non pas relatif. Et pour éviter tout problème, nous pourrons modifier cette constante une fois en ligne pour que cela ait une répercussion immédiate partout où elle sera appelée..

`$contenu = "`; est une variable initialisée à vide pour éviter d'avoir des erreurs undefined si jamais nous tentons de l'afficher.

Nous nous servirons de cette variable pour retenir des messages que nous devrions adresser à l'internaute, cela nous permettra de faire 1 affichage global de tous nos éventuels messages à un endroit précis (et non pas au dessus du doctype par exemple).

`require_once("fonction.inc.php");` nous allons inclure notre fichier de fonction avec nous. Du coup, lorsque nous appellerons le fichier init.inc.php, cela aura aussi pour effet d'inclure le fichier fonction.inc.php, (2 en 1) !

### Etape 3. Ecriture du fichier inc/haut.inc.php

Nous allons écrire les balises indispensables de notre site web au format HTML.

```
inc/haut.inc.php

<!Doctype html>
<html>
  <head>
    <title>Mon Site</title>
    <link rel="stylesheet" href="<?php echo RACINE_SITE; ?>inc/css/style.css">
  </head>
  <body>
    <header>
      <div class="conteneur">
        <div>
          <a href="" title="Mon Site">MonSite.com</a>
        </div>
        <nav>
          <a href="<?php echo RACINE_SITE; ?>inscription.php">Inscription</a>
          <a href="<?php echo RACINE_SITE; ?>connexion.php">Connexion</a>
          <a href="<?php echo RACINE_SITE; ?>boutique.php">Accès à la boutique</a>
          <a href="<?php echo RACINE_SITE; ?>panier.php">Voir votre panier</a>
        </nav>
      </div>
    </header>
    <section>
      <div class="conteneur">
```

Ce fichier nous permet de déclarer le haut du site avec le menu comprenant quelques liens (anticipés, puisque les pages ne sont pas encore créées).

Les balises ne sont pas toutes fermées, c'est volontaire puisque nous allons mettre du contenu entre le fichier haut.inc.php et bas.inc.php.

### Etape 3. Ecriture du fichier inc/bas.inc.php

Nous allons fermer les balises indispensables de notre site web au format HTML.

```
inc/bas.inc.php

</div>
</section>
<footer>
  <div class="conteneur">
    <code><?php echo date('Y); ?> - Tous droits réservés - MonNom MonPrenom.</code>
  </div>
</footer>
</body>
```

```
</html>
```

Ce fichier à l'avantage d'être simple !

### Etape 3. Ecriture du fichier inc/css/style.css

Nous allons écrire un peu de code permettant de faire une mise en forme minimale.

```
inc/css/style.css
```

```
*{ margin: 0; }
a{ text-decoration: none; color: #000; }
.conteneur{ margin: 0 auto; max-width: 1170px; }
***** HAUT *****/
header { background: #000000; padding: 5px; text-align: center; }
header span{ color: #fff; font-weight: bold; text-transform: uppercase; margin-right: 5%; }
header nav{ display: inline; }
header a{ color: #fff; text-decoration: none; padding: 5px; }
header nav a:hover{ background: #04bafe; }
***** MILIEU *****/
section{ padding: 30px; min-height: 800px; }
***** BAS *****/
footer{ background: #000; color: white; text-align: center; padding: 7px 0; }
***** GENERAL *****/
.erreur{ background: #ff0000; padding: 5px; margin: 5px; }
.validation{ background: #669933; padding: 5px; margin: 5px; }
```

Durant le projet, nous aurons l'occasion de revenir dans ce fichier pour l'alimenter d'avantages

### Etape 3. Ecriture de notre fichier inc/fonction.inc.php

Nous allons créer 2 fichiers pour nous aider dans la conception du site web.

- Une fonction que nous nommerons `executeRequete` pour devinez quoi... exécuter des requêtes SQL !

En effet, échanger des informations avec la base est une action que nous devrons faire certainement plusieurs fois par page web, autant nous faciliter un peu la vie avec une fonction déjà prête.

- Une fonction que nous nommerons `debug` pour nous debugger !

En PHP, il est souvent nécessaire d'effectuer des `var_dump` ou `print_r` pour voir le contenu d'un tableau array, d'un objet ou de variables, nous allons donc prévoir un code en conséquence afin de gagner du temps.

```
inc/fonction.inc.php
```

```
function executeRequete($req)
{
    global $mysqli;
    $resultat = $mysqli->query($req);
    if(!$resultat) //
    {
        die("Erreur sur la requete sql.<br>Message : " . $mysqli->error . "<br>Code: " . $req);
    }
    return $resultat; //
}

function debug($var, $mode = 1)
{
    echo '<div style="background: orange; padding: 5px; float: right; clear: both; ">';
    $trace = debug_backtrace();
    $trace = array_shift($trace);
    echo 'Debug demandé dans le fichier : ' . $trace[file] . ' à la ligne ' . $trace[line];
    if($mode === 1)
    {
        echo '<pre>'; print_r($var); echo '</pre>';
    }
    else
    {
        echo '<pre>'; var_dump($var); echo '</pre>';
    }
    echo '</div>';
}
```

#### La fonction executeRequete

`function executeRequete($req)` La fonction sera destinée à recevoir 1 argument entrant (la requête SQL arrivera dans la variable de réception \$req prévue à cet effet).

`global $mysqli;` permet d'avoir accès à la variable \$mysqli définie dans le fichier `init.inc.php` (espace global) à l'intérieur de notre fonction (espace local).

```

$résultat = $mysqli->query($req); on exécute la requête reçue en argument et on gardera les résultats dans la variable $résultat.

if(!$résultat) si la variable $résultat renvoie false, c'est qu'il y a une erreur de requête SQL.

die("Erreur sur la requête sql.<br>Message : " . $mysqli->error . "<br>Code: " . $req); Dans le cas où la requête échoue, on lui demande d'adresser 1 message et d'arrêter l'exécution du code avec l'utilisation de die.

return $résultat; en cas d'une requête de SELECTION, on retournera un objet issu de la classe mysqli_result. Sinon (pour INSERT/UPDATE/DELETE), nous retournerons un boolean TRUE (1).

```

#### La fonction debug

```

function debug($var, $mode = 1) La fonction sera destinée à recevoir 1 ou 2 argument(s) entrant(s). En premier ce sera la variable/array/object à explorer et en second ce sera 1 mode d'affichage.

$trace = debug_backtrace(); Fonction prédéfinie retournant un tableau Array contenant des informations tel que la ligne et le fichier où est exécuté la fonction.

$trace = array_shift($trace); Extrait la première valeur d'un tableau et la retourne. Dans notre cas cela permet de retirer une dimension au tableau array $trace.

echo "Debug demandé dans le fichier : $trace[file] à la ligne $trace[line].<br>"; Au moment de l'affichage, cela permettra de savoir de quel fichier vient la demande de debug

if($mode === 1)<br>; Si le mode 1 est précisé en argument (ou qu'il n'y a pas d'informations contraires), nous ferons un print_r

else Sinon, nous ferons un var_dump

```

137

## Etape 4. Ecriture de la page inscription.php

Nous allons commencer notre site web par l'espace membre et donc la page d'inscription !

Pour cela, commençons par le formulaire HTML :

```

inscription.php

<?php require_once("inc/init.inc.php"); ?>
<?php require_once("inc/haut.inc.php"); ?>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo" maxlength="20" placeholder="votre pseudo" pattern="[a-zA-Z0-9-_]{1,20}" title="caractères acceptés : a-zA-Z0-9-_<br>" required="required"><br><br>

    <label for="mdp">Mot de passe</label><br>
    <input type="password" id="mdp" name="mdp" required="required"><br><br>

    <label for="nom">Nom</label><br>
    <input type="text" id="nom" name="nom" placeholder="votre nom"><br><br>

    <label for="prenom">Prénom</label><br>
    <input type="text" id="prenom" name="prenom" placeholder="votre prénom"><br><br>

    <label for="email">Email</label><br>
    <input type="email" id="email" name="email" placeholder="exemple@gmail.com"><br><br>

    <label for="civilité">Civilité</label><br>
    <input name="civilité" value="m" checked="" type="radio">Homme
    <input name="civilité" value="f" type="radio">Femme<br><br>

    <label for="ville">Ville</label><br>
    <input type="text" id="ville" name="ville" placeholder="votre ville" pattern="[a-zA-Z0-9-_]{5,15}" title="caractères acceptés : a-zA-Z0-9-_<br><br>"

    <label for="cp">Code Postal</label><br>
    <input type="text" id="code_postal" name="code_postal" placeholder="code postal" pattern="[0-9]{5}" title="5 chiffres requis : 0-9"><br><br>

    <label for="adresse">Adresse</label><br>
    <textarea id="adresse" name="adresse" placeholder="votre dresse" pattern="[a-zA-Z0-9-_]{5,15}" title="caractères acceptés : a-zA-Z0-9-_<br><br>*></textarea><br><br>

```

```

<input type="submit" name="inscription" value="S'inscrire">
</form>
<?php require_once("inc/bas.inc.php"); ?>

```

Nous incluons le fichier init.inc.php, le haut du site, le bas du site, et entre le haut et le bas nous mettons notre formulaire html afin que nos futurs internautes puissent s'inscrire.

Il est très important que les attributs name du formulaire soient prévus afin de pouvoir récupérer et exploiter les saisies en PHP. De préférence, nous pouvons garder les mêmes name que le nom de nos champs dans notre base de données.

**MONSITE.COM**

**Inscription** Connexion Accès à la boutique Voir votre panier

Pseudo  
votre pseudo

Mot de passe

Nom  
votre nom

Prénom  
votre prénom

Email  
exemple@gmail.com

Sexe  
Homme Femme

Ville  
votre ville

Cp  
code postal

Adresse  
votre dresse

S'inscrire

Résultat - inscription.php

Il faut aussi prévoir la partie traitement en PHP, voici la suite du code :

```

inscription.php

<?php require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
if($_POST)
{
    debug($_POST);
    $verif_caractere = preg_match("#^[a-zA-Z0-9_-]+#$", $_POST['pseudo']);
    if(!$verif_caractere && (strlen($_POST['pseudo']) < 1 || strlen($_POST['pseudo']) > 20) )
    {
        $contenu .= "<div class='erreur>Le pseudo doit contenir entre 1 et 20 caractères. <br> Caractère accepté : Lettre de A à Z et chiffre de 0 à 9</div>";
    }
    else
    {
        $membre = executeRequete("SELECT * FROM membre WHERE pseudo='".$_POST['pseudo']."'");
        if($membre->num_rows > 0)
        {
            $contenu .= "<div class='erreur>Pseudo indisponible. Veuillez en choisir un autre svp.</div>";
        }
        else
        {
            // $_POST['mdp'] = md5($_POST['mdp']);
            foreach($_POST as $indice => $valeur)

```

```

    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("INSERT INTO membre (pseudo, mdp, nom, prenom, email, civilete, ville, code_postal, adresse) VALUES ('$_POST[pseudo]',"
'$_POST[mdp]', '$_POST[nom]', '$_POST[prenom]', '$_POST[email]', '$_POST[civilete]', '$_POST[ville]', '$_POST[code_postal]', '$_POST[adresse]'");
    $contenu .= "<div class='validation'>Vous êtes inscrit à notre site web. <a href='connexion.php'><u>Cliquez ici pour vous connecter</u></a></div>";
}
//----- AFFICHAGE HTML -----//?
<?php require_once("inc/haut.inc.php"); ?>
<?php echo $contenu; ?>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
    <label for="pseudo">Pseudo</label><br>
    <input type="text" id="pseudo" name="pseudo" maxlength="20" placeholder="votre pseudo" pattern="^a-zA-Z0-9-_]{1,20}" title="caractères acceptés : a-zA-Z0-9-_ " required="required"><br><br>

    <label for="mdp">Mot de passe</label><br>
    <input type="password" id="mdp" name="mdp" required="required"><br><br>

    <label for="nom">Nom</label><br>
    <input type="text" id="nom" name="nom" placeholder="votre nom"><br><br>

    <label for="prenom">Prénom</label><br>
    <input type="text" id="prenom" name="prenom" placeholder="votre prénom"><br><br>

    <label for="email">Email</label><br>
    <input type="email" id="email" name="email" placeholder="exemple@gmail.com"><br><br>

    <label for="civilité">Civilité</label><br>
    <input name="civilité" value="m" checked="" type="radio">Homme
    <input name="civilité" value="f" type="radio">Femme<br><br>

    <label for="ville">Ville</label><br>
    <input type="text" id="ville" name="ville" placeholder="votre ville" pattern="^a-zA-Z0-9-_]{5,15}" title="caractères acceptés : a-zA-Z0-9-_ " required="required"><br><br>

    <label for="cp">Code Postal</label><br>
    <input type="text" id="code_postal" name="code_postal" placeholder="code postal" pattern="^0-9]{5}" title="5 chiffres requis : 0-9"><br><br>

    <label for="adresse">Adresse</label><br>
    <textarea id="adresse" name="adresse" placeholder="votre dresse" pattern="^a-zA-Z0-9-_]{5,15}" title="caractères acceptés : a-zA-Z0-9-_ " required="required"><br><br>

    <input type="submit" name="inscription" value="S'inscrire">
</form>

<?php require_once("inc/bas.inc.php"); ?>

```

### Quelques Explications

**if(\$\_POST)** Cette condition IF permet de détecter si l'internaute à cliquer sur le bouton submit pour s'inscrire.

**debug(\$\_POST);** Si l'internaute sollicite une inscription, nous allons utiliser notre fonction debug afin de voir les saisies qu'il a postées (le temps de faire des tests). Cette fonction a été inclut par le fichier init.inc.php puisqu'il inclue lui même fonction.inc.php

**\$verif\_caractere = preg\_match('#^a-zA-Z0-9-\_]+#\$', \$\_POST['pseudo']);** Nous vérifions qu'il n'y ai pas de mauvais caractère dans le pseudo. (return 0 si mauvais caractère dans le pseudo, 1 sinon). vous pouvez écrire echo \$verif\_caractere;

**preg\_match()** est une expression régulière (regex) toujours entourée du symbole # dieze afin de préciser des options choisies :

**^** désigne le début de la chaîne.

**\$** désigne la fin de la chaîne.

**+ est présent pour dire que les lettres autorisées peuvent apparaître plusieurs fois.**

**if(!\$verif\_caractere && (strlen(\$\_POST['pseudo']) < 1 || strlen(\$\_POST['pseudo']) > 20) )** A travers cette condition, nous vérifions qu'il n'y ai pas un caractère interdit ou un problème de taille sur le pseudo. Cela reste faible, dans une version plus aboutie il faudrait penser à renforcer les contrôles (sur le pseudo mais aussi les autres champs).

**\$contenu .= "<div class='erreur'>Le pseudo doit contenir entre 1 et 20 caractères. <br> Caractère accepté : Lettre de A à Z et chiffre de 0 à 9</div>";** En cas d'erreur, nous allons en informer l'internaute mais pas tout de suite ! sinon nous serions au dessus du doctype niveau code-source, nous allons donc retenir l'affichage du message dans la variable \$contenu que nous remplissons afin de l'afficher plus tard.

**else** Sinon, la variable \$contenu est vide c'est qu'il n'y a pas eu d'erreur précédemment.

**\$membre = executeRequete("SELECT \* FROM membre WHERE pseudo='\$\_POST[pseudo]'");** Nous allons utiliser notre fonction executeRequete pour allez voir si le pseudo que l'internaute tente de prendre n'est pas déjà attribué à un autre membre.

**if(\$membre->num\_rows > 0)** Si la requête renvoie plus de 0 résultat (donc au moins 1), c'est que le pseudo est déjà attribué à quelqu'un d'autre.

```
$contenu .= "<div class='erreur'>Pseudo indisponible. Veuillez en choisir un autre svp.</div>"; Nous invitons le membre à choisir un autre pseudo si celui qu'il convoite est déjà attribué.
```

```
else Sinon, on lance l'inscription.
```

```
// $_POST['mdp'] = md5($_POST['mdp']); ce code est en commentaire, nous pouvons crypter le mot de passe afin qu'il ne soit pas affiché en clair dans la base de données. !\ Attention, si vous activez cette ligne, il faudra penser à ajouter 1 ligne de cryptage au moment de la connexion (le membre envoie son mot de passe, on le recryptera pour le comparer avec la chaîne cryptée en base). Pour la suite du cours, nous ne garderons pas cette ligne et travaillerons avec des mots de passe en clair (le temps de l'entraînement).
```

```
foreach($_POST as $indice => $valeur){ $_POST[$indice] = htmlEntities(addSlashes($valeur));} nous bouclons sur toutes les saisies afin de les passer dans les fonctions prédéfinies PHP htmlEntities et addSlashes. !\ Cela permet d'effectuer 1 premier traitement mais ce n'est pas pour autant complètement sécurisé.
```

```
executeRequete("INSERT INTO membre (pseudo, mdp, nom, prenom, email, civilite, ville, code_postal, adresse) VALUES ('$_POST[pseudo]', '$_POST[mdp]', '$_POST[nom]', '$_POST[prenom]', '$_POST[email]', '$_POST[civilite]', '$_POST[ville]', '$_POST[code_postal]', '$_POST[adresse]')"); Cette requête permet d'insérer le membre dans la base ! C'est à ce moment-là que l'enregistrement se fait.
```

```
$contenu .= "<div class='validation'>Vous êtes inscrit à notre site web. <a href=\"connexion.php\"><u>Cliquez ici pour vous connecter</u></a></div>"; Nous l'informons de son inscription et lui proposons de se connecter. (nous pourrions aussi prévoir de connecter l'internaute automatiquement).
```

Résultat

inscription.php

MONSITE.COM

Inscription Connexion Accès à la boutique Voir votre panier

Vous êtes inscrit à notre site web. [Cliquez ici pour vous connecter](#)

Pseudo  
juju

Mot de passe  
\*\*\*\*\*

Nom  
Cottet

Prénom  
Julien

Email  
julien.cottet@gmail.com

Civilité  
 Homme  Femme

Ville  
Paris

Code Postal  
75015

Adresse  
300 rue de vaugirard

S'inscrire

Debug demandé dans le fichier : C:\wamp\www\dev\tutomooc\site\inscription.php à la ligne 5.  
Array  
(  
 [pseudo] => juju  
 [mdp] => soleil  
 [nom] => Cottet  
 [prenom] => Julien  
 [email] => julien.cottet@gmail.com  
 [civilite] => m  
 [ville] => Paris  
 [code\_postal] => 75015  
 [adresse] => 300 rue de vaugirard  
 [inscription] => S'inscrire  
)

Vous devriez retrouver la trace de votre 1er membre dans la base de données :

← T →	id_membre	pseudo	mdp	nom	prenom	email	civilite	ville	code_postal	adresse	statut
	1	juju	soleil	Cottet	Julien	julien.cottet@gmail.com	m	Paris	75015	300 rue de vaugirard	0

Maintenant que votre formulaire fonctionne, inscrivez plusieurs membres différents histoire que nous ayons plusieurs enregistrements pour faire des tests plus tard :

id_membre	pseudo	mdp	nom	prenom	email	civilite	ville	code_postal	adresse	statut
1	juju	soleil	Cottet	Julien	julien.cottet@gmail.com	m	Paris	75015	300 rue de vaugirard	0
2	lamarie	planete	thoyer	marie	marie.thoyer@yahoo.fr	f	Lyon	69003	10 rue paul bert	0
3	fab	avatar13	grand	fabrice	fabrice.grand@gmail.com	m	Marseille	13009	70 rue de la république	0
4	membre	membre	membre	membre	membre@example.com	f	Toulouse	31000	55 rue bayard	0
5	admin	admin	admin	admin	admin@example.com	m	Paris	75015	33 rue mademoiselle	1

Rendez-vous dans PhpMyAdmin pour modifier le statut de l'un des membres afin de mettre son statut sur "1". Cela nous permettra d'avoir 1 administrateur (indispensable pour la suite des tests).

138

## Etape 5. Ecriture de la page connexion.php

Maintenant que nous avons une page d'inscription fonctionnelle et plusieurs inscrits, nous allons attaquer la page de connexion !

Pour cela, commençons par le formulaire HTML :

```
inscription.php

<?php require_once("inc/init.inc.php"); ?>
<?php require_once("inc/haut.inc.php"); ?>

<form method="post" action="<br>
<label for="pseudo">Pseudo</label><br>
<input type="text" id="pseudo" name="pseudo"><br> <br>

<label for="mdp">Mot de passe</label><br>
<input type="text" id="mdp" name="mdp"><br><br>

<input type="submit" value="Se connecter">
</form>

<?php require_once("inc/bas.inc.php"); ?>
```

Nous incluons le fichier init.inc.php, le haut du site, le bas du site, et entre le haut et le bas nous mettons notre formulaire html afin que nos futurs internautes puissent se connecter.

Il est très important que les attributs name du formulaire soit prévue afin de pouvoir récupérer et exploiter les saisies en PHP. De préférence, pour une meilleure cohérence, nous pouvons garder les mêmes name que le nom de nos champs dans notre base de données.

Pseudo

votre pseudo

Mot de passe

Nom

votre nom

Prénom

votre prénom

Email

exemple@gmail.com

Sexe

Homme Femme

Ville

votre ville

Cp

code postal

Adresse

votre dresse

S'inscrire

Résultat - inscription.php

Il faut aussi prévoir la partie traitement en PHP, voici la suite du code :

## connexion.php

```

<?php require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
if($_POST)
{
    // $contenu .= "pseudo : " . $_POST['pseudo'] . "<br>mdp : " . $_POST['mdp'] . "";
    $résultat = executeRequete("SELECT * FROM membre WHERE pseudo='".$_POST['pseudo']."'");

    if($résultat->num_rows != 0)
    {
        // $contenu .= '<div style="background:green">pseudo connu!</div>';
        $membre = $résultat->fetch_assoc();
        if($membre['mdp'] == $_POST['mdp'])
        {
            // $contenu .= '<div class="validation">mdp connu!</div>';
            foreach($membre as $indice => $element)
            {
                if($indice != 'mdp')
                {
                    $_SESSION['membre'][$indice] = $element;
                }
            }
            header("location:profil.php");
        }
        else
        {
            $contenu .= '<div class="erreur">Erreur de MDP</div>';
        }
    }
    else
    {
        $contenu .= '<div class="erreur">Erreur de pseudo</div>';
    }
}
//----- AFFICHAGE HTML -----
?>
<?php require_once("inc/haut.inc.php"); ?>
<?php echo $contenu; ?>

<form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']); ?>">
    <label for="pseudo">Pseudo</label><br>

```

```

<input type="text" id="pseudo" name="pseudo"><br> <br>
<label for="mdp">Mot de passe</label><br>
<input type="text" id="mdp" name="mdp"><br><br>
<input type="submit" value="Se connecter">
</form>
<?php require_once("inc/bas.inc.php"); ?>

```

### Quelques Explications

`if($_POST)` Cette condition IF permet de détecter si l'internaute à cliqué sur le bouton submit pour se connecter.

`$résultat = executeRequete("SELECT * FROM membre WHERE pseudo='".$_POST[pseudo]'")`; Nous allons utiliser notre fonction `executeRequete` pour aller consulter la base afin de savoir si le pseudo avec lequel l'internaute tente de se connecter correspond bien à 1 compte réel sur notre site web. Y'a t'il un enregistrement correspondant dans notre base ?

`if($résultat->num_rows != 0)` Si le nombre de retours est différent de 0 (donc 1 logiquement :p), c'est que le pseudo est connu et que le compte existe, on peut avancer...

`else` Sinon, nous informons l'internaute qu'il y a une erreur sur son pseudo...

`$membre = $résultat->fetch_assoc();` Revenons sur le cas du pseudo valide, nous devons absolument traiter (`fetch_assoc`) pour connaître les informations récupérées en base. En effet, nous devons savoir si le membre a le bon pseudo mais aussi s'il possède le bon mot de passe associé.

`if($membre['mdp'] == $_POST['mdp'])` On compare le mdp posté (dans le formulaire de connexion) avec le mdp du membre (récupéré dans la base de données), s'il s'agit du même mdp dans les deux cas, on crée à l'internaute une session et on la remplit avec certaines informations (c'est ce qui permet réellement de connecter et de maintenir connecté quelqu'un sur 1 site web).

`else` Sinon, le mdp est mauvais et nous informons l'internaute.

`foreach($membre as $indice => $element){if($indice != 'mdp'){ $_SESSION['membre'][$indice] = $element; }}` Nous créons une session avec les éléments de la base de données. La boucle `foreach` évite d'écrire les lignes suivantes : `$_SESSION['membre']['id_membre'] = $membre['id_membre']; $_SESSION['membre']['pseudo'] = $membre['pseudo']; etc..` Par sécurité et comme nous n'en n'aurons pas besoin, nous ne conserverons pas le mdp dans la session (condition IF).

Pour rappel, sans le `session_start()` placée dans le fichier `init.inc.php`, nous n'aurions pas pu se servir du système de session en PHP

`header("location:profil.php");` Si l'accouplement pseudo/mot de passe est bon, nous redirigeons l'internaute vers sa page de profil (puisque il est maintenant connecté !)

The screenshot shows a login interface. At the top, there's a navigation bar with links for 'Inscription', 'Connexion' (which is highlighted in blue), 'Accès à la boutique', and 'Voir votre panier'. Below the navigation, there's a form area with two input fields: one for 'Pseudo' and one for 'Mot de passe'. Both fields have placeholder text ('Pseudo' and 'Mot de passe' respectively). Below the fields is a blue 'Se connecter' button. The entire form is set against a white background with a thin gray border.

Résultat - inscription.php

Faites plusieurs tests !

- Avec 1 mauvais pseudo,
- Avec 1 bon pseudo mais 1 mauvais mdp,
- Avec 1 bon pseudo et 1 mauvais mdp !

ps : dans le cas d'un bon pseudo et mot de passe, si vous arrivez sur une page "non trouvée" c'est tout à fait normal car nous n'avons pas encore créé la page de profil.

Quoi qu'il en soit "être connecté à un site web" ne signifie pas seulement avoir le bon pseudo et le bon mot de passe.

Pour maintenir une connexion il faut avoir un fichier de session (sur le serveur du site) et un fichier cookie sur votre ordinateur pour assurer la liaison.

Session dans le dossier /tmp/ du serveur :



Ce fichier a été créé par `session_start()` (placé dans le fichier `inc/init.inc.php`) automatiquement.

Ce même fichier a été rempli par la superglobale `$_SESSION` lors de la connexion.

Cookie sur l'ordinateur de l'internaute :

A screenshot of a browser's settings interface. On the left, a sidebar shows options like Général, Recherche, Contenu, Applications, Vie privée (which is selected), Sécurité, Sync, and Avancé. The main area is titled 'Vie privée' and contains sections for 'Pistage' (with a checkbox for 'Indiquer aux sites que je ne souhaite pas être pisté') and 'Historique' (with checkboxes for 'Toujours utiliser le mode de navigation privée', 'Conserver l'historique de navigation et de recherche', 'Accepter les cookies', and 'Accepter les cookies tiers'). Below these are sections for 'Barre d'adresse' and 'Autres'. On the right, a 'Cookies' panel is open. It has a search bar and a table titled 'Cookies'. The table shows a single cookie: 'Nom du cookie' is 'PHPSESSID', 'Site' is 'localhost', and 'Contenu' is '3blp1q6mh1jf22gsc3q0aeioff6'. Below the table, details for this cookie are shown: 'Nom : PHPSESSID', 'Contenu : 3blp1q6mh1jf22gsc3q0aeioff6', 'Hôte : localhost', 'Chemin : /', 'Envoyé pour : Tout type de connexion', and 'Expire : À la fin de la session'. At the bottom of the panel are buttons for 'Supprimer le cookie sélectionné', 'Tout supprimer', and 'Fermer'.

A présent, nos visiteurs peuvent s'inscrire et aussi se connecter !

C'est un bon début mais si les internautes font la démarche de s'inscrire et de se connecter c'est pour arriver quelque part et ainsi profiter d'un espace réservé aux membres sur le site web.

Nous allons donc prévoir une page de profil avec quelques informations !

Pour récupérer les informations, nous pourrons nous servir du fichier de session (*par l'intermédiaire de la superglobale \$\_SESSION*) que nous avons créé au moment de la connexion. (C'est pratique car ces informations de session seront accessibles sur tout le site web).

Avant de rentrer dans l'espace de profil nous devons nous assurer que l'internaute qui tente d'y accéder à les droits nécessaires (c'est à dire, être passé par la page de connexion avec un bon pseudo et un bon mot de passe. En gros avoir un fichier de session).

Par exemple, vous ne pouvez pas accéder à votre boîte de réception d'email sans vous être connecté au préalable. De la même manière, nous allons tester si 1 fichier de session existe avant d'accepter l'internaute sur l'espace de profil.

Pour cela, nous allons écrire dans notre fichier inc/fonction.inc.php :

```
inc/fonction.inc.php

function executeRequete($req)
{
    global $mysqli;
    $résultat = $mysqli->query($req);
    if(!$résultat)
    {
        die("Erreur sur la requête sql.<br>Message : " . $mysqli->error . "<br>Code: " . $req);
    }
    return $résultat;
}
//-----
function debug($var, $mode = 1)
{
    echo '<div style="background: orange; padding: 5px; float: right; clear: both; ">';
    $trace = debug_backtrace();
    $trace = array_shift($trace);
    echo 'Debug demandé dans le fichier : ' . $trace[file] . ' à la ligne ' . $trace[line];
    if($mode === 1)
    {
        print '<pre>'; print_r($var); print '</pre>';
    }
    else
    {
        print '<pre>'; var_dump($var); print '</pre>';
    }
    echo '</div>';
}
//-----
function internauteEstConnecte()
{
    if(!isset($_SESSION['membre'])) return false;
    else return true;
}
//-----
function internauteEstConnecteEtEstAdmin()
{
    if(internauteEstConnecte() && $_SESSION['membre']['statut'] == 1) return true;
    else return false;
}
```

### Explications

La fonction `internauteEstConnecte()` va nous permettre de savoir si l'internaute est connecté par une simple condition :

```
if(!isset($_SESSION['membre'])) return false; si la session membre n'existe pas, c'est que l'internaute n'est pas passé par la page de connexion (ou alors qu'il a été déconnecté depuis). on retournera la valeur "FALSE" pour dire "Faux l'internaute n'est pas connecté".
```

```
else sinon, c'est que la session membre existe et donc que l'internaute est connecté (avec 1 fichier de session + cookie). on retournera la valeur "TRUE" pour dire "Vrai l'internaute est connecté".
```

L'avantage d'avoir mis ce code dans une fonction `internauteEstConnecte()` et non pas directement dans la page web c'est qu'il sera plus facile de s'en resservir sur les autres pages web (plutôt que re-trimballer un même morceau code en copier/coller d'un fichier à l'autre). Cela sera pratique de savoir si l'internaute est connecté à divers endroits du site web.

La fonction `internauteEstConnecteEtEstAdmin()` va nous permettre de savoir si l'internaute est connecté en tant qu'administrateur (statut à 1) ou en tant que membre (statut à 0) :

```
if(internauteEstConnecte() && $_SESSION['membre']['statut'] == 1 si la fonction internauteEstConnecte() renvoie "TRUE", l'internaute est connecté (avec 1 fichier de session + cookie), on vérifie donc si son statut est à 1. Si oui, nous renverrons TRUE pour dire "Vrai, cet internaute est connecté et est admin".
```

```
else sinon, c'est soit que l'internaute n'est pas connecté ou soit que l'internaute est connecté mais sans avoir les droits d'administration, nous
```

renverrons donc "FALSE" pour dire "Faux cet internaute n'est pas administrateur".

L'avantage d'avoir mis ce code dans une fonction `internauteEstConnecte()` et non pas directement dans la page web c'est qu'il sera plus facile de s'en reserver sur les autres pages web (plutôt que re-trimbaler un même morceau code en copier/coller d'un fichier à l'autre). Cela sera pratique de savoir si l'internaute est connecté à divers endroits du site web.

Maintenant que nous avons 2 fonctions qui vont pouvoir nous aider, nous allons créer la page de profil :

```
profil.php

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
if(!internauteEstConnecte()) header("location:connexion.php");
// debug($_SESSION);
$contenu .= '<p class="centre">Bonjour <strong>' . $_SESSION["membre"]["pseudo"] . '</strong></p>';
$contenu .= '<div class="cadre"><h2> Voici vos informations </h2>';
$contenu .= '<p> votre email est ' . $_SESSION["membre"]["email"] . '<br>';
$contenu .= 'votre ville est ' . $_SESSION["membre"]["ville"] . '<br>';
$contenu .= 'votre cp est ' . $_SESSION["membre"]["code_postal"] . '<br>';
$contenu .= 'votre adresse est: ' . $_SESSION["membre"]["adresse"] . '</p></div><br><br>';
//----- AFFICHAGE HTML -----
require_once("inc/haut.inc.php");
echo $contenu;
require_once("inc/bas.inc.php");
```

#### Explications

Nous vérifions `if(!internauteEstConnecte())` si l'internaute (!) N'EST PAS connecté (le point d'exclamation demande si la fonction renvoie false, donc si l'internaute n'est pas connecté).

Si l'internaute n'est pas connecté, il n'a rien à faire sur la page de profil, nous le renvoyons vers la page de connexion  
`header("location:connexion.php") ;`.

Pour construire la page de profil, nous piochons dans le fichier session (dans le dossier /tmp/ sur le serveur, par l'intermédiaire de la superglobale `$_SESSION`) afin d'afficher les informations de l'internaute connecté.

Connectez-vous, vous verrez que notre profil est encore incomplet mais qu'il se construit peu à peu :

Résultat

-

profil.php

Bonjour **juju****Voici vos informations de profil**

votre email est: julien.cottet@gmail.com  
votre ville est: Paris  
votre cp est: 75015  
votre adresse est: 300 rue de vaugirard

2015 - Tous droits réservés.

Plus tard, nous pourrons ajouter le suivi des commandes dans l'espace de profil d'un membre.

**Etape 6. Un menu évolutif en fonction de notre statut**

Au sein de notre site web, il y a plusieurs statuts :

- Visiteur : correspond à un internaute non connecté
- Membre : correspond à un internaute connecté
- Admin : correspond à un internaute connecté avec des droits d'administration

Il est donc normal que selon son statut nous n'ayons pas accès aux mêmes droits sur le site web.

Voici donc la mise à jour de notre fichier inc/haut.inc.php (au niveau de la navigation/menu) :

```
inc/haut.inc.php
```

```
<!Doctype html>
<html>
<head>
<title>Mon Site</title>
<link rel="stylesheet" href="php echo RACINE_SITE; ?&gt;inc/css/style.css"&gt;
&lt;/head&gt;
&lt;body&gt;
&lt;header&gt;
&lt;div class="conteneur"&gt;
&lt;div&gt;
&lt;a href="" title="Mon Site"&gt;MonSite.com&lt;/a&gt;
&lt;/div&gt;
&lt;nav&gt;
&lt;?php
if(internauteEstConnecteEtEstAdmin())
{
    echo '&lt;a href="' . RACINE_SITE . 'admin/gestion_membre.php"&gt;Gestion des membres&lt;/a&gt;';
    echo '&lt;a href="' . RACINE_SITE . 'admin/gestion_commande.php"&gt;Gestion des commandes&lt;/a&gt;';
    echo '&lt;a href="' . RACINE_SITE . 'admin/gestion_boutique.php"&gt;Gestion de la boutique&lt;/a&gt;';
}</pre
```

```

if(internauteEstConnecte())
{
    echo '<a href="" . RACINE_SITE . "profil.php">Voir votre profil</a>';
    echo '<a href="" . RACINE_SITE . "boutique.php">Accès à la boutique</a>';
    echo '<a href="" . RACINE_SITE . "panier.php">Voir votre panier</a>';
    echo '<a href="" . RACINE_SITE . "connexion.php?action=deconnexion">Se déconnecter</a>';
}
else
{
    echo '<a href="" . RACINE_SITE . "inscription.php">Inscription</a>';
    echo '<a href="" . RACINE_SITE . "connexion.php">Connexion</a>';
    echo '<a href="" . RACINE_SITE . "boutique.php">Accès à la boutique</a>';
    echo '<a href="" . RACINE_SITE . "panier.php">Voir votre panier</a>';
}
?>
</nav>
</div>
</header>
<section>
<div class="conteneur">

```

#### Explications

Si l'internaute est Administrateur **if(internauteEstConnecteEtEstAdmin())** nous lui proposerons des liens de gestion (backOffice) pour gérer ses produits, ses commandes, ses membres, etc.

Si l'internaute est Membre **if(internauteEstConnecte())** nous lui proposerons plusieurs liens dont son espace de profil

Si l'internaute est Visiteur **else** nous lui proposerons d'autres liens notamment l'inscription et la connexion

En conclusion :

- 1 visiteur aura donc accès à 4 liens
- 1 membre aura également 4 liens affichés
- 1 administrateur verra 7 liens sur la page web (1 administrateur est aussi 1 membre).

Au passage, pour gérer la deconnexion, nous enverrons dans l'url une information **?action=deconnexion** afin de pouvoir détecter que l'internaute ai bien cliqué sur le lien "se déconnecter".

Sinon comment le savoir ? il nous faut une information dans l'url que nous pourrons aller chercher plus tard avec la superglobale **\$\_GET** afin d'associer le traitement permettant de déconnecter l'internaute.

Nous gérerons la deconnexion sur la page de connexion.

#### **Etape 6. La deconnexion et la fin de l'espace membre**

Que l'on puisse se connecter, c'est super, mais il faut aussi pouvoir se déconnecter :p.

Pour terminer l'espace membre, nous allons mettre à jour la page de connexion :

```

connexion.php

<?php require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
if(isset($_GET['action']) && $_GET['action'] == "deconnexion")
{
    session_destroy();
}
if(internauteEstConnecte())
{
    header("location:profil.php");
}
if($_POST)
{
    // $contenu .= "pseudo : " . $_POST[pseudo] . "<br>mdp : " . $_POST[mdp] . "";
    $resultat = executeRequete("SELECT * FROM membre WHERE pseudo=$_POST[pseudo]");
    if($resultat->num_rows != 0)
    {
        // $contenu .= '<div style="background:green">pseudo connu!</div>';
        $membre = $resultat->fetch_assoc();
        if($membre[mdp] == $_POST[mdp])
        {
            // $contenu .= '<div class="validation">mdp connu!</div>';
            foreach($membre as $indice => $element)
            {
                if($indice != 'mdp')
                {
                    $_SESSION[membre][$indice] = $element;
                }
            }
            header("location:profil.php");
        }
    }
}
?>
```

```

        else
        {
            $contenu .= '<div class="erreur">Erreur de MDP</div>';
        }
    else
    {
        $contenu .= '<div class="erreur">Erreur de pseudo</div>';
    }
//----- AFFICHAGE HTML -----
?>
<?php require_once("inc/haut.inc.php"); ?>
<?php echo $contenu; ?>

<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
<label for="pseudo">Pseudo</label><br>
<input type="text" id="pseudo" name="pseudo"><br> <br>
<label for="mdp">Mot de passe</label><br>
<input type="text" id="mdp" name="mdp"><br><br>
<input type="submit" value="Se connecter">
</form>
<?php require_once("inc/bas.inc.php"); ?>

```

### Quelques Explications

**if(isset(\$\_GET['action']) && \$\_GET['action'] == "deconnexion")** Si l'internaute clic sur le lien deconnexion, nous arriverons sur la page connexion.php avec l'information suivante dans l'url ?action=deconnexion. C'est la raison pour laquelle nous utilisons la superglobale \$\_GET afin de détecter cette action et de déconnecter l'internaute via session\_destroy();.

Au passage, nous en profitons pour améliorer l'espace membre et dire que si l'internaute est déjà connecté mais qu'il tente d'aller sur la page de connexion (volontairement ou involontairement), nous le renverrons automatiquement dans son espace de profil : header("location:profil.php");.

Cela paraît logique, est-ce que vous voyez la page de connexion GMAIL alors que vous êtes connecté ? non, cela a été prévu dans le code...

Et oui le code c'est aussi ça, PENSER A TOUT ! le fonctionnel, les contrôles, la sécurité, traquer les éventuelles incohérences, la recherche, les tests, etc. On dit qu'un site est totalement terminé lorsque tous les cas sont prévus par le script ! Autant dire qu'il y a beaucoup de sites qui ne sont pas terminés...

**Vous pouvez faire des tests :**

- Connectez-vous et déconnectez-vous ! accédez au site en tant qu'Admin, en tant que Membre, en tant que Visiteur.
- Tenter d'aller sur la page de profil en étant seulement visiteur.
- Vous pouvez aussi tenter d'aller sur la page de connexion en étant déjà connecté en tant que membre.

Cela vous permettra de voir comment le site réagit.

**Erreur header :**

Si vous avez une erreur du type : "Cannot modify header information - headers already sent by ... , Cannot send session cookie - headers already sent by ...", il faudra faire attention à ne pas mettre d'echo ou d'espace avant d'effectuer les redirections header.

Généralement, si vous regardez bien ces erreurs, vous y verrez 2 lignes concernées, la ligne correspondant à l'appel de la fonction header qui ne peut pas se faire dans de bonnes conditions et la ligne correspondant à la partie de code qui pose problème.

Si vous m'avez suivi jusque-là, félicitations ! Notre espace membre est terminé !

*Bien-sûr nous pouvons améliorer cet espace membre, le sécuriser, etc. mais ça, ce sera pour plus tard...*

### **Le BackOffice - Gestion Boutique - Ajout de produits**

Le BackOffice d'un site web est une interface de gestion pour les réglages, réservée seulement à l'administrateur (ou aux administrateurs).

Nous avons prévu une partie (dans le dossier /admin/) réservée à l'administrateur pour la gestion de son site web (produits, commandes, membres).

Cette partie ne doit être accessible que par un internaute connecté ayant son statut fixé à 1 (autrement dit : un administrateur).

Imaginez si 1 visiteur ou si 1 membre arrivait à accéder à cette partie, il pourrait supprimer des produits, modifier des prix, etc. ça pourrait vite être un carnage !

Pour cela, nous allons utiliser la fonction `utilisateurEstConnecteEtEstAdmin` puisque nous avions prévu un code permettant de renvoyer TRUE (vrai tu est admin) ou FALSE (faux tu n'est pas admin).

Nous prévoirons également un formulaire d'ajout de produit avec récupération des données en POST et requête d'enregistrement (INSERT) dans la base de données :

```
admin/gestion_boutique.php

<?php
require_once("../inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- VERIFICATION ADMIN ---
if(!internauteEstConnecteEtEstAdmin())
{
    header("location:/connexion.php");
    exit();
}
//--- ENREGISTREMENT PRODUIT ---
if(empty($_POST))
{
    // debug($_POST);
    $photo_bdd = "";
    if(empty($_FILES['photo']['name']))
    {
        // debug($_FILES);
        $nom_photo = $_POST['reference'] . ' ' . $_FILES['photo']['name'];
        $photo_bdd = RACINE_SITE . "photo/$nom_photo";
        $photo_dossier = $_SERVER['DOCUMENT_ROOT'] . RACINE_SITE . "/photo/$nom_photo";
        copy($_FILES['photo']['tmp_name'],$photo_dossier);
    }
    foreach($_POST as $indice => $valeur)
    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("INSERT INTO produit (id_produit, reference, categorie, titre, description, couleur, taille, public, photo, prix, stock) values ('",
    "$_POST[reference]", "$_POST[categorie]", "$_POST[titre]", "$_POST[description]", "$_POST[couleur]", "$_POST[taille]", "$_POST[public]", '$photo_bdd',
    "$_POST[prix]", "$_POST[stock]");
    $contenu .= '<div class="validation">Le produit a été ajouté</div>';
}
//----- AFFICHAGE HTML -----
require_once("../inc/haut.inc.php");
echo $contenu;
?>


# Formulaire Produits </h1> <form method="post" enctype="multipart/form-data" action=""> <label for="reference">reference</label><br> <input type="text" id="reference" name="reference" placeholder="la référence de produit"> <br><br> <label for="categorie">categorie</label><br> <input type="text" id="categorie" name="categorie" placeholder="la categorie de produit"><br><br> <label for="titre">titre</label><br> <input type="text" id="titre" name="titre" placeholder="le titre du produit"> <br><br> <label for="description">description</label><br> <textarea name="description" id="description" placeholder="la description du produit"></textarea><br><br> <label for="couleur">couleur</label><br> <input type="text" id="couleur" name="couleur" placeholder="la couleur du produit"> <br><br> <label for="taille">Taille</label><br> <select name="taille"> <option value="S">S</option> <option value="M">M</option> <option value="L">L</option> <option value="XL">XL</option> </select><br><br> <label for="public">public</label><br> <input type="radio" name="public" value="m" checked>Homme <input type="radio" name="public" value="f">Femme<br><br> <label for="photo">photo</label><br> <input type="file" id="photo" name="photo"><br><br> <label for="prix">prix</label><br> <input type="text" id="prix" name="prix" placeholder="le prix du produit"><br><br> <label for="stock">stock</label><br> <input type="text" id="stock" name="stock" placeholder="le stock du produit"><br><br> <input type="submit" value="enregistrement du produit"> </form> <?php require_once("../inc/bas.inc.php"); ?>


```

## Résultat

**Formulaire Produits**reference  
11-d-23categorie  
tshirttitre  
Tshirt Col Vdescription  
Tee-shirt en coton  
flammé liseré  
contrastantcouleur  
bleuTaille  
M ▾public  
Homme Femmephoto  
Parcourir... bleu.jpgprix  
20stock  
53**enregistrement****Quelques Explications****Le contrôle admin**

Nous demandons dès les premières lignes `if(!internauteEstConnecteEtEstAdmin())` si l'internaute n'est pas admin, alors nous effectuons une redirection vers la page de connexion.

Nous en profitons pour mettre un `exit()` de manière à ce que l'interpréteur ne décode pas la suite du code et que la redirection se fasse immédiatement.

**Le formulaire d'ajout de produit**

La balise `<form>` comporte l'attribut `enctype` qui est indispensable pour permettre l'upload de fichier au sein de ce formulaire.

Les tailles de produit ont été mises en lettre, mais cela peut être changé.

**Le traitement PHP pour l'ajout de produit**

Nous prévoyons un traitement PHP si le contenu du POST n'est pas vide `if(!empty($_POST))` (autrement dit s'il a été rempli par un clic sur le bouton submit).

Nous initialisons une variable `$photo_bdd` à vide pour éviter plus tard une erreur `undefined` si aucune photo n'est ajoutée.

Un fichier transmis par upload ne se récupère pas avec `$_POST` mais avec `$_FILES` (il s'agit d'une autre superglobale).

Si une photo a été uploadée `if(!empty($_FILES['photo']['name']))`, nous changeons le nom de la photo `$nom_photo = $_POST['reference'] . '_' . $_FILES['photo']['name'];` car par défaut 2 photos du même nom s'écrasent et se remplacent.

Il est nécessaire que le dossier `/photo/` soit existant sur le serveur.

Il y a une sauvegarde du chemin de la photo dans la base de données, et une sauvegarde physique du fichier photo sur le serveur dans le dossier prévu à cet effet.

Dans l'idéal, il serait intéressant d'engager plusieurs traitements sur une photo uploadée :

- nom : donner 1 nom plus cohérent
- dimension : faire attention à ce qu'on ne vous envoie pas des photos miniatures ou au contraire une taille trop grande
- poids : des poids trop volumineux peuvent encombrer votre serveur
- extension : attention, on peut très bien vous envoyer des .exe ou des fichiers d'attaques) (même si effectivement l'administrateur n'est pas censé pirater son propre site)

- etc.

Faites des tests :

- Tenter d'accéder à cette page d'administration en tant que visiteur, membre ou admin.
- Ajouter des produits (avec photo de préférence) et vérifier les enregistrements de votre base (via PhpMyAdmin).

## Etape 7. Le BackOffice - Gestion Boutique - Affichage de produits

Maintenant que vous avez pu ajouter des produits, nous allons les afficher sur la page web

En effet, cela évitera d'aller dans PhpMyAdmin et surtout le commerçant pourra voir sa liste de produits en cliquant sur un lien prévu à cet effet.

### Affichage des Articles

Nombre de produit(s) dans la boutique : 8

id_produit	reference	categorie	titre	description	couleur	taille	public	photo	prix	stock	Modification	Supression
1	11-d-23	tshirt	Tshirt Col V	Tee-shirt en coton flammé liseré contrastant	bleu	M	m		20	53		
2	66-f-15	tshirt	Tshirt Col V rouge	c'est vraiment un super tshirt en soirée !	rouge	L	m		15	230		
3	88-g-77	tshirt	Tshirt Col rond vert	Il vous faut ce tshirt Made In France !!!	vert	L	m		29	63		
4	55-b-38	tshirt	Tshirt jaune	le jaune reviens à la mode, non? :-)	jaune	S	m		20	3		
5	31-p-33	tshirt	Tshirt noir original	voici un tshirt noir très original :p	noir	XL	m		25	80		
6	56-a-65	chemise	Chemise Blanche	Les chemises c'est bien mieux que les tshirts	blanc	L	m		49	73		
7	63-s-63	chemise	Chemise Noir	Comme vous pouvez le voir c'est une chemise noir...	noir	M	m		59	120		
8	77-p-79	pull	Pull gris	Pull gris pour l'hiver	gris	XL	m		79	99		

```

admin/gestion_boutique.php

<?php
require_once("../inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- VERIFICATION ADMIN ---
if(!l'internauteEstConnecteEtEstAdmin())
{
    header("location:../connexion.php");
    exit();
}
//--- ENREGISTREMENT PRODUIT ---
if(empty($_POST))
{
    // debug($_POST);
    $photo_bdd = "";
    if(empty($_FILES['photo']['name']))
    {
        // debug($_FILES);
        $nom_photo = $_POST['reference'] . ' ' . $_FILES['photo']['name'];
        $photo_bdd = RACINE_SITE . "photo/$nom_photo";
        $photo_dossier = $_SERVER['DOCUMENT_ROOT'] . RACINE_SITE . "/photo/$nom_photo";
        copy($_FILES['photo'][tmp_name],$photo_dossier);
    }
    foreach($_POST as $indice => $valeur)
    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("INSERT INTO produit (id_produit, reference, categorie, titre, description, couleur, taille, public, photo, prix, stock) values (".
        "$_POST[reference]", "$_POST[categorie]", "$_POST[titre]", "$_POST[description]", "$_POST[couleur]", "$_POST[taille]", "$_POST[public]", '$photo_bdd',
        "$_POST[prix]", "$_POST[stock]");
        $contenu .= '<div class="validation">Le produit a été ajouté</div>';
    }
//--- LIENS PRODUITS ---
$contenu .= '<a href="?action=affichage">Affichage des produits</a><br>';
$contenu .= '<a href="?action=ajout">Ajout d'un produit</a><br><br><hr><br>';
//--- AFFICHAGE PRODUITS ---
if(isset($_GET['action']) && $_GET['action'] == "affichage")
{
}

```

```

$résultat = executeRequete("SELECT * FROM produit");

$contenu .= '<h2> Affichage des Produits </h2>';
$contenu .= 'Nombre de produit(s) dans la boutique : ' . $résultat->num_rows;
$contenu .= '<table border="1"><tr>';

while($colonne = $résultat->fetch_field())
{
    $contenu .= '<th>' . $colonne->name . '</th>';
}
$contenu .= '<th>Modification</th>';
$contenu .= '<th>Suppression</th>';
$contenu .= '</tr>';

while ($ligne = $résultat->fetch_assoc())
{
    $contenu .= '<tr>';
    foreach ($ligne as $indice => $information)
    {
        if($indice == "photo")
        {
            $contenu .= '<td></td>';
        }
        else
        {
            $contenu .= '<td>' . $information . '</td>';
        }
    }
    $contenu .= '<td><a href="?action=modification&id_produit=' . $ligne['id_produit'] .'"></a></td>';
    $contenu .= '<td><a href="?action=suppression&id_produit=' . $ligne['id_produit'] .'" OnClick="return(confirm('En êtes vous certain ?'));"></a></td>';
    $contenu .= '</tr>';
}
$contenu .= '</table><br><hr><br>';

//----- AFFICHAGE HTML -----
require_once("../inc/haut.inc.php");
echo $contenu;
if(isset($_GET['action']) && ($_GET['action'] == 'ajout'))
{
    echo '
<h1> Formulaire Produits </h1>
<form method="post" enctype="multipart/form-data" action="">
    <label for="reference">référence</label><br>
    <input type="text" id="reference" name="reference" placeholder="la référence de produit"> <br><br>

    <label for="categorie">catégorie</label><br>
    <input type="text" id="categorie" name="categorie" placeholder="la catégorie de produit"><br><br>

    <label for="titre">titre</label><br>
    <input type="text" id="titre" name="titre" placeholder="le titre du produit"> <br><br>

    <label for="description">description</label><br>
    <textarea name="description" id="description" name="description" placeholder="la description du produit"></textarea><br><br>

    <label for="couleur">couleur</label><br>
    <input type="text" id="couleur" name="couleur" placeholder="la couleur du produit"> <br><br>

    <label for="taille">Taille</label><br>
    <select name="taille">
        <option value="S">S</option>
        <option value="M">M</option>
        <option value="L">L</option>
        <option value="XL">XL</option>
    </select><br><br>

    <label for="public">public</label><br>
    <input type="radio" name="public" value="m" checked>Homme
    <input type="radio" name="public" value="f">Femme<br><br>

    <label for="photo">photo</label><br>
    <input type="file" id="photo" name="photo"><br><br>

    <label for="prix">prix</label><br>
    <input type="text" id="prix" name="prix" placeholder="le prix du produit"><br><br>

    <label for="stock">stock</label><br>
    <input type="text" id="stock" name="stock" placeholder="le stock du produit"><br><br>

    <input type="submit" value="enregistrement du produit">
</form>';

require_once("../inc/bas.inc.php"); ?>

```

### Quelques Explications

Nous avons prévue 2 liens permettant soit d'afficher les produits, soit d'ajouter 1 produit.

Techniquement, cela permet de passer dans l'url ?action=affichage ou ?action=ajout et donc en récupérant l'information véhiculer dans l'url (via \$\_GET), le site peut détecter l'action à déclencher.

L'administrateur peut donc choisir l'action qu'il souhaite mettre en oeuvre : ajout ou affichage.

L'affichage des produits se fait dans 1 tableau (1 tableau), si cela n'est pas clair [vous pouvez reconstruire cette partie en suivant ce lien](#).

Nous ajoutons 2 liens (représentés d'images pour l'occasion) afin de proposer la modification et la suppression de produits pour l'administrateur (le commerçant).

Pour la modification et la suppression, nous passerons par l'action par l'url ainsi que l'id du produit correspondant.

L'action nous permettra de savoir que nous devons supprimer/modifier un produit et l'id nous permettra de savoir du quel il s'agit.

## Etape 7. Le BackOffice - Gestion Boutique - Suppression des produits

Le commerçant (administrateur) peut ajouter et observer ses produits mais pour lui offrir une gestion complète il est important de lui proposer les options de suppression et de modification de produit.

Commençons avec la suppression des produits :

```
admin/gestion_boutique.php

<?php
require_once("../inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- VERIFICATION ADMIN ---
if(!internauteEstConnecteEtEstAdmin())
{
    header("location:../connexion.php");
    exit();
}
//--- SUPPRESSION PRODUIT ---
if(isset($_GET['action']) && $_GET['action'] == "suppression")
{
    // $contenu .= $_GET['id_produit'];
    $resultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
    $produit_a_supprimer = $resultat->fetch_assoc();
    $chemin_photo_a_supprimer = $_SERVER['DOCUMENT_ROOT'] . $produit_a_supprimer['photo'];
    if(empty($produit_a_supprimer['photo']) && file_exists($chemin_photo_a_supprimer)) unlink($chemin_photo_a_supprimer);
    executeRequete("DELETE FROM produit WHERE id_produit=$_GET[id_produit]");
    $contenu .= '<div class="validation">Suppression du produit : ' . $_GET['id_produit'] . '</div>';
    $_GET['action'] = 'affichage';
}
//--- ENREGISTREMENT PRODUIT ---
if(empty($_POST))
{
    // debug($_POST);
    $photo_bdd = "";
    if(empty($_FILES['photo']['name']))
    {
        // debug($_FILES);
        $nom_photo = $_POST['reference'] . ' ' . $_FILES['photo']['name'];
        $photo_bdd = RACINE_SITE . "photo/$nom_photo";
        $photo_dossier = $_SERVER['DOCUMENT_ROOT'] . RACINE_SITE . "/photo/$nom_photo";
        copy($_FILES['photo'][tmp_name],$photo_dossier);
    }
    foreach($_POST as $indice => $valeur)
    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("INSERT INTO produit (id_produit, reference, categorie, titre, description, couleur, taille, public, photo, prix, stock) values ('", $_POST['reference'], "'$_POST[categorie]", "'$_POST[titre]", "'$_POST[description]", "'$_POST[couleur]", "'$_POST[taille]", "'$_POST[public]", "'$photo_bdd', '$_POST[prix]', '$_POST[stock]')");
    $contenu .= '<div class="validation">Le produit a été enregistré</div>';
}
//--- LIENS PRODUITS ---
$contenu .= '<a href="?action=affichage">Affichage des produits</a><br>';
$contenu .= '<a href="?action=ajout">Ajout d'un produit</a><br><br><hr><br>';
//--- AFFICHAGE PRODUITS ---
if(isset($_GET['action']) && $_GET['action'] == "affichage")
{
    $resultat = executeRequete("SELECT * FROM produit");

    $contenu .= '<h2> Affichage des Produits </h2>';
    $contenu .= 'Nombre de produit(s) dans la boutique : ' . $resultat->num_rows;
    $contenu .= '<table border="1"><tr>';

    while($colonne = $resultat->fetch_field())
    {
        $contenu .= '<th>' . $colonne->name . '</th>';
    }
    $contenu .= '<th>Modification</th>';
    $contenu .= '<th>Suppression</th>';
    $contenu .= '</tr>';

    while ($ligne = $resultat->fetch_assoc())
    {
        $contenu .= '<tr>';
        foreach ($ligne as $indice => $information)
        {
            if($indice == "photo")
            {
                $contenu .= '<td><img src="" . $information . "" =70" height="70"></td>';
            }
            else
            {
                $contenu .= '<td>' . $information . '</td>';
            }
        }
        $contenu .= '<td><a href="?action=modification&id_produit=' . $ligne['id_produit'] . '"></a></td>';
        $contenu .= '<td><a href="?action=suppression&id_produit=' . $ligne['id_produit'] . '" OnClick="return(confirm(\'En êtes vous certain ?\'))"><img'
    }
}
```

```

src="../inc/img/delete.png"></a></td>';
$contenu .= '</tr>';
}
$contenu .= '</table><br><hr><br>';
}

//----- AFFICHAGE HTML -----
require_once("../inc/haut.inc.php");
echo $contenu;
if(isset($_GET['action']) && ($_GET['action'] == 'ajout'))
{
    echo '
<h1> Formulaire Produits </h1>
<form method="post" enctype="multipart/form-data" action="">
    <label for="reference">reference</label><br>
    <input type="text" id="reference" name="reference" placeholder="la référence de produit"> <br><br>

    <label for="categorie">categorie</label><br>
    <input type="text" id="categorie" name="categorie" placeholder="la categorie de produit"><br><br>

    <label for="titre">titre</label><br>
    <input type="text" id="titre" name="titre" placeholder="le titre du produit"> <br><br>

    <label for="description">description</label><br>
    <textarea name="description" id="description" placeholder="la description du produit"></textarea><br><br>

    <label for="couleur">couleur</label><br>
    <input type="text" id="couleur" name="couleur" placeholder="la couleur du produit"> <br><br>

    <label for="taille">Taille</label><br>
    <select name="taille">
        <option value="S">S</option>
        <option value="M">M</option>
        <option value="L">L</option>
        <option value="XL">XL</option>
    </select><br><br>

    <label for="public">public</label><br>
    <input type="radio" name="public" value="m" checked>Homme
    <input type="radio" name="public" value="f">Femme<br><br>

    <label for="photo">photo</label><br>
    <input type="file" id="photo" name="photo"><br><br>

    <label for="prix">prix</label><br>
    <input type="text" id="prix" name="prix" placeholder="le prix du produit"><br><br>

    <label for="stock">stock</label><br>
    <input type="text" id="stock" name="stock" placeholder="le stock du produit"><br><br>

    <input type="submit" value="enregistrement du produit">
</form>';
}
require_once("../inc/bas.inc.php"); ?>

```

#### Quelques Explications

Nous détectons l'action suppression dans l'url `if(isset($_GET['action']) && $_GET['action'] == "suppression")`.

Nous allons d'abord sélectionner toutes les informations sur ce produit dans la base, avec notamment le chemin vers la photo afin de la supprimer de notre serveur (puisque nous ne gérons pas l'attribution d'une même image pour plusieurs produits).

Nous formulons une 2e requête pour supprimer réellement le produit de notre base et nous rebasculons vers l'action d'affichage pour observer tous les produits.

#### **Etape 7. Le BackOffice - Gestion Boutique - modification des produits**

Nous devons également proposer la modification de produits, pour cela nous aurons besoin d'un formulaire permettant d'accueillir les données en vue d'une modification.

Pour factoriser le code (factoriser = réduire le code, comme ici on ré-utilise du code pour plusieurs choses différentes), nous pouvons nous servir d'un seul formulaire pour l'ajout et la modification.

Et oui, qu'on ajoute ou que l'on modifie, il s'agit des mêmes champs.

Seule différence, lorsque l'on modifie il nous faudra les champs de formulaire pré-saisis avec les informations actuelles.

Voici une version modifiée :

admin/gestion_boutique.php	?
<pre> &lt;?php require_once("../inc/init.inc.php"); //----- TRAITEMENTS PHP ----- </pre>	

```

<!-- VÉRIFICATION ADMIN --//
if(!l'internauteEstConnectéEtEstAdmin())
{
    header("location:../connexion.php");
    exit();
}

<!-- SUPPRESSION PRODUIT --//
if(isset($_GET['action']) &amp;&amp; $_GET['action'] == "suppression")
{
    // $contenu .= $_GET['id_produit'];
    $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
    $produit_a_supprimer = $résultat-&gt;fetch_assoc();
    $chemin_photo_a_supprimer = $_SERVER[DOCUMENT_ROOT] . $produit_a_supprimer['photo'];
    if(empty($produit_a_supprimer['photo']) &amp;&amp; file_exists($chemin_photo_a_supprimer)) unlink($chemin_photo_a_supprimer);
    $contenu .= '&lt;div class="validation"&gt;Suppression du produit : ' . $_GET['id_produit'] . '&lt;/div&gt;';
    executeRequete("DELETE FROM produit WHERE id_produit=$_GET[id_produit]");
    $_GET['action'] = 'affichage';
}

<!-- ENREGISTREMENT PRODUIT --//
if(!empty($_POST))
{
    // debug($_POST);
    $photo_bdd = "";
    if(isset($_GET['action']) &amp;&amp; $_GET['action'] == 'modification')
    {
        $photo_bdd = $_POST['photo_actuelle'];
    }
    if(empty($_FILES['photo']['name']))
    {
        // debug($_FILES);
        $nom_photo = $_POST['reference'] . ' ' . $_FILES['photo']['name'];
        $photo_bdd = RACINE_SITE . "photo/$nom_photo";
        $photo_dossier = $_SERVER[DOCUMENT_ROOT] . RACINE_SITE . "/photo/$nom_photo";
        copy($_FILES['photo'][tmp_name],$photo_dossier);
    }
    foreach($_POST as $indice =&gt; $valeur)
    {
        $_POST[$indice] = htmlEntities(addSlashes($valeur));
    }
    executeRequete("REPLACE INTO produit (id_produit, reference, categorie, titre, description, couleur, taille, public, photo, prix, stock) values ('$_POST[id_produit]', '$_POST[reference]', '$_POST[categorie]', '$_POST[titre]', '$_POST[description]', '$_POST[couleur]', '$_POST[taille]', '$_POST[public]', '$photo_bdd', '$_POST[prix]', '$_POST[stock]')");
    $contenu .= '&lt;div class="validation"&gt;Le produit a été ajouté&lt;/div&gt;';
    $_GET['action'] = 'affichage';
}

<!-- LIENS PRODUITS --//
$contenu .= '&lt;a href="?action=affichage"&gt;Affichage des produits&lt;/a&gt;&lt;br&gt;';
$contenu .= '&lt;a href="?action=ajout"&gt;Ajout d'un produit&lt;/a&gt;&lt;br&gt;&lt;br&gt;&lt;br&gt;&lt;br&gt;';

<!-- AFFICHAGE PRODUITS --//
if(isset($_GET['action']) &amp;&amp; $_GET['action'] == "affichage")
{
    $résultat = executeRequete("SELECT * FROM produit");

    $contenu .= '&lt;h2&gt; Affichage des produits &lt;/h2&gt;';
    $contenu .= 'Nombre de produit(s) dans la boutique : ' . $résultat-&gt;num_rows;
    $contenu .= '&lt;table border="1" cellpadding="5"&gt;&lt;tr&gt;';

    while($colonne = $résultat-&gt;fetch_field())
    {
        $contenu .= '&lt;th&gt;' . $colonne-&gt;name . '&lt;/th&gt;';
    }
    $contenu .= '&lt;th&gt;Modification&lt;/th&gt;';
    $contenu .= '&lt;th&gt;Supression&lt;/th&gt;';
    $contenu .= '&lt;/tr&gt;';

    while ($ligne = $résultat-&gt;fetch_assoc())
    {
        $contenu .= '&lt;tr&gt;';
        foreach ($ligne as $indice =&gt; $information)
        {
            if($indice == "photo")
            {
                $contenu .= '&lt;td&gt;&lt;img src="" . $information . "" = "70" height="70"&gt;&lt;/td&gt;';
            }
            else
            {
                $contenu .= '&lt;td&gt;' . $information . '&lt;/td&gt;';
            }
        }
        $contenu .= '&lt;td&gt;&lt;a href="?action=modification&amp;id_produit=' . $ligne['id_produit'] . '"&gt;&lt;img src="../inc/img/edit.png"&gt;&lt;/a&gt;&lt;/td&gt;';
        $contenu .= '&lt;td&gt;&lt;a href="?action=suppression&amp;id_produit=' . $ligne['id_produit'] . '" OnClick="return(confirm('En êtes vous certain ?'));"&gt;&lt;img src="../inc/img/delete.png"&gt;&lt;/a&gt;&lt;/td&gt;';
        $contenu .= '&lt;/tr&gt;';
    }
    $contenu .= '&lt;/table&gt;&lt;br&gt;&lt;br&gt;&lt;br&gt;';
}

<!-- AFFICHAGE HTML -----
require_once("../inc/haut.inc.php");
echo $contenu;

if(isset($_GET['action']) &amp;&amp; ($_GET['action'] == 'ajout' || $_GET['action'] == 'modification'))
{
    if(isset($_GET['id_produit']))
    {
        $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_GET[id_produit]");
        $produit_actuel = $résultat-&gt;fetch_assoc();
    }
    echo '
&lt;h1&gt; Formulaire Produits &lt;/h1&gt;
&lt;form method="post" enctype="multipart/form-data" action=""&gt;

    &lt;input type="hidden" id="id_produit" name="id_produit" value="" . if(isset($produit_actuel['id_produit'])) echo $produit_actuel['id_produit']; echo "&gt;

    &lt;label for="reference"&gt;reference&lt;/label&gt;&lt;br&gt;
    &lt;input type="text" id="reference" name="reference" placeholder="la référence de produit" value="" . if(isset($produit_actuel['reference'])) echo $produit_actuel['reference']; echo "&gt;&lt;br&gt;&lt;br&gt;</pre>

```

```

<label for="categorie">catégorie</label><br>
<input type="text" id="categorie" name="categorie" placeholder="la catégorie de produit" value="" if(isset($produit_actuel['categorie'])) echo $produit_actuel['categorie']; echo "<br><br>

<label for="titre">titre</label><br>
<input type="text" id="titre" name="titre" placeholder="le titre du produit" value="" if(isset($produit_actuel['titre'])) echo $produit_actuel['titre']; echo "> <br><br>

<label for="description">description</label><br>
<textarea name="description" id="description" placeholder="la description du produit">; if(isset($produit_actuel['description'])) echo $produit_actuel['description']; echo '</textarea><br><br>

<label for="couleur">couleur</label><br>
<input type="text" id="couleur" name="couleur" placeholder="la couleur du produit" value="" if(isset($produit_actuel['couleur'])) echo $produit_actuel['couleur']; echo "> <br><br>

<label for="taille">Taille</label><br>
<select name="taille">
    <option value="S"; if(isset($produit_actuel) && $produit_actuel['taille'] == 'S') echo ' selected '; echo '>S</option>
    <option value="M"; if(isset($produit_actuel) && $produit_actuel['taille'] == 'M') echo ' selected '; echo '>M</option>
    <option value="L"; if(isset($produit_actuel) && $produit_actuel['taille'] == 'L') echo ' selected '; echo '>L</option>
    <option value="XL"; if(isset($produit_actuel) && $produit_actuel['taille'] == 'XL') echo ' selected '; echo '>XL</option>
</select><br><br>

<label for="public">public</label><br>
<input type="radio" name="public" value="m"; if(isset($produit_actuel) && $produit_actuel['public'] == 'm') echo ' checked ';
elseif(!isset($produit_actuel) && !isset($_POST['public'])) echo 'checked'; echo '>Homme
<input type="radio" name="public" value="f"; if(isset($produit_actuel) && $produit_actuel['public'] == 'f') echo ' checked '; echo '>Femme<br><br>

<label for="photo">photo</label><br>
<input type="file" id="photo" name="photo"><br><br>;
if(isset($produit_actuel))
{
    echo '<i>Vous pouvez uploader une nouvelle photo si vous souhaitez la changer</i><br>';
    echo <br>;
    echo '<input type="hidden" name="photo_actuelle" value="'.$produit_actuel['photo'].'"><br>;
}

echo '
<label for="prix">prix</label><br>
<input type="text" id="prix" name="prix" placeholder="le prix du produit" value="" if(isset($produit_actuel['prix'])) echo $produit_actuel['prix']; echo "><br><br>

<label for="stock">stock</label><br>
<input type="text" id="stock" name="stock" placeholder="le stock du produit" value="" if(isset($produit_actuel['stock'])) echo $produit_actuel['stock'];
echo "><br><br>

<input type="submit" value="" echo ucfirst($_GET['action']) . ' du produit">
</form>';
}
require_once("../inc/bas.inc.php");
?>
```

## Quelques Explications



### ➤ Attention

Plusieurs lignes ont changé, si vous ne récupérez pas l'intégralité du code, à vous d'être vigilant et ne rien oublier.

Nous demandons à ce que le formulaire produit s'affiche également en cas de modification.

Nous récupérons les informations actuelles d'un produit pour les disposer et les pré-saisir dans le formulaire afin que le commerçant puisse modifier seulement ce qu'il souhaite (sans pour autant perdre les autres informations).

Nous utilisons l'action se trouvant dans l'url (\$\_GET) pour définir le texte du bouton submit.

Nous créons un champ hidden (caché) pour transmettre l'id du produit devant être modifié, cela n'aura pas d'impact en cas d'ajout car ce champ n'enverra pas d'id (puisque le produit n'existera pas encore au moment de la création).

En ce qui concerne la photo, nous récupérons dans cet ordre : la photo actuelle du produit, et ensuite la photo qui aura été uploadée (si elle a été uploadé) pour écraser la photo actuelle.

Nous faisons évoluer la requête SQL afin que celle-ci devienne REPLACE.

- REPLACE se comporte en INSERT (ajout) s'il n'y a pas d'id\_produit connu
- REPLACE se comporte en UPDATE (modification) s'il y a 1 id\_produit connu (transmis par le champ id\_produit caché en hidden)

Autant dire que le code est assez diffus mais nous avons l'avantage d'utiliser 1 seul formulaire et 1 seule requête SQL pour 2 choses différentes (l'ajout et la modification).

## Résultat

Affichage des produits  
Ajout d'un produit

**Formulaire Produits**

reference

11-d-23

categorie

tshirt

titre

Tshirt Col V

description

Tee-shirt en coton  
flammé liseré  
contrastant

couleur

bleu

Taille

M

public

Homme Femme

photo

Parcourir... Aucun fichier sélectionné.

Vous pouvez uploader une nouvelle photo si vous souhaitez la changer



prix

20

stock

53

**Modification du produit****141****Etape 8. Le FrontOffice - La boutique**

Maintenant que nous avons des produits en boutique, il faut que l'on puisse les exposer aux internautes dans la partie FRONT.

Pour créer réellement notre boutique, nous allons travailler dans le fichier : **boutique.php**

```
boutique.php

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- AFFICHAGE DES CATEGORIES ---
$categories_des_produits = executeRequete("SELECT DISTINCT categorie FROM produit");
$contenu .= "<div class='boutique-gauche'>";
$contenu .= "<ul>";
while($cat = $categories_des_produits->fetch_assoc())
{
    $contenu .= "<li><a href=?categorie=" . $cat['categorie'] . ">" . $cat['categorie'] . "</a></li>";
}
$contenu .= "</ul>";
$contenu .= "</div>";
//--- AFFICHAGE DES PRODUITS ---
$contenu .= "<div class='boutique-droite'>";
if(isset($_GET['categorie']))
{
    $donnees = executeRequete("select id_produit,reference,titre,photo,prix from produit where categorie=?");
    while($produit = $donnees->fetch_assoc())
    {
        $contenu .= '<div class="boutique-produit">';
        $contenu .= "<h2>$produit[titre]</h2>";
        $contenu .= "<a href=fiche_produit.php?id_produit=$produit[id_produit]><img src=\"$produit[photo]\" width=130 height=100/></a>";
        $contenu .= "<p>$produit[prix] €</p>";
        $contenu .= '<a href=fiche_produit.php?id_produit=' . $produit['id_produit'] . ">Voir la fiche</a>";
        $contenu .= '</div>';
    }
}
```

```
$contenu .= '</div>';
//----- AFFICHAGE HTML -----
require_once("inc/haut.inc.php");
echo $contenu;
require_once("inc/bas.inc.php"); ?>
```

### Quelques Explications

Nous commençons par sélectionner les catégories de produits différents (DISTINCT) afin de les afficher dans une liste de liens.

Si l'internaute clique sur l'une des catégories que nous proposons, nous passerons l'information dans l'url, par exemple ?categorie=tshirt.

Lorsque l'url change, la page se recharge et le code va détecter qu'une information est passée dans l'url if(isset(\$\_GET['categorie']))

Nous ferons donc une 2e requête SQL consistant à récupérer tous les produits relatifs à une catégorie dans la base (par exemple : tous les tshirts).

Nous prévoyons également des liens sur les produits afin de proposer l'affichage d'un produit particulier dans une fiche produit dédiée.

### Résultat

The screenshot shows a website layout. At the top, there's a navigation bar with links: MONSITE.COM, Gestion des membres, Gestion des commandes, Gestion de la boutique, Voir votre profil, Accès à la boutique, Voir votre panier, and Se déconnecter. Below this, on the left, is a sidebar with a blue header containing the text "tshirt". Underneath, there are three categories: "chemise" and "pull", both of which are currently collapsed. The main content area displays four t-shirt products arranged in two rows of two. Each product is shown with its name, a small image of the shirt, its price, and a link labeled "Voir la fiche".

Produit	Prix	Action
Tshirt Col V	20 €	Voir la fiche
Tshirt Col V rouge	15 €	Voir la fiche
Tshirt Col rond vert	29 €	Voir la fiche
Tshirt noir original	25 €	Voir la fiche

Libre à vous d'adapter la présentation comme vous le souhaitez, voici le code CSS correspondant à notre résultat :

```
inc/css/style.css
```

```
*{ margin: 0; }
a{ text-decoration: none; color: #000; }
.conteneur{ margin: 0 auto; max: 1170px; }
/****** HAUT *****/
header { background: #000000; padding: 5px; text-align: center; }
header span{ color: #fff; font-weight: bold; text-transform: uppercase; margin-right: 5%; }
header nav{ display: inline; }
header a{ color: #fffff; text-decoration: none; padding: 5px; }
header nav a:hover{ background: #04ba6; }
/****** MILIEU *****/
section{ padding: 30px; min-height: 800px; }
/****** BAS *****/
footer{ background: #000; color: white; text-align: center; padding: 7px 0; }
/****** GENERAL *****/
.erreur{ background: #ff0000; padding: 5px; margin: 5px; }
.validation{ background: #669933; padding: 5px; margin: 5px; }
/****** BOUTIQUE *****/
.boutique-gauche{ float: left; : 30%; background: #f2f2f2; min-height: 500px; margin-right: 10%; padding-top: 10px; text-align: center; }
```

```

.boutique-gauche ul{ list-style: none; padding: 0; }
.boutique-gauche ul li a{ display: block; padding: 10px; }
.boutique-gauche ul li a:hover{ background: #04bafe; color: #fff; }
.boutique-droite{float: left; : 60%; }
.boutique-produit{ float: left; : 30%; text-align: center; padding: 5%; border-bottom: 1px solid #c0c0c0; }
.boutique-produit:hover{ background: #c0c0c0; }

```

Vous pouvez ajouter un produit en BackOffice sur la page gestion\_boutique, et vous le verrez automatiquement sur la page produit en FrontOffice. C'est ce qu'on appelle de l'affichage dynamique !

142

## Etape 9. Boutique - La fiche produit

Avec la fiche produit, nous allons créer une page qui permettra de visualiser plus en détail un produit.

```
fiche_boutique.php
```

```

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
if(isset($_GET['id_produit'])) { $rезультат = executeRequete("SELECT * FROM produit WHERE id_produit = '".$_GET['id_produit']."'"); }
if($rезультат->num_rows <= 0) { header("location:boutique.php"); exit(); }

$produit = $rезультат->fetch_assoc();
$contenu .= "<h2>Titre : $produit[titre]</h2><hr><br>";
$contenu .= "<p>Catégorie: $produit[categorie]</p>";
$contenu .= "<p>Couleur: $produit[couleur]</p>";
$contenu .= "<p>Taille: $produit[taille]</p>";
$contenu .= "<img src=\"$produit[photo]\" alt='Image du produit' style='width:150px; height:150px;'>";
$contenu .= "<p><i>Description: $produit[description]</i></p><br>";
$contenu .= "<p>Prix : $produit[prix]</p><br>";

if($produit['stock'] > 0)
{
    $contenu .= "<i>Nombre d'produit(s) disponible : $produit[stock] </i><br><br>";
    $contenu .= "<form method='post' action='panier.php'>";
    $contenu .= "<input type='hidden' name='id_produit' value='$produit[id_produit]'>";
    $contenu .= '<label for="quantite">Quantité : </label>';
    $contenu .= '<select id="quantite" name="quantite">';
    for($i = 1; $i <= $produit['stock'] && $i <= 5; $i++)
    {
        $contenu .= "<option>$i</option>";
    }
    $contenu .= '</select>';
    $contenu .= '<input type="submit" name="ajout_panier" value="ajout au panier">';
    $contenu .= '</form>';
}
else
{
    $contenu .= 'Rupture de stock !';
}
//----- AFFICHAGE HTML -----
require_once("inc/haut.inc.php");
echo $contenu;
require_once("inc/bas.inc.php");
?>
```

### Quelques Explications

Si l'id d'un produit se trouve dans l'url `if(isset($_GET['id_produit']))`, nous le récupérons via une requête SQL.

Si nous ne récupérons pas de produit dans la base `if($résultat->num_rows <= 0)`, cela indique que le produit n'existe pas ou plus, nous redirigeons l'internaute vers la page boutique.

Si l'interpréteur continue d'exécuter la suite des instructions, c'est que nous devons afficher le produit. Pour cela nous réalisons un traitement `$produit = $résultat->fetch_assoc();` afin de rendre le résultat exploitable sous forme de tableau ARRAY. Nous ne faisons pas de boucle dans la mesure où nous récupérons 1 seul enregistrement de la base de données.

Nous affichons les informations du produit, toujours par l'intermédiaire de la variable `$contenu`, afin que cela n'apparaisse pas avant les premières balises type doctype, html, body, etc.

Nous vérifions s'il y a du stock disponible afin de proposer la sélection d'une quantité et l'ajout au panier.

Si nous avons un stock disponible de 230 exemplaires pour un même produit, nous proposerons de prendre une quantité 5 par 5 (sauf si le stock est

inférieur à 5, nous proposerons le stock restant).

La boucle for intègre la variable \$stock et le 5 à chaque tour de boucle pour savoir si on ne le dépasse pas. cela prend donc le plus petit des deux nombres pour délimiter la boucle.

En cliquant sur le bouton "ajout au panier" l'internaute sera envoyé vers la page panier.php.

Si le produit n'a plus de stock disponible, nous informons l'internaute de la rupture.

Pour terminer, si l'internaute ne souhaite pas acheter ce produit, nous proposons un lien permettant de le relancer dans notre boutique directement dans la catégorie du produit courant

#### Résultat

MONSITE.COM Gestion des membres Gestion des commandes Gestion de la boutique Voir votre profil Accès à la boutique Voir votre panier Se déconnecter

**Titre : Tshirt Col V rouge**

Categorie: tshirt  
Couleur: rouge  
Taille: L



Description: c'est vraiment un super tshirt en soirée !

Prix : 15 €

Nombre produit(s) disponible : 230

Quantité :

[Retour vers la sélection de tshirt](#)

143

## Etape 10. Boutique - Le panier

### Etape 10. Boutique - Le panier - Ajout de produit

Maintenant que notre boutique existe, nous allons créer une fonctionnalité panier avec une simulation de paiement.

Techniquement, nous allons créer un panier dans notre fichier de session.

Nous gérons le panier dans un fichier de session afin que les informations soient gardées de page en page durant la navigation.

Nous ne gérons pas cela en base de données car d'une part ces informations ne sont pas destinées à rester très longtemps, et d'autre part, beaucoup de paniers sont abandonnés (il est donc préférable de ne pas polluer la base avec des données temporaires, ni alourdir le site web avec des requêtes SQL trop gourmandes).

Voici un schéma représentatif de nos tableaux ARRAY multidimensionnels de session :

\$_SESSION['panier']	
clé	valeur
titre	ARRAY
id_produit	ARRAY
quantite	ARRAY
prix	ARRAY

\$_SESSION['panier']['titre']	
clé	valeur
0	Tshirt Rouge Col V
1	Chemise Blanche
2	Pull gris

\$_SESSION['panier']['id_produit']	
clé	valeur
0	2
1	6
2	8

\$_SESSION['panier']['quantite']	
clé	valeur
0	1
1	1
2	3

\$_SESSION['panier']['prix']	
clé	valeur
0	15
1	49
2	79

De cette manière, nous savons que le produit en position 1 dans chaque sous tableau array correspond à une chemise blanc, il s'agit de l'id\_produit n°6, avec une quantité de 1 et un prix de 49€.

Avant de créer notre page panier.php, nous allons prévoir quelques fonctions afin de préparer au mieux cette fonctionnalité.

Voici la mise à jour du fichier inc/fonction.inc.php

```
inc/fonction.inc.php

function executeRequete($req)
{
    global $mysqli;
    $resultat = $mysqli->query($req);
    if(!$resultat)
    {
        die("Erreur sur la requete sql.<br>Message : " . $mysqli->error . "<br>Code: " . $req);
    }
    return $resultat;
}
//-----
function debug($var, $mode = 1)
{
    echo '<div style="background: orange; padding: 5px; float: right; clear: both; ">';
    $trace = debug_backtrace();
    $trace = array_shift($trace);
    echo 'Debug demandé dans le fichier : ' . $trace[file] . ' à la ligne ' . $trace[line];
    if($mode === 1)
    {
        print '<pre>'; print_r($var); print '</pre>';
    }
    else
    {
        print '<pre>'; var_dump($var); print '</pre>';
    }
    echo '</div>';
}
//-----
function internauteEstConnecte()
{
    if(!isset($_SESSION['membre'])) return false;
    else return true;
}
//-----
```

```

function internauteEstConnecteEtEstAdmin()
{
    if(internauteEstConnecte() && $_SESSION['membre']['statut'] == 1) return true;
    else return false;
}
//-----
function creationDuPanier()
{
    if(!isset($_SESSION['panier']))
    {
        $_SESSION['panier'] = array();
        $_SESSION['panier']['titre'] = array();
        $_SESSION['panier']['id_produit'] = array();
        $_SESSION['panier']['quantite'] = array();
        $_SESSION['panier']['prix'] = array();
    }
}
//-----
function ajouterProduitDansPanier($titre, $id_produit, $quantite, $prix)
{
    creationDuPanier();
    $position_produit = array_search($id_produit, $_SESSION['panier']['id_produit']);
    if($position_produit !== false)
    {
        $_SESSION['panier'][$quantite][$position_produit] += $quantite ;
    }
    else
    {
        $_SESSION['panier']['titre'][] = $titre;
        $_SESSION['panier']['id_produit'][] = $id_produit;
        $_SESSION['panier']['quantite'][] = $quantite;
        $_SESSION['panier']['prix'][] = $prix;
    }
}
//-----
function montantTotal()
{
    $total=0;
    for($i = 0; $i < count($_SESSION['panier']['id_produit']); $i++)
    {
        $total += $_SESSION['panier']['quantite'][$i] * $_SESSION['panier']['prix'][$i];
    }
    return round($total,2);
}
//-----
function retirerProduitDuPanier($id_produit_a_supprimer)
{
    $position_produit = array_search($id_produit_a_supprimer, $_SESSION['panier']['id_produit']);
    if ($position_produit !== false)
    {
        array_splice($_SESSION['panier']['titre'], $position_produit, 1);
        array_splice($_SESSION['panier']['id_produit'], $position_produit, 1);
        array_splice($_SESSION['panier']['quantite'], $position_produit, 1);
        array_splice($_SESSION['panier']['prix'], $position_produit, 1);
    }
}
//-----
```

### Quelques Explications

#### Creation du panier

Nous créons une première fonction `creationDuPanier()` afin de créer l'espace nécessaire dans le fichier de session pour accueillir les données des futurs produits.

Nous vérifierons si la session existe déjà, et seulement dans le cas où elle n'existe pas nous la crérons : `if(!isset($_SESSION['panier']))`

#### Ajout d'un produit dans le panier

Nous exécutons d'abord `creationDuPanier()` afin que l'espace permettant d'accueillir les données produits soit créé.

Notre fonction `ajouterProduitDansPanier()` est destinée à recevoir 4 arguments : le titre, l'id du produit, la quantité désirée et le prix (*ce sont les informations que nous garderons sur un produit dans notre panier*).

Avec la fonction préédéfinie `array_search`, nous allons d'abord procéder à une vérification. Avant de l'ajouter, est-ce que le produit est déjà présent dans le panier ?

Par exemple, cela pourrait arriver si l'internaute ajoute le produit 2 (tshirt rouge) dans le panier, qu'il repart de la page panier, qu'il rentre dans la fiche produit 2 et qu'il ajoute à nouveau un autre tshirt rouge (donc toujours le produit 2).

Pour éviter d'avoir 2 lignes de produits différentes dans notre panier (comme s'il s'agissait de 2 produits différents), nous mettrons à jour uniquement la quantité du produit en question (déjà présente dans le panier).

La fonction préédéfinie `array_search` est pratique car elle nous renvoie précisément la position du produit dans le tableau ARRAY et nous savons donc à quel endroit intervenir pour adapter la quantité.

Le symbole `+=` permet de ne pas perdre la quantité précédente. S'il y avait 1 en quantité, et que l'internaute ajoute 2 autres tshirt rouges (toujours le produit n°2) et bien nous ne voulons pas remplacer le chiffre 1 par 2 mais faire l'opération suivante :  $1 + 2 = 3$ .

Dans le cas contraire, si l'id du produit n'est pas connue dans le panier, nous l'ajouterons normalement dans notre fichier session.

#### Montant Total

La fonction `montantTotal()` va nous permettre de multiplier chaque prix par chaque quantité et donc de faire la somme des prix contenus dans le panier.

```
for($i = 0; $i < count($_SESSION['panier']['id_produit']); $i++) tant que $i est inférieur au nombre de produits contenus dans la session panier, nous allons parcourir les éléments.
```

```
$total += $_SESSION['panier']['quantite'][$i] * $_SESSION['panier']['prix'][$i]; on multiplie la quantité par le prix; ex 1*10€ ou 3*10€ sans remplacer pour autant la dernière valeur contenue dans la variable $total (+=). Addition et multiplication.
```

```
return round($total,2); nous retournons le prix total pour tous les produits (avec round nous demandons à avoir 2 chiffres après la virgule)
```

#### retirerProduitDuPanier

Cette fonction permet de retirer un produit du panier.

La fonction préédéfinie `Array_Search` retourne un chiffre afin de savoir à quel indice se trouve le produit à supprimer du panier. Lorsqu'un produit est retiré du panier, son indice dans les sous tableaux array titre, id\_produit, quantite et prix vont être vidés.

Pour éviter de garder des indices vides dans nos tableaux ARRAY, nous nous servons de la fonction préédéfinie `array_splice` pour faire glisser les produits des indices supérieurs vers des indices numériques du tableau inférieur.

Voici notre page panier qui exploitera notre fichier de fonction :

```
panier.php

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- AJOUT PANIER --//
if(isset($_POST['ajout_panier']))
{
    // debug($_POST);
    $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_POST[id_produit]");
    $produit = $résultat->fetch_assoc();
    ajouterProduitDansPanier($produit['titre'],$_POST['id_produit'],$_POST['quantite'],$produit['prix']);
}
//----- AFFICHAGE HTML -----
include("inc/haut.inc.php");
echo $contenu;
echo "<table border='1' style='border-collapse: collapse' cellpadding='7'>";
echo "<tr><td colspan='5'>Panier</td></tr>";
echo "<tr><th>Titre</th><th>Produit</th><th>Quantité</th><th>Prix Unitaire</th></tr>";
if(empty($_SESSION['panier'][id_produit])) // panier vide
{
    echo "<tr><td colspan='5'>Votre panier est vide</td></tr>";
}
else
{
    echo "<tr><td colspan='5'>Votre panier contient des produits</td></tr>";
}
echo "</table><br>";
echo "<i>Règlement par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS</i><br>";
echo "<hr>session panier:<br>", debug($_SESSION);
include("inc/bas.inc.php");
?>
```

#### Quelques Explications

Si le POST a été réalisé `if(isset($_POST['ajout_panier']))`, les informations proviennent de la page `fiche_produit.php`

Nous réalisons une requête SQL pour récupérer des informations sur le produit.

Nous nous servons de la fonction `ajouterProduitDansPanier` qui entraîne l'exécution de la fonction `creationDuPanier`.

Le prix et le titre viennent de la base de données par l'intermédiaire de la variable `$produit`, tandis que l'`id_produit` et la `quantite` proviennent de la `fiche_produit.php`.

Un peu plus bas dans le code, nous regardons si le produit est vide ou non et adressons un message à l'internaute.

Panier				
Titre	produit	Quantité	Prix Unitaire	Action
Votre panier contient des produits				

Réglément par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS

## Etape 10. Boutique - Le panier - Affichage des produits et enregistrement de la commande

Si nous avons réussi à rentrer des produits dans notre panier, nous allons maintenant les afficher afin que l'internaute puisse les observer sur la page web.

```
panier.php

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- AJOUT PANIER ---
if(isset($_POST['ajout_panier']))
{ // debug($_POST);
    $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_POST[id_produit]");
    $produit = $résultat->fetch_assoc();
    ajouterProduitDansPanier($produit['titre'],$_POST['id_produit'],$_POST['quantite'],$produit['prix']);
}
//----- AFFICHAGE HTML -----
include("inc/haut.inc.php");
echo $contenu;
echo "<table border='1' style='border-collapse: collapse' cellpadding='7'>";
echo "<tr><td colspan='5'>Panier</td></tr>";
echo "<tr><th>Titre</th><th>Produit</th><th>Quantité</th><th>Prix Unitaire</th></tr>";
if(empty($_SESSION['panier'][id_produit])) // panier vide
{
    echo "<tr><td colspan='5'>Votre panier est vide</td></tr>";
}
else
{
    for($i = 0; $i < count($_SESSION['panier'][id_produit]); $i++)
    {
        echo "<tr>";
        echo "<td>" . $_SESSION['panier'][titre][$i] . "</td>";
        echo "<td>" . $_SESSION['panier'][id_produit][$i] . "</td>";
        echo "<td>" . $_SESSION['panier'][quantite][$i] . "</td>";
        echo "<td>" . $_SESSION['panier'][prix][$i] . "</td>";
        echo "</tr>";
    }
    echo "<tr><th colspan='3'>Total</th><td colspan='2'>" . montantTotal() . " euros</td></tr>";
    if(internauteEstConnecte())
    {
        echo '<form method="post" action="">';
        echo '<tr><td colspan="5"><input type="submit" name="payer" value="Valider et déclarer le paiement"></td></tr>';
        echo '</form>';
    }
    else
    {
        echo '<tr><td colspan="3">Veuillez vous <a href="inscription.php">inscrire</a> ou vous <a href="connexion.php">connecter</a> afin de pouvoir payer</td></tr>';
    }
    echo "<tr><td colspan='5'><a href=?action=vider>Vider mon panier</a></td></tr>";
}
echo "</table><br>";
echo "<i>Règlement par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS</i><br>";
echo "<hr>session panier:<br>"; debug($_SESSION);
include("inc/bas.inc.php");
?>
```

### Quelques Explications

Nous faisons plusieurs tours avec la boucle for afin de parcourir tous les produits présents dans notre panier.

L'affichage se fait par l'intermédiaire de la superglobale \$\_SESSION (puisque toutes les informations sont enregistrées dans le fichier de session).

Faites des tests et remplissez votre panier avec plusieurs produits et plusieurs quantités !

Panier							
Titre	Produit	Quantité	Prix Unitaire	Action			
Tshirt Col V rouge	2	2	15	retirer			
Chemise Blanche	6	3	49	retirer			
Pull gris	8	5	79	retirer			
<b>Total</b>		572 euros					
<b>Valider et déclarer le paiement</b>							
<b>Vider mon panier</b>							

Règlement par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS

Nous prévoyons un formulaire permettant de déclarer le paiement avec un bouton submit "Valider et déclarer le paiement".

1 autre lien est disponible pour vider l'ensemble du panier.

Voici les traitements associés :

```
panier.php

<?php
require_once("inc/init.inc.php");
//----- TRAITEMENTS PHP -----
//--- AJOUT PANIER ---
if(isset($_POST['ajout_panier']))
{ // debug($_POST);
    $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=$_POST[id_produit]");
    $produit = $résultat->fetch_assoc();
    ajouterProduitDansPanier($produit['titre'],$_POST['id_produit'],$_POST['quantite'],$produit['prix']);
}
//--- VIDER PANIER ---
if(isset($_GET['action']) && $_GET['action'] == "vider")
{
    unset($_SESSION['panier']);
}
//--- PAIEMENT ---
if(isset($_POST['payer']))
{
    for($i=0 ;$i < count($_SESSION['panier']['id_produit']) ; $i++)
    {
        $résultat = executeRequete("SELECT * FROM produit WHERE id_produit=" . $_SESSION['panier']['id_produit'][$i]);
        $produit = $résultat->fetch_assoc();
        if($produit['stock'] < $_SESSION['panier'][quantite][$i])
        {
            $contenu .= '<hr><div class="erreur">Stock Restant: ' . $produit['stock'] . '</div>';
            $contenu .= '<div class="erreur">Quantité demandée: ' . $_SESSION['panier'][quantite][$i] . '</div>';
            if($produit['stock'] > 0)
            {
                $contenu .= '<div class="erreur">la quantité de l\'produit ' . $_SESSION['panier'][id_produit][$i] . ' à été réduite car notre stock était insuffisant, veuillez vérifier vos achats.</div>';
                $_SESSION['panier'][quantite][$i] = $produit['stock'];
            }
            else
            {
                $contenu .= '<div class="erreur">l\'produit ' . $_SESSION['panier'][id_produit][$i] . ' à été retiré de votre panier car nous sommes en rupture de stock, veuillez vérifier vos achats.</div>';
                retirerProduitDuPanier($_SESSION['panier'][id_produit][$i]);
                $i--;
            }
            $erreure = true;
        }
    }
    if(!isset($erreure))
    {
        executeRequete("INSERT INTO commande (id_membre, montant, date_enregistrement) VALUES (" . $_SESSION['membre'][id_membre] . "," .
montantTotal() . ", NOW())");
        $id_commande = $mysqli->insert_id;
        for($i = 0; $i < count($_SESSION['panier']['id_produit']); $i++)
        {
            executeRequete("INSERT INTO details_commande (id_commande, id_produit, quantite, prix) VALUES ($id_commande, " .
$_SESSION['panier'][id_produit][$i] . "," . $_SESSION['panier'][quantite][$i] . "," . $_SESSION['panier'][prix][$i] . ")");
        }
        unset($_SESSION['panier']);
        mail($_SESSION['membre'][email], "confirmation de la commande", "Merci votre n° de suivi est le $id_commande",
"From:vendeur@dp_site.com");
        $contenu .= "<div class='validation'>Merci pour votre commande. votre n° de suivi est le $id_commande</div>";
    }
}
//----- AFFICHAGE HTML -----
include("inc/haut.inc.php");
```

```

echo $contenu;
echo "<table border='1' style='border-collapse: collapse' cellpadding='7'>";
echo "<tr><td colspan='5'>Panier</td></tr>";
echo "<tr><th>Titre</th><th>Produit</th><th>Quantité</th><th>Prix Unitaire</th></tr>";
if(empty($_SESSION['panier'][id_produit])) // panier vide
{
    echo "<tr><td colspan='5'>Votre panier est vide</td></tr>";
}
else
{
    for($i = 0; $i < count($_SESSION['panier'][id_produit]); $i++)
    {
        echo "<tr>";
        echo "<td>" . $_SESSION['panier']['titre'][$i] . "</td>";
        echo "<td>" . $_SESSION['panier']['id_produit'][$i] . "</td>";
        echo "<td>" . $_SESSION['panier']['quantite'][$i] . "</td>";
        echo "<td>" . $_SESSION['panier']['prix'][$i] . "</td>";
        echo "</tr>";
    }
    echo "<tr><th colspan='3'>Total</th><td colspan='2'>" . montantTotal() . " euros</td></tr>";
    if(internauteEstConnecte())
    {
        echo '<form method="post" action="">';
        echo '<tr><td colspan="5"><input type="submit" name="payer" value="Valider et déclarer le paiement"></td></tr>';
        echo '</form>';
    }
    else
    {
        echo '<tr><td colspan="3">Veuillez vous <a href="inscription.php">inscrire</a> ou vous <a href="connexion.php">connecter</a> afin de pouvoir payer</td></tr>';
    }
    echo "<tr><td colspan='5'><a href=?action=vider>Vider mon panier</a></td></tr>";
}
echo "</table><br>";
echo "<i>Règlement par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS</i><br>";
// echo "<hr>session panier:<br>"; debug($_SESSION);
include("inc/bas.inc.php");
?>

```

## Résultat

MONSITE.COM

Voir votre profil Accès à la boutique Voir votre panier Se déconnecter

Merci pour votre commande, votre n° de suivi est le 1

Panier				
Titre	Produit	Quantité	Prix Unitaire	Action
Votre panier est vide				

Règlement par CHÈQUE uniquement à l'adresse suivante : 300 rue de vaugirard 75015 PARIS

## Quelques Explications

Pour tous les produits dans le panier, nous allons observer si les quantités demandées sont encore en stock.

Pour cela, nous devons formuler une requête SQL qui va chercher en base les informations sur 1 produit, nous devons faire ça pour tous les produits c'est la raison pour laquelle la requête SQL se trouve dans une boucle afin d'entamer une répétition sur le traitement.

Logiquement cela devrait être le cas mais imaginez la situation suivante :

Le produit n° 2 tshirt rouge est présent 2 fois en stock.

Julien arrive à 10h07 pour en prendre 2 dans son panier.

Olivier arrive à 10h08 pour en prendre 1 dans son panier.

A 10h10, Olivier valide son panier et son achat du tshirt rouge.

A 10h20 Julien souhaite valider l'achat de ses 2 tshirts mais il n'en reste plus assez puisque le site web en aura vendu 1 à Olivier quelques minutes avant. Par conséquent Julien pourra acheter qu'un seul tshirt rouge.

Comme dans un magasin physique, le produit ne vous appartient pas au moment où vous le mettez dans votre panier mais bien quand vous avez payé en caisse. C'est bien au moment du paiement qu'il faut vérifier la disponibilité du produit (et non pas au moment de l'ajout dans le panier)

Voici 3 situations qui peuvent arriver :

- Si la quantité demandée est couverte par le stock, il n'y a pas de soucis.

- Si la quantité demandée est inférieure au stock mais qu'il reste du stock, nous diminuons la quantité demandée par le stock restant (afin de vendre les derniers exemplaires).

l'internaute demande 6 tshirts, il en reste 2. Si le stock est supérieur à zéro, on lui remplace le 6 par un 2.

- Si la quantité demandée est inférieure au stock et qu'il n'y a plus de stock, nous retirons totalement le produit du panier.  
sinon, nous sommes en rupture de stock, on lui retire carrément le produit. On ne va pas remplacer son 6 par un 0

Nous aurions pu faire une boucle à l'envers au niveau du code `$i--;`. Nous avons choisi d'ajouter un retour dans la boucle, voici l'explication : si le produit en indice zéro pose problème pour cause de rupture de stock, il est retiré.

Du coup le produit en indice 1 passe en indice 0 (réordonner via array splice de la fonction `retirerproduitDuPanier()`) et ne sera pas vérifié (car le tour de boucle 0 sera passé, nous serons à 1).

Il faut donc que je reparsous l'indice 0 pour me préoccuper du produit qui vient de se décaler à l'indice 0 et qui n'est pas encore testé.  
Ceci explique la présence du code `$i--;`.

Quoi qu'il en soit (stock réduit ou rupture totale de stock), s'il y a eu un souci dans le traitement, nous mettrons la variable \$erreur à TRUE `$erreur = true;` N'hésitez pas à faire des tests, vous prenez le navigateur firefox en tant que membre et trouvez un produit avec seulement 3 exemplaires en stock. Vous ajoutez les 3 derniers exemplaires dans votre panier.

Dans le même temps, vous prenez le navigateur chrome en tant qu'admin (pour simuler la présence de l'administrateur sur un autre ordinateur), modifiez le stock de ce même produit à 1 (car on imagine que vous venez d'en vendre 2).

Revenez sur Firefox en tant que membre et tentez de valider votre panier pour voir si cela vous bloque et vous informe bien du stock réduit.

Pour traiter et enregistrer la commande, la variable \$erreur ne doit pas exister `if(!isset($erreur))` (sinon cela voudrait dire que nous sommes passés dans le stock réduit ou dans la rupture de stock totale).

Nous enregistrons la commande dans la table commande : l'id du membre est récupéré dans le fichier de session (membre actuellement connecté), le prix est calculé par notre fonction `montantTotal()`, la date d'enregistrement est générée via la fonction SQL `NOW()`

Pour une commande il peut y avoir plusieurs produits et donc plusieurs lignes d'informations (dans le cas où il s'agit d'une commande avec plusieurs produits différents).

Nous récupérons donc l'id de la commande (qui a été générée en `AUTO_INCREMENT`) : `$id_commande = $mysqli->insert_id;`

Il y a 1 seule commande mais plusieurs détails de commande à enregistrer, nous refaisons donc une boucle pour tous les enregistrer en les reliant au numéro de commande principale.

Nous pourrions réalisé la décrémentation du stock mais le paiement se déroule par cheque alors nous attendrons que l'administrateur reçoive ce cheque et décrémente lui même son stock via le BackOffice.

Une fois que l'internaute à déclaré le paiement, nous vidons son panier car tous les produits sont censés être payés !

Nous envoyons un email de confirmation au membre qui vient de réaliser la commande.

Au sein de ce fichier, nous mélangeons à la fois des instructions d'affichage `echo` et la variable `$contenu`.

La variable `$contenu` correspond à des affichages que nous devons faire lors des traitements mais que l'on doit retarder pour garder une structure viable.

## Résultat

<input type="checkbox"/> <input type="button" value="T"/> <input type="button" value="F"/>	<code>id_commande</code>	<code>id_membre</code>	<code>montant</code>	<code>date_enregistrement</code>	<code>etat</code>
<input type="checkbox"/> <input type="button" value="P"/> <input type="button" value="X"/>	1	4	301	2015-07-10 14:44:46	en cours de traitement

<input type="checkbox"/> <input type="button" value="T"/> <input type="button" value="F"/>	<code>id_details_commande</code>	<code>id_commande</code>	<code>id_produit</code>	<code>quantite</code>	<code>prix</code>
<input type="checkbox"/> <input type="button" value="P"/> <input type="button" value="X"/>	1	1	2	1	15
<input type="checkbox"/> <input type="button" value="P"/> <input type="button" value="X"/>	2	1	6	1	49
<input type="checkbox"/> <input type="button" value="P"/> <input type="button" value="X"/>	3	1	8	3	79

il y a bien 1 commande principale et 3 produits rattachés à cette commande.

Si vous en êtes arrivés jusque-là, je vous dis BRAVO !

Cette première version représente une base ré-utilisable et surtout un rassemblement de tous les sujets que nous avons pu voir durant le cours PHP, dans

un contexte précis de site web.

Bien entendu cela doit être amélioré, testé, sécurisé et continué d'être développé. Ca prend du temps de créer 1 site web complet !

## Entrainement - Développer davantage votre site et vos connaissances

Pour d'avantage développer notre site, je vous propose plusieurs suggestions d'exercices pour être plus performant :

144

## Les développements complémentaires pour faire évoluer le site

### Exercice 1 - BackOffice / Gestion des commandes

Ajoutez (dans le dossier `/admin/`) une page que vous nommerez « `gestion_commandes.php` » dans le dossier « `/admin` » afin d'afficher toutes les commandes passées sur le site ainsi que le détail des commandes.

L'affichage de toutes les informations aideront le commerçant :

- Le numéro de commande (`id_commande`) ainsi que la date et le montant afin que le commerçant puisse faire le rapprochement avec les chèques qu'il reçoit.
- Les informations sur les articles commandés (`id_article`) ainsi que le titre, la photo, la quantité demandée, etc. afin qu'il puisse honorer la commande du client.
- Le numéro du membre (`id_membre`) ainsi que son pseudo, adresse, ville et cp afin que le commerçant puisse envoyer le colis.

L'état de la commande doit s'afficher et doit pouvoir être modifiable par l'administrateur (s'il souhaite changer une commande jusqu'à présent « en cours de traitement » par « en cours d'envoi » par exemple).

L'idéal serait que le membre reçoive un email de notification pour l'informer de l'évolution de sa commande.

L'affichage du chiffre d'affaires doit également apparaître sur cette page.

De l'accessibilité et de la navigation seraient un plus pour le commerçant, en effet il serait bon de lui offrir la possibilité de trier les commandes par date, état, montant, etc.

Seul l'administrateur doit avoir accès à cette page.

### Exercice 2 - BackOffice / Gestion des membres

Ajoutez une page que vous nommerez « `gestion_membres.php` » dans le dossier « `/admin` » afin d'afficher tous les membres inscrits sur le site. Seul l'administrateur doit avoir accès à cette page.

Dans cette même page, ajoutez la possibilité à l'administrateur de pouvoir supprimer un membre inscrit au site.

Donnez-lui la possibilité d'ajouter d'autres comptes « administrateur ».

### Exercice 3 - FrontOffice / Modification du compte membre

Ajoutez un lien « mettre à jour mes informations » dans l'espace « `profil.php` » afin que les personnes inscrites au site aient la possibilité de modifier leurs informations relatives à leurs comptes tels que leur adresse, mot de passe, ville, cp, adresse, etc.

Cela implique la création d'une nouvelle page « `membres.php` ».

### Exercice 4 - FrontOffice / Suppression du compte membre

Ajoutez un lien « se désinscrire » dans l'espace « `profil.php` » afin que les personnes inscrites au site aient la possibilité de supprimer leur compte membre.

### Exercice 5 - FrontOffice / Amélioration de l'espace membre

Donner la possibilité aux membres d'avoir des avatars de profil.

Permettre à l'internaute de solliciter le site en cas de mot de passe perdu.

Sur la page « profil.php », affichez le suivi des commandes en cours, l'historique des commandes passées, etc.

Permettre à l'internaute de pouvoir se connecter avec son pseudo ou son adresse email.

### Exercice 6 - FrontOffice / Boutique

Sur la page « boutique.php » :

Effectuer un moteur de recherche par mots-clés avec une suggestion de saisie.

Effectuer une zone de recherche permettant de trier par tranche de prix.

Pour la sélection des produits, prévoyez une pagination s'il y en a beaucoup afin de les afficher par tranche de 10 produits.

### Exercice 7 - Suggestion de produits

Sur la page « fiche\_produit.php », affichez une suggestion de produits similaires au produit qui est sélectionné par l'internaute afin de faciliter les recherches croisées.

Sur la page « profil.php », affichez une suggestion de produits similaires aux produits que l'internaute a déjà achetés.

Dans le cas où il n'a pas encore acheté sur notre site, affichez une suggestion de produits correspondants à la dernière catégorie de produits qu'il a consultés sur notre boutique.

### Exercice 8 - Amélioration du panier

Sur la page « panier.php », affichez le prix HT et TTC unitaire et aussi multiplié par le nombre de produits.

Affichez le titre de du produit, ainsi que la photo à coté des autres informations déjà présentes.

Donner la possibilité à l'internaute d'augmenter ou réduire les quantités directement sur la page panier.

Nous allons améliorer notre système permettant de retirer ou baisser les articles si les quantités demandées par notre panier sont supérieures à notre stock restant en base de données.

Pour cela, il va falloir modifier la page « fiche\_produit.php ».

Actuellement, s'il reste 3 jeans, l'internaute peut prendre les 3 derniers et les mettre dans son panier.

En revanche, il peut revenir sur la fiche détaillée de cet article et les reprendre 3 fois autant de fois qu'il le souhaite.

Nous allons faire en sorte que si l'internaute ait déjà cet article 1 fois dans son panier, le site lui propose d'en prendre 2 seulement. Si l'internaute en possède déjà 2 dans son panier, le site lui propose d'en prendre 1 supplémentaire seulement.

Dans le panier, il y aura moins de risques pour l'internaute de demander des quantités supérieures au stock restant en base de données.

### Exercice 9 - Amélioration du système de commande

Intégrez la solution paypal sandbox pour réaliser des tests de paiement. Ce qui permettra d'aller plus loin qu'une simple simulation de paiement par chèque

Prevoyez de générer des factures PDF avec les produits commandés par l'internaute.

### Exercice 10 - Amélioration du graphisme

Afin d'avoir un meilleur retour des internautes, nous allons améliorer le visuel :

Intégrer un design dans le contexte du site.

Ajoutez de l'animation et des effets pour éviter d'avoir un site figé où rien ne bouge.

## Evaluation

Pour cet exercice, nous allons créer une calculatrice en PHP, c'est un classique qui a le mérite d'être formateur dans les débuts !

Créez une page « calculatrice.php ».

La page calculatrice est un formulaire avec un menu déroulant qui nous permet de choisir le signe de l'opération (addition, soustraction, multiplication, division).

**Exemple :**

146

## Exercice 2 : Création d'un Répertoire

**Exercice 2.1** Créez une base de données que vous appellerez « repertoire ».

A l'intérieur de celle-ci, vous créerez une table que vous appellerez « annuaire » avec les champs suivant :

- id\_annuaire (INT, 3, AI - PK)
- nom (VARCHAR, 30)
- prenom (VARCHAR, 30)
- telephone (INT, 10, zerofill)
- profession (VARCHAR, 30)
- ville (VARCHAR, 30)
- codepostal (INT, 5, zerofill)
- adresse (VARCHAR, 30)
- date\_de\_naissance (DATE)
- sexe (ENUM, 'm','f')
- description (TEXT)

**Exercice 2.2** Créez une page « formulaire.php » qui aura une apparence similaire à l'image ci-dessous.

Informations

Nom \*

Prénom \*

Téléphone \*

Profession

Ville

Code Postal

Adresse

Date de Naissance  
28/06/1980

Sexe  
homme:  femme:

Description

- Afficher le récapitulatif des saisies au dessus du formulaire (sur la même page).

**Exercice 2.3** Une fois les valeurs récupérées du formulaire, il faudra développer le code permettant l'insertion des saisies dans la table « annuaire » de la base de données « repertoire ».

Chaque validation du formulaire doit ajouter une nouvelle ligne d'enregistrement dans la table « annuaire ».

id_annuaire	nom	prenom	telephone	profession	ville	codepostal	adresse	date_de_naissance	sexe	description
1	cottet	julien	0185997733	informatique	paris	75011	dans la rue	1980-12-03	m	me voilà!

**Exercice 2.3** Une fois les valeurs récupérées du formulaire, il faudra développer le code permettant l'insertion des saisies dans la table « annuaire » de la base de données « repertoire ».

Chaque validation du formulaire doit ajouter une nouvelle ligne d'enregistrement dans la table « annuaire ».

id_annuaire	nom	prenom	telephone	profession	ville	codepostal	adresse	date_de_naissance	sexe	description
1	cottet	julien	0185997733	informatique	paris	75011	dans la rue	1980-12-03	m	me voilà!

**Exercice 2.4** Créez une page « affichage\_annuaire.php » qui permettra de récupérer les données et ainsi afficher le nom des champs suivis des informations contenues à l'intérieur de la table « annuaire ».

## Liste des entrées dans l'annuaire

id_annuaire	nom	prenom	telephone	profession	ville	codepostal	adresse	date_de_naissance	sexe	description
1	cottet	julien	0185997733	informatique	paris	75011	dans la rue	1980-12-03	m	me voilà!

**Exercice 2.5,** Sur la page « affichage\_annuaire.php », Ajouter 2 informations :

- Le nombre d'hommes
- Le nombre de femmes

## Liste des entrées dans l'annuaire

id_annuaire	nom	prenom	telephone	profession	ville	codepostal	adresse	date_de_naissance	sexe	description
1	cottet	julien	0185997733	informatique	paris	75011	dans la rue	1980-12-03	m	me voilà!

il y a 1 homme(s) et 0 femme(s)

**Exercice 2.6, MODIFICATION ET SUPPRESSION** Sur la page « affichage\_annuaire.php » :

- Donnez la possibilité de modifier les enregistrements (ouvrant un formulaire pour effectuer les modifications)
- Donnez la possibilité de supprimer les enregistrements (avec un message demandant une confirmation).

Ces deux actions doivent être possibles directement via la page web.

**Exercice 3.1** Créez une base de données que vous appellerez « bibliotheque ».

A l'intérieur de celle-ci, vous créerez trois tables que vous appellerez « abonne », « emprunt » et « livre » avec les champs suivants :

Table : abonne

Champs :

- id\_abonne (PK – AI – INT(3))
- prenom (VARCHAR(25))

Table : emprunt

Champs :

- id\_emprunt (PK – AI – INT(3))
- id\_livre (FK - INT(3))
- id\_abonne (FK - INT(3))
- date\_sortie (DATE)
- date\_rendu (DATE – DEFAULT NULL)

Table : livre

Champs :

- id\_livre(PK – AI – INT(3))
- auteur (VARCHAR(25))
- titre (VARCHAR(50))

Voici le contenu des tables à insérer :

Table abonne :

id_abonne	prenom
1	Guillaume
2	Benoit
3	Chloe
4	Laura

Table livre :

id_livre	auteur	titre
100	GUY DE MAUPASSANT	Une vie
101	GUY DE MAUPASSANT	Bel-Ami
102	HONORE DE BALZAC	Le pere Goriot
103	ALPHONSE DAUDET	Le Petit chose
104	ALEXANDRE DUMAS	La Reine Margot
105	ALEXANDRE DUMAS	Les Trois Mousquetaires

Table emprunt :

id_emprunt	id_livre	id_abonne	date_sortie	date_rendu
1	100	1	2011-12-17	2011-12-18
2	101	2	2011-12-18	2011-12-20
3	100	3	2011-12-19	2011-12-22
4	103	4	2011-12-19	2011-12-22
5	104	1	2011-12-19	2011-12-28
6	105	2	2012-03-20	2012-03-26
7	105	3	2013-06-13	NULL
8	100	2	2013-06-15	NULL

Exercice 3.2 - Créez une page « ajout\_abonne.php » qui aura une apparence similaire à l'image ci-dessous.

- Réaliser des contrôles de saisies.
- Afficher le récapitulatif des saisies au dessus du formulaire (sur la même page).

Ajout d'un abonné

Prenom

- Créez une page « ajout\_livre.php » qui aura une apparence similaire à l'image ci-dessous.
- Réalisez des contrôles de saisies.
- Affichez le récapitulatif des saisies au dessus du formulaire (sur la même page).

Ajout d'un livre

Auteur	<input type="text"/>
Titre	<input type="text"/>
<input type="button" value="Enregistrement"/>	

- Créez une page « ajout\_emprunt.php » qui aura une apparence similaire à l'image de gauche.
- Réalisez des contrôles de saisies.
- Affichez le récapitulatif des saisies au dessus du formulaire (sur la même page).

Ajout d'un emprunt

Abonné	<input type="text" value="1 - Guillaume"/>
Livre	<input type="text" value="100 - GUY DE MAUPASSANT   Une vie"/>
Date Sortie	<input type="text" value="23/07/2013"/>
Date Rendu	<input type="text" value="jj/mm/aaaa"/>
<input type="button" value="Enregistrement"/>	

#### Exercice 3.3 ENREGISTREMENT DES DONNEES

Une fois les valeurs récupérées des formulaires, il faudra développer le code permettant l'insertion des saisies dans les tables « abonne », « livre » et « emprunt » de la base de données « bibliotheque ».

Chaque validation des formulaires doit ajouter une nouvelle ligne d'enregistrement dans les tables concernées.

#### Exercice 3.4 AFFICHAGE DES DONNEES

Créez une page « affichage.php » qui permettra de récupérer les données et ainsi afficher le nom des champs suivis des informations contenues à l'intérieur des table « abonne », « livre » et « emprunt ».

Les affichages des différentes tables peuvent être gérés via des onglets de navigation.

#### Exercice 3.5 MODIFICATION ET SUPPRESSION

Sur la page « affichage.php » :

- o Donnez la possibilité de modifier les enregistrements (ouvrant un formulaire pour effectuer les modifications)
- o Donnez la possibilité de supprimer les enregistrements (avec un message demandant une confirmation).

Ces deux actions doivent être possibles directement via la page web pour toutes les données (abonne, livre et emprunt).

#### Exercice 3.6 AFFICHAGE DIVERS

- o Afficher le nombre d'abonnés.
- o Afficher le nombre de livres.
- o Afficher le nombre d'emprunts.
- o Afficher les numéros et titres des livres n'ayant pas été rendus à la bibliothèque
- o Afficher le n° de(s) livre(s) que Chloé a emprunté à la bibliothèque
- o Afficher la liste des abonnés ayant déjà emprunté un livre d'Alphonse DAUDET
- o Afficher les titres des livres que Chloé n'a pas encore rendus à la bibliothèque.
- o Afficher les titres des livres que Chloé n'a pas encore empruntés.
- o Afficher le prénom de l'abonné ayant emprunté le plus de livres
- o Afficher le nombre de livre emprunté par Guillaume
- o Afficher la liste des abonnés ayant emprunté le livre « Une Vie » sur l'année 2011
- o Afficher le nombre de livres empruntés par chaque abonné
- o Afficher la liste des abonnés avec les titres des livres qu'ils ont empruntés ainsi que la date de l'emprunt

Voici le cahier des charges :

### Présentation

LOKISALLE est une société imaginaire. Les informations concernant LOKISALLE sont factices.

- Raison sociale : LOKISALLE
- Type de structure : SARL
- Adresse : 300 Boulevard de Vaugirard, 75015 Paris, France
- Mission : La société est spécialisée dans la location de salle pour l'organisation de réunion par les entreprises ou les particuliers.
- Périmètre géographique de l'activité : La société dispose de salles dans toute la France et plus précisément à Paris, Lyon et Marseille.

### Objectifs et Bénéfices

Le projet vise à offrir un ensemble de services basés sur les TIC permettant notamment :

- au grand public (visiteurs) de découvrir l'activité de LOKISALLE.
- au grand public de consulter le catalogue produits ainsi que les fiches des produits proposés par LOKISALLE.
- aux clients d'acheter en ligne des produits proposés dans le catalogue produits de LOKISALLE.
- aux membres de partager des avis sur les salles, de noter les salles, d'être informés des nouveautés et promotions de LOKISALLE.
- aux gérants (administrateurs du site) de LOKISALLE d'alimenter le catalogue produits, de consulter la liste des commandes, d'envoyer une newsletter aux membres, etc.

### Architecture logicielle

L'architecture logicielle sous-jacente au site devra utiliser les technologies récentes éprouvées.

La solution devra autant que possible remplir les spécifications suivantes :

- architecture ouverte basée sur l'utilisation du couple PHP/MySQL.
- technologies standard pour la structuration et la description des données aussi bien que des interfaces (HTML/CSS).

### Ergonomie

La page d'accueil du portail a une importance primordiale : elle doit, en quelques secondes, donner à l'utilisateur une vision de ce qu'il peut attendre du site. Tout malentendu à ce niveau peut frustrer l'utilisateur, interrompre la visite et entacher l'image de LOKISALLE.

Le site web doit être adaptable et prévoir une version responsive design.

Ces principes devront rester cohérents avec les objectifs et les contraintes du site.

Les schémas introduits dans ce document sont présents à titre explicatif, il ne s'agit pas d'une suggestion de mise en forme.

Les fonctionnalités seront les mêmes d'un projet à un autre (respect du cahier des charges).

De ce fait, il serait donc intéressant que vous nous démarquiez via le design de votre site.

Vous aurez donc carte blanche sur cette partie, l'idéal serait d'innover et d'être inventif sur les choix d'ergonomie. Le jury y sera sensible.

### Contraintes

L'ensemble du portail devra être compatible avec les navigateurs les plus utilisés du marché, soit :

- Mozilla Firefox - dernières versions,
- Internet Explorer - dernières versions,
- Chrome - dernières versions.

La sécurité sur le site devra couvrir les notions de confidentialité, de droit d'accès et d'authentification.

### Référencement

Le référencement et le positionnement d'un site au niveau des différents outils de recherche de l'Internet (annuaires, moteurs) sont d'une importance cruciale pour attirer une bonne audience.

## Planification du projet

Le planning de la prestation se conformera aux étapes suivantes :

- Création des maquettes graphiques
- Lancement de l'intégration et du développement
- Création d'une version du site Internet
- Tests, débogages, validation finale
- Mise en Ligne
- Livraison

## Précisions

« LOKISALLE est une société imaginaire. Les informations concernant LOKISALLE sont factices »

Pour le bon déroulement du projet, veillez à prendre des images et texte libre de droit.

N'oubliez pas de préciser dans les mentions légales que ce site n'est pas réalisé dans un but commercial mais dans le but de répondre à un atelier PHP.

Le site doit être sur internet et hébergé sur le web pour une présentation client.

Arguments à bannir : « je n'ai pas eu le temps », « j'étais en entreprise », « c'était trop difficile je ne l'ai pas fait », « je ne savais pas que le rendu était pour aujourd'hui », « mon pc est tombé en panne hier et je n'avais pas d'autres sauvegardes », « j'avais d'autres projets urgents », « j'étais en plein déménagement, désolé ! », etc.

## Conseil

Pensez à mettre votre site en ligne et tout « re-tester » au moins 10 jours avant les soutenances.

On ne va pas en présentation chez un client avec un site mis en ligne la veille.

## Justesse

Le respect du cahier des charges est fondamental. Vous devrez également complètement « rentré(e) » dans votre sujet afin de prévoir des tests de débogage ainsi que des tests de cohérence qui ne sont pas forcément à mentionner par le client.

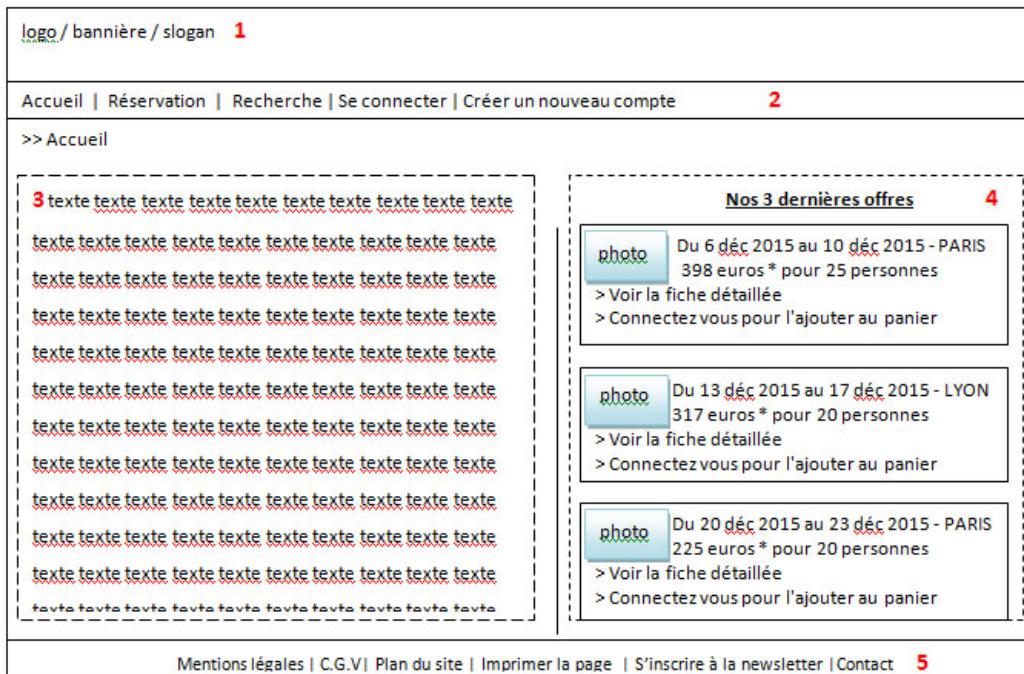
Exemple :

- Que se passe t-il si un membre tente de s'inscrire deux fois à la newsletter ?
- Peut-on créer ou réserver un produit sur des dates antérieures dans le temps ?
- Y'a-t-il des moyens d'altérer le comportement initialement prévu du site via l'url ou d'autres points d'entrée (sécurité) ?
- Que se passe t'il lorsqu'un membre est supprimé (commandes, avis, newsletter, etc.) ?
- Que se passe t'il lorsqu'une salle est supprimée (produits, avis, commandes, etc.) ?
- Est-il possible de créer 2 produits sur la même salle et sur des dates qui se chevauchent ?

Des tests de cohérence seront réalisés lors de la livraison du projet.

L'aspect sécurité est également important.

Il faut bien discerner la limite entre l'assurance de livrer un produit fiable et la responsabilité du client sur ce produit.



**Description :**

- 1 correspond à la zone du haut, nous y retrouverons le logo et la bannière.
  - 2 correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.
  - 3 correspond au texte qui présente notre activité (location de salles).
  - 4 correspond aux trois dernières offres (évidemment supérieures à la date du jour et « réservable »)
  - 5 correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommendations :

Il serait judicieux pour la zone **2** de créer un fichier « menu.inc.php » afin d'y mettre tous les liens et de l'inclure sur toutes les pages. (idem pour la zone **1** « haut.inc.php » : doctype, appel vers la feuille de style, etc. - et pour la zone **5** « bas.inc.php » : à nouveau des liens).

Texte cohérent avec l'activité pour la zone 3.

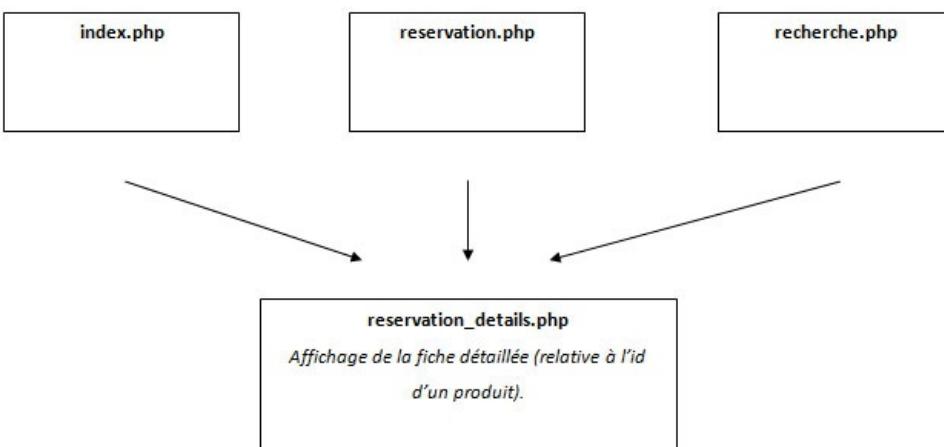
Bien entendu pour la zone **4** il s'agit d'un affichage « dynamique » les informations proviennent de la base de données (voir les tables : produits et salles).

Le lien « connectez-vous pour l'ajouter au panier » pointe sur la page connexion.php.

- 4** Deux choix s'offrent alors à l'internaute :

  - o Voir la fiche détaillée le lien pointe sur « reservation\_details.php » (cette page n'est pas répertoriée dans la zone 2 du menu). (il convient impérativement de passer l'id du produit pour avoir l'affichage complet relatif à ce produit).

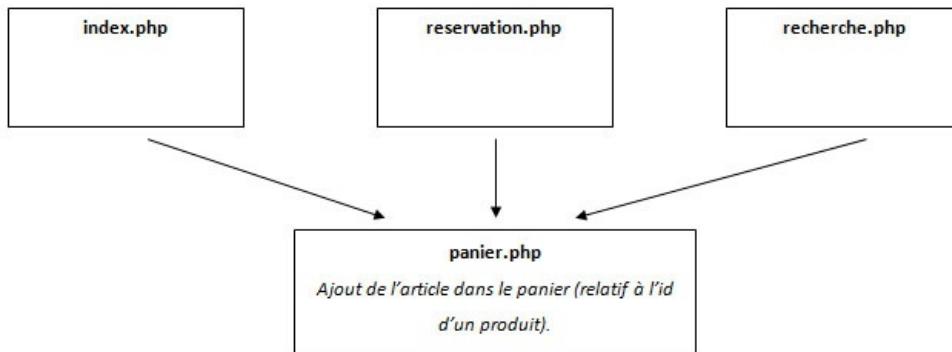
Et c'est le cas pour trois des pages du menu :



- o Après s'être connecté : Ajouter directement le produit au panier, le bouton pointe sur « panier.php ».

 <b>photo</b> Du 20 déc 2015 au 23 déc 2015 - PARIS 225 euros* pour 20 personnes	<a href="#">&gt; Voir la fiche détaillée ou</a> <a href="#" style="background-color: #f2e0e0; border: 1px solid #ccc; padding: 2px 5px;">Ajouter au panier</a>
---	--

(il faudra alors passer l'id du produit). Et c'est une possibilité pour trois des pages du menu :



Pour les pages index.php – reservation.php – recherche.php ; Si aucun produit n'est réservable, nous prévoirons un message d'information « aucune offre pour le moment ».

#### Exercice 4 : Réservation (reservation.php)

logo / bannière / slogan <b>1</b>	
Accueil   Réservation   Recherche   Se connecter   Crée un nouveau compte <b>2</b>	
>> Réservation	
<b>Toutes nos offres</b> <b>3</b>	
 Du 6 déc 2015 au 10 déc 2015 - PARIS 398 euros* pour 25 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>	 Du 13 déc 2015 au 17 déc 2015 - PARIS 387 euros* pour 20 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>
 Du 20 déc 2015 au 26 déc 2015 - LYON 325 euros* pour 20 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>	 Du 13 déc 2015 au 17 déc 2015 - PARIS 317 euros* pour 20 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>
 Du 20 déc 2015 au 23 déc 2015 - PARIS 225 euros* pour 20 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>	 Du 20 déc 2015 au 23 déc 2015 - LYON 285 euros* pour 20 personnes <a href="#">&gt; Voir la fiche détaillée</a> <a href="#">&gt; Connectez vous pour l'ajouter au panier</a>
<a href="#">Mentions légales</a>   <a href="#">C.G.V</a>   <a href="#">Plan du site</a>   <a href="#">Imprimer la page</a>   <a href="#">S'inscrire à la newsletter</a>   <a href="#">Contact</a> <b>5</b>	

#### Description :

- 1** correspond à la zone du haut, nous y retrouverons le logo et la bannière.
- 2** correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.
- 3** correspond à l'ensemble de toutes nos offres (évidemment supérieures à la date du jour et « réservable »).
- 5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommandations :

Bien entendu pour la zone **3** il s'agit d'un affichage « dynamique » les informations proviennent de la base de données (voir les tables : produits et salles).

Le lien « connectez-vous pour l'ajouter au panier » pointe sur la page connexion.php.

Cette page est similaire à la page d'accueil. Deux choix s'offrent à nous :

- Voir la fiche détaillée le lien pointe sur « reservation\_details.php » (cette page n'est pas répertoriée dans la zone **2** du menu). (il convient

impérativement de passer l'id du produit pour avoir l'affichage complet relatif à ce produit).

- Après s'être connecté : Ajouter directement le produit au panier, le bouton pointe sur « panier.php ». (Il faudra alors passer l'id du produit sur la page panier).

photo Du 20 déc 2015 au 23 déc 2015 - PARIS  
225 euros \* pour 20 personnes

> Voir la fiche détaillée ou **Ajouter au panier**

#### Exercice 4 : Recherche (recherche.php)

logo / bannière / slogan **1**

Accueil | Réservation | Recherche | Se connecter | Crée un nouveau compte **2**

>> Recherche

Recherche d'une location de salle pour réservation. **4**

*A la date du*

Mois Année

Janvier 2016 Ex : Paris

Recherche

**Resultats de votre recherche** **3**

Nombre de résultat(s) : 6

Du 6 déc 2015 au 10 déc 2015 - PARIS 398 euros * pour 25 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier	Du 13 déc 2015 au 17 déc 2015 - PARIS 387 euros * pour 20 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier
Du 20 déc 2015 au 23 déc 2015 - LYON 325 euros * pour 20 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier	Du 13 déc 2015 au 17 déc 2015 - PARIS 317 euros * pour 20 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier
Du 20 déc 2015 au 23 déc 2015 - PARIS 225 euros * pour 20 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier	Du 20 déc 2015 au 23 déc 2015 - LYON 285 euros * pour 20 personnes > Voir la fiche détaillée > Connectez vous pour l'ajouter au panier

Mentions légales | C.G.V | Plan du site | Imprimer la page | S'inscrire à la newsletter | Contact **5**

#### Description :

- correspond à la zone du haut, nous y retrouverons le logo et la bannière.
  - correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.
  - correspond à l'ensemble de toutes nos offres (évidemment supérieures à la date choisie par l'internaute et « réservable »)
  - correspond à la recherche l'internaute peut ainsi choisir la date à laquelle il souhaite louer une salle et le site affiche les salles « réservable » pour ou à partir de cette date.
- \* Par défaut, les champs déroulants seront positionnés sur le mois et l'année en cours.
- correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommendations :

Bien entendu pour la zone 3 il s'agit d'un affichage « dynamique » les informations proviennent de la base de données (voir les tables : produits et salles).

Deux choix s'offrent à nous :

- Voir la fiche détaillée le lien pointe sur « reservation\_details.php » (cette page n'est pas répertoriée dans la zone **2** du menu). (il convient impérativement de passer l'id du produit pour avoir l'affichage complet relatif à ce produit).
- Après s'être connecté : Ajouter directement le produit au panier, le bouton pointe sur « panier.php ».

 Du 20 déc 2015 au 23 déc 2015 - PARIS  
 225 euros \* pour 20 personnes  
[Voir la fiche détaillée](#) ou [Ajouter au panier](#)

Il faudra alors passer l'id du produit sur la page panier.

Il s'agit du même type d'affichage que sur la page reservation.php.

**4**, il faudra bien penser à récupérer 3 valeurs lors de la validation de la recherche par l'utilisateur et ne pas oublier de prendre en compte le format de date américain (ANNEE/MOIS/JOUR) avant de transmettre le tout.

#### Exercice 4 : Réservation en détail (reservation\_details.php)

logo / bannière / slogan **1**

Accueil | Réservation | Recherche | Se connecter | Créer un nouveau compte **2**

>> Réservation en details

**Salle Cézanne** (7/10 moyenne sur 2 avis) **3**

	Texte de description - Texte de description - Texte de description - Texte de description - Texte de description - Texte de description - Capacité : N - Catégorie : N
---	--

Informations complémentaires

Pays : ...  
 Ville : ...  
 Adresse : ...  
 Cp : ...  
 Date d'arrivée : 06/12/2015 10h  
 Date de départ : 10/12/2015 18h  
 Prix : 380€ \*  
 \*Ce prix est hors taxes  
 Accès :   
**4**

Avis

David, le 10/05/2012 à 13h53 (7/10)  
*Superbe lieu, idéal pour une conférence  
du projecteur*

Caroline, le 01/03/2012 à 10h30 (7/10)  
*Tout ce que j'attendais. Merci.*

François, le 11/12/2011 à 18h30 (7/10)  
*Spacieuse.*

Ajouter un commentaire - Note  
 **0/10**   
**...10/10**

**6\*** 

[Ajouter au panier](#)

Autres Suggestions / **7**

		
Du 6 déc au 10 déc 2015 – PARIS 398 euros * pour 25 personnes <a href="#">Voir la fiche de la salle Cézanne</a>	Du 7 déc au 10 déc 2015 – PARIS 328 euros * pour 22 personnes <a href="#">Voir la fiche de la salle Baron</a>	Du 6 déc au 10 déc 2015 – PARIS 376 euros * pour 20 personnes <a href="#">Voir la fiche de la salle Duval</a>

Mentions légales | C.G.V | Plan du site | Imprimer la page | S'inscrire à la newsletter | Contact **5**

#### Description :

- correspond à la zone du haut, nous y retrouverons le logo et la bannière.
- correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes partiellement dans le cas visiteur.
- correspond aux informations importantes de la salle : photo, titre, texte de description, capacité et catégorie.
- correspond aux informations complémentaires de la salle.
- correspond à la zone d'affichage de commentaires et de notations. Attribution possible de commentaires et de notes par les personnes connectées (admin ou membres) uniquement.

Nous restons sur la même page après soumission du commentaire ET de la note.

**6\*** le schéma montre la vision d'une personne membre pour cette zone.

**5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommendations :

Il faut bien souligner le fait que nous sommes sur la page détaillée d'un produit.

Afin d'arriver sur cette page et d'obtenir les informations attendues, cela implique que l'internaute ait cliqué sur « voir la fiche détaillée » via la page index.php ou reservation.php ou encore recherche.php et soit conduit vers la page reservation\_details.php ?id\_produit=N.

Bien entendu pour la zone **3** et **4** il s'agit d'un affichage « dynamique » les informations proviennent de la base de donnée (voir les tables suivantes : salles, produits). L'affichage d'un plan d'accès peut accompagner les informations.

Concernant la zone **6**, nous retrouvons l'affichage des commentaires des internautes souhaitant laisser leur avis sur la location de cette salle (salle et non produit, une salle peut « être » affiliée à plusieurs produits à la fois). Il faut se rapprocher des tables suivantes : (membres, avis, salles)

Dans cette zone, si l'internaute qui se rend sur cette page n'est pas membre ou admin (il sera donc visiteur) et à la place du formulaire pour le dépôt de commentaire, nous lui afficherons la phrase suivante : « il faut être connecté pour pouvoir déposer des commentaires ».

Dans cette même zone, si l'internaute à déjà laisser un commentaire pour cette salle nous lui afficherons les commentaires des autres internautes puis un message « merci pour votre contribution » à la place du formulaire d'ajout de commentaire (un membre ne peut donc laisser qu'un commentaire par salle).

Il faudra ajouter la TVA à 20% en plus pour obtenir le prix total.

Si l'internaute qui se rend sur cette page n'est pas membre ou admin (il sera donc visiteur) et à la place du bouton « ajouter au panier », nous lui afficherons la phrase suivante : « Veuillez vous inscrire ou vous connecter pour pouvoir effectuer une réservation ».

**7** Affichez une suggestion de produits similaires au produit qui est sélectionné par l'internaute afin de faciliter les recherches croisées.

Il faudra cependant proposer des produits similaires en termes de dates et de localisation choisie par l'internaute.

#### Exercice 4 : Front - Se connecter (connexion.php)

The screenshot shows a web page for user login. At the top, there is a logo/bannière/slogan labeled **1**. Below it is a navigation bar with links: Accueil | Réservation | Recherche | Se connecter | Crée un nouveau compte. A red number **2** is placed above the 'Se connecter' link. The main content area is titled '>> Connexion'. It contains two side-by-side boxes. The left box, labeled **3**, is for existing members and contains fields for 'Pseudo' and 'Mot de passe', both with input boxes, and a 'Se souvenir de moi' checkbox. The right box, labeled **4**, is for new members and contains a link 'Inscrivez-vous'. At the bottom of the page, there is a footer with links: Mentions légales | C.G.V | Plan du site | Imprimer la page | S'inscrire à la newsletter | Contact. A red number **5** is placed next to the 'Contact' link.

Description : **1** correspond à la zone du haut, nous y retrouverons le logo et la bannière.

**2** correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.

**3** correspond à l'interface de connexion afin qu'un membre puisse accéder à son compte.

**4** correspond à un simple lien qui propose à un visiteur de s'inscrire afin de devenir membre.

**5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommendations :

**3** Seulement si l'internaute a le bon pseudo ET le bon mot de passe il doit pouvoir accéder à son profil « profil.php », il suffit de faire une redirection. Attention si ce n'est pas le cas, nous restons sur cette page en lui signifiant un message d'erreur.

La case « Se souvenir de moi » peut être cochée par l'utilisateur, ce choix est facultatif, mais s'il la coche vous devrez sauvegarder son pseudo pour les prochaines connexions. C'est-à-dire que s'il se déconnecte et qu'il tente de se reconnecter via cette même page « connexion.php », il ne devrait plus avoir à renseigner son pseudo.

**4** Voici un simple lien qui mène vers la page « inscription.php » si le membre n'a pas encore de compte.

**2** Le lien « Inscrivez-vous » sur la partie de droite pointe vers la page « inscription.php ».

#### Exercice 4 : Front - Mot de passe perdu (motdepasseperdu.php)

logo / bannière / slogan <b>1</b>	
Accueil   Réservation   Recherche   Se connecter   Crée un nouveau compte	<b>2</b>
>> Mot de passe oublié	
<div style="border: 1px dashed black; padding: 10px;"> <b>3</b>            Afin de pouvoir réinitialiser votre mot de passe, vous devez nous fournir votre adresse email : <input type="text"/>  <input type="button" value="Valider"/> </div>	
Mentions légales   C.G.V   Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>	

Description :

- 1** correspond à la zone du haut, nous y retrouverons le logo et la bannière.
- 2** correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.
- 3** correspond à la zone de saisie permettant à l'internaute de saisir son adresse email pour recevoir son nouveau mot de passe.
- 5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

Recommandations : **2** Cette page n'est pas répertoriée dans la zone 2 du menu et n'est accessible qu'après s'être rendu sur la page de connexion.

**3** L'internaute doit saisir l'adresse email qu'il a utilisée lors de son inscription. A partir de cette information, vous pourrez lui générer un nouveau mot de passe et lui communiquer par email sans oublier de mettre à jour la base de données. Il ne faut pas oublier de gérer le cas où l'internaute saisit une adresse email qui n'existe pas.

**Exercice 4 : Front - Crée un nouveau compte (inscription.php)**

logo / bannière / slogan <b>1</b>	
Accueil   Réservation   Recherche   Se connecter   Crée un nouveau compte	<b>2</b>
>> Mot de passe oublié	
<div style="border: 1px dashed black; padding: 10px;"> <b>3</b>            Afin de pouvoir réinitialiser votre mot de passe, vous devez nous fournir votre adresse email : <input type="text"/>  <input type="button" value="Valider"/> </div>	
Mentions légales   C.G.V   Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>	

Description :

- 1** correspond à la zone du haut, nous y retrouverons le logo et la bannière.
- 2** correspond au menu quand un visiteur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas visiteur.
- 3** correspond au formulaire d'inscription afin qu'un membre puisse s'inscrire.
- 5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

Recommandations :

**3** Vérifier que vous avez récupéré toutes les informations du formulaire avant de les envoyer en base de données pour l'insertion dans la table membres.  
 Vérifier que l'adresse email soit correcte.  
 Le pseudo et le mot de passe doivent faire au minimum 3 caractères.

Les pages de connexion, inscription, mot de passe oublié, etc. ne doivent être accessibles uniquement en tant que visiteur (non connecté).  
 Si un membre connecté venait à se rendre sur ces pages par mégarde, nous le redirigerons ou lui mentionnerons qu'il est déjà connecté

#### Exercice 4 : Front - Mentions Légales (mentions.php)

Description :

3 correspond au texte des Mentions Légales

Recommendations :

Aucune recommandation particulière

#### Exercice 4 : Front - Conditions Générales de Vente (cgv.php)

Description :

3 correspond au texte des CGV

Recommendations :

Aucune recommandation particulière

#### Exercice 4 : Front - Plan du site (plan.php)

Description :

3 correspond au texte des CGV

Recommendations :

Aucune recommandation particulière

#### Exercice 4 : Front - Imprimer la page

Un simple lien pour imprimer la page courante.

#### Exercice 4 : Front - S'inscrire à la newsletter (newsletter.php)

Description :

La page contient un lien permettant de s'inscrire à la newsletter.

Recommendations :

Attention l'internaute doit être membre (ou admin) pour voir apparaître cette proposition d'abonnement. Si l'internaute est visiteur, il ne possèdera pas d'id\_membre, nous l'inviterons à s'inscrire au site avant de s'abonner à la newsletter. (table : newsletter).

logo / bannière / slogan <b>1</b>
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b>
>> S'inscrire à la newsletter
Je souhaite m'abonner à la newsletter et recevoir les actualités de LOKISALLE. <b>3</b>

Description :

La page contient un lien permettant de s'inscrire à la newsletter.

Recommendations :

Attention l'internaute doit être membre (ou admin) pour voir apparaître cette proposition d'abonnement. Si l'internaute est visiteur, il ne possèdera pas d'id\_membre, nous l'inviterons à s'inscrire au site avant de s'abonner à la newsletter. (table : newsletter).

#### Exercice 4 : Front - Contact (contact.php)

logo / bannière / slogan <b>1</b>	
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b>	
>> Contact	
Sujet	<input type="text"/>
Message	<input type="text"/> <b>3</b>
<input type="button" value="Envoyer"/>	
Mentions légales   C.G.V  Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>	

Description :

**3** correspond à l'affichage du formulaire d'envoi d'email.

Recommandations :

**3**, un membre peut envoyer un message à l'administrateur du site.

Si l'internaute qui souhaite contacter l'admin est simple visiteur, il peut également lui envoyer un message, pour cela il faudra ajouter un champ « expéditeur » dans le formulaire de contact afin que l'administrateur sache à quelle adresse email il doit répondre.

**Exercice 4 : Front - Profil (profil.php)**

logo / bannière / slogan <b>1</b>										
Accueil   Réservation   Recherche   Se déconnecter   Voir votre profil   Voir votre panier <b>2</b>										
>> Profil										
<b>3</b> voici vos informations	<b>4</b> vos dernières commandes									
Votre pseudo : test votre email est: test@test.fr votre ville est: Paris votre cp est: 75005 votre adresse est: pas loin  <a href="#">Mettre à jour mes informations</a>	<table border="1"> <thead> <tr> <th>Numéro de suivi</th> <th>date</th> <th>Facture</th> </tr> </thead> <tbody> <tr> <td>71622</td> <td>27/05/2011</td> <td><a href="#">Voir</a></td> </tr> <tr> <td>71653</td> <td>15/07/2013</td> <td><a href="#">Voir</a></td> </tr> </tbody> </table>	Numéro de suivi	date	Facture	71622	27/05/2011	<a href="#">Voir</a>	71653	15/07/2013	<a href="#">Voir</a>
Numéro de suivi	date	Facture								
71622	27/05/2011	<a href="#">Voir</a>								
71653	15/07/2013	<a href="#">Voir</a>								
Mentions légales   C.G.V  Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>										

Description :

**1** correspond à la zone du haut, nous y retrouverons le logo et la bannière.

**2** correspond au menu (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas membre. Vous pouvez remarquer que les liens « Connexion | Créer un nouveau compte » se sont transformés en « Se déconnecter ».

**3** correspond à la zone récapitulative des informations du membre. Il est également possible qu'un membre mette à jour ses informations.

**4** correspond à la zone récapitulative des dernières commandes passées sur le site.

**5** correspond à la zone du bas, nous y retrouvons plusieurs liens.

Recommandations :

**3** et **4** Un membre ou un admin ne peut accéder à son profil que s'il s'est connecté avant via la page « connexion.php », et lors de sa connexion nous lui « remplissons » une session, il suffit de ressortir les informations de cette session sur la page « profil.php » afin de récupérer les informations sur le membre ou l'admin. Cette page n'est donc pas disponible pour les visiteurs.

**3** Le lien « mettre à jour mes informations » doit ouvrir un formulaire de modification de compte. Il sera similaire au formulaire d'inscription. Cependant les informations actuelles du membre doivent apparaître dans les zones de saisies du formulaire afin que le membre puisse rectifier seulement ce qu'il souhaite sans devoir re-saisir chacune des informations.

**Exercice 4 : Front - Panier (panier.php)**

logo / bannière / slogan <b>1</b>																														
Accueil   Réservation   Recherche   Se déconnecter   Voir votre profil   Voir votre panier <b>2</b>																														
>> Panier																														
<b>Votre panier</b> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Produit</th><th>Salle</th><th>Photo</th><th>Ville</th><th>Capacité</th><th>Date Arrivée</th><th>Date Départ</th><th>Retirer</th><th>Prix HT</th><th>TVA</th></tr> </thead> <tbody> <tr> <td>2</td><td>Baron</td><td></td><td>Paris</td><td>70</td><td>17 décembre 2012</td><td>21 décembre 2012</td><td>x</td><td>590€</td><td>19.6 %</td></tr> <tr> <td colspan="8" style="text-align: right;"><b>Prix Total TTC :</b></td><td colspan="2"><b>705.64 €</b></td></tr> </tbody> </table> <p>J'accepte les conditions générales de vente (<a href="#">voir</a>) <input type="checkbox"/></p> <p>Utiliser un code promo : <input type="text"/></p> <p style="text-align: right;"><b>Payer</b></p> <p><a href="#">+ Vider mon panier</a> <b>3</b></p>	Produit	Salle	Photo	Ville	Capacité	Date Arrivée	Date Départ	Retirer	Prix HT	TVA	2	Baron		Paris	70	17 décembre 2012	21 décembre 2012	x	590€	19.6 %	<b>Prix Total TTC :</b>								<b>705.64 €</b>	
Produit	Salle	Photo	Ville	Capacité	Date Arrivée	Date Départ	Retirer	Prix HT	TVA																					
2	Baron		Paris	70	17 décembre 2012	21 décembre 2012	x	590€	19.6 %																					
<b>Prix Total TTC :</b>								<b>705.64 €</b>																						
Tous nos articles sont calculés avec le taux de TVA à 20% Règlement: Par Chèque uniquement Nous attendons votre règlement par chèque à l'adresse suivante: Ma boutique - 1 Rue Boswellia, 75000 Paris, France																														
Mentions légales   C.G.V   Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>																														

#### Description :

- 1 correspond à la zone du haut, nous y retrouverons le logo et la bannière.
- 2 correspond au menu (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas membre. Vous pouvez remarquer que les liens « Connexion | Créer un nouveau compte » se sont transformés en « Se déconnecter ».
- 3 correspond au panier du membre ou admin.
- 5 correspond à la zone du bas, nous y retrouvons plusieurs liens.

#### Recommendations :

- 3 On ne peut accéder qu'à son panier en tant que membre, administrateur via la zone de menu 2 ou alors en étant envoyé par le bouton « ajouter au panier » sur les pages suivantes :  
« index.php » - « reservation.php » - « reservation\_details.php » - « recherche.php »

Si l'internaute n'est pas connecté (donc simple visiteur) et se retrouve sur la page « panier.php » nous lui afficherons « vous devez vous connecter avant d'accéder à votre panier ».

Si l'internaute (membre ou admin) qui accède à son panier tente d'ajouter deux fois le même produit, nous lui afficherons « ce produit est déjà dans votre panier ». Effectivement il ne peut réserver deux fois le même produit.

Vider mon panier est un lien vers la même page « panier.php » qui déclenche une action afin de « vider » le contenu de son panier.

La croix « x » à côté du produit est un lien permettant de « retirer » un des produit(s) du panier sans affecter le restant des produits dans le panier.

L'internaute peut payer (seulement s'il est membre ou admin) et surtout s'il a coché la case « J'accepte les conditions générales de vente » le lien « voir » mène vers « cgv.php ».

Si l'internaute utilise un code promo, il faudra impérativement vérifier si le code promo qui l'utilise est associé au produit qu'il commande et si c'est le cas porter les répercussions sur le prix de ce produit.

L'idéal serait de recharger la page avant validation définitive du panier et d'afficher la réduction réalisée en euros, le nouveau prix ainsi que le % économisé entre le prix de base et le nouveau prix réduit calculé.

C'est seulement après avoir accepté les CGV et avoir payé qu'une occurrence se crée dans les tables commandes et details\_commandes.

C'est également à ce moment-là que l'état du produit passe à 1 pour cet id de produit dans la table produit. (0 correspond à « réservable » donc « disponible », 1 correspond à « réserver » et par conséquent « n'étant plus disponible »).

Logiquement ce produit à présent payé ne doit plus apparaître dans les produits « réservables » sur les pages suivantes : « index.php » - « reservation.php » - « reservation\_details.php » - « recherche.php »

Après avoir payé, le membre connecté reçoit un email de confirmation lui indiquant son numéro de commande et en pièce jointe le récapitulatif du produit acheté sur une facture générée au format PDF (cette facture devra également être consultable sur l'espace profil du membre).

## Exercice 4 : Back - Gestion des salles (gestion\_salles.php)

The screenshot shows a website layout for 'Gestion des salles'. At the top, there's a header with a logo/bannière/slogan (labeled 1). Below it is a navigation menu with links like Accueil, Réservation, Recherche, Se déconnecter, Voir votre Profil, Voir votre Panier, + Gestion des salles, + Gestion des produits, + Gestion des membres, + Gestion des commandes, + Gestion des avis, + Gestion codes promo, + Statistiques, + Envoyer la newsletter (labeled 2). A sidebar on the left contains a link '>> Gestion des salles' and two items under a dashed box (labeled 3): '- Ajouter une salle' (labeled 3.1) and '- Affichage des salles' (labeled 3.2). At the bottom, there are links for Mentions légales, C.G.V, Plan du site, Imprimer la page, S'inscrire à la newsletter, and Contact (labeled 5).

Description :

**2** correspond au menu quand un administrateur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas administrateur.

**3**, Les deux liens pointent sur la même page : « gestion\_salles.php » et déclenchent une action.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3.1**, dans cette zone nous avons un premier lien, si l'administrateur clique sur « ajouter une salle » un formulaire apparaît en dessous et il doit pouvoir entrer des informations pour ajouter une salle (ces informations sont ensuite sauvegardées dans la table salle).

**3.2**, nous avons également un second lien, si l'administrateur clique dessus « affichage des salles » il doit pouvoir observer toutes les salles dont il dispose (il s'agit du contenu de la table salle).

Suite à cet affichage, il pourra modifier une salle existante si l'administrateur le souhaite en cliquant sur un lien « modification », lors de cette action, il doit retrouver toutes les valeurs actuelles de la salle qu'il s'apprête à changer dans le formulaire de modification affiché.

Nous lui donnerons également la possibilité de supprimer une salle avec toutes les conséquences que cela implique.

## Exercice 4 : Back - Gestion des produits (gestion\_produits.php)

logo / bannière / slogan <b>1</b>				
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b> + Gestion des salles      + Gestion des produits      + Gestion des membres      + Gestion des commandes + Gestion des avis      + Gestion codes promo      + Statistiques      + Envoyer la newsletter				
>> Gestion des produits				
<div style="border: 1px dashed black; padding: 5px;"> <b>3</b>  - Ajouter un produit  - Affichage des produits </div>				
<a href="#">Mentions légales</a>   <a href="#">C.G.V</a>   <a href="#">Plan du site</a>   <a href="#">Imprimer la page</a>   <a href="#">S'inscrire à la newsletter</a>   <a href="#">Contact</a> <b>5</b>				

Description :

**2** correspond au menu quand un administrateur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas administrateur.

**3**, Les deux liens pointent sur la même page : « gestion\_products.php » et déclenchent une action.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

3, dans cette zone nous avons un premier lien, si l'administrateur clique sur « ajouter un produit » un formulaire apparaît en dessous et il doit pouvoir entrer des informations pour ajouter un produit (ces informations sont ensuite sauvegardées dans la table produit).

Voici un exemple du formulaire attendu :

>> Gestion des produits – Ajouter un produit

Choisir une salle parmi les salles existantes :

Date d'arrivée :

Date de départ :

Prix :

Attribution remise parmi les codes promo existant :

La date de départ doit naturellement être supérieure à la date d'arrivée.

La date d'arrivée doit être supérieure à la date du jour.

Suite à la validation de ce formulaire, il faudra donc lancer une insertion dans la table produit afin que les deux (produits et salles) soient bien reliés.

Attention : une salle peut être associée à plusieurs produits à la fois.

Exemple :

Salles	Produits	Description
id_salle : 1	salle Cézanne – paris – id_produit : 1 Date_arrivee : 2015-11-22 Date_arrivee : 2015-11-27 Prix : 550 euros	L'id_salle 1 est relié avec l'id_produit 1. C'est-à-dire que la salle Cézanne à Paris sera disponible du 22 au 27 novembre pour 550 euros
id_salle : 1	salle Cézanne – paris – id_produit : 2 Date_arrivee : 2015-11-29	L'id_salle 1 est relié avec l'id_produit 2. C'est-à-dire que cette même salle Cézanne à Paris sera disponible la semaine suivante du 29 novembre au 03 décembre pour 600 euros

	Date_arrivee : 2015-12-03 Prix : 600 euros	Salle de 29 novembre au 03 décembre pour ces tarifs.
id_salle : 2	salle Mozart - paris - id_produit : 3 Date_arrivee : 2015-11-29 Date_arrivee : 2015-12-03 Prix : 380 euros	L'id_salle 2 est relié avec l'id_produit 3. C'est-à-dire que la salle Mozart à Paris sera disponible du 29 nov au 03 déc pour 380 euros

Une salle est donc susceptible d'appartenir à plusieurs produits. En revanche chaque produit est unique.

3, nous avons également un second lien, si l'administrateur clique dessus « affichage des produits » il doit pouvoir observer tous les produits dont il dispose (il s'agit du contenu de la table produit).

De plus, il doit pouvoir cliquer sur les entêtes suivantes : « date\_arrivee » – « date\_depart » ou encore l'entête « prix » afin d'afficher les produits dans l'ordre croissant/décroissant des prix.

L'administrateur doit aussi pouvoir modifier et supprimer ses produits.

#### Exercice 4 : Back - Gestion des membres (gestion\_membres.php)

logo / bannière / slogan <b>1</b>																																																
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b> + Gestion des salles      + Gestion des produits      + Gestion des membres      + Gestion des commandes + Gestion des avis      + Gestion codes promo      + Statistiques      + Envoyer la newsletter																																																
>> Gestion des membres																																																
<b>3</b> <table border="1"> <thead> <tr> <th><u>id_membre</u></th> <th>Pseudo</th> <th>Mdp</th> <th>Nom</th> <th>Prenom</th> <th>Email</th> <th>Sexe</th> <th>Ville</th> <th>Cp</th> <th>adresse</th> <th>statut</th> <th>supprimer</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>test</td> <td>test</td> <td>test</td> <td>test</td> <td>test@test.fr</td> <td>f</td> <td>Paris</td> <td>75</td> <td>Par ici</td> <td>0</td> <td>X</td> </tr> <tr> <td>2</td> <td>admin</td> <td>admin</td> <td>admin</td> <td>admin</td> <td>a@a.fr</td> <td>m</td> <td>Paris</td> <td>75</td> <td>Par ici</td> <td>1</td> <td></td> </tr> <tr> <td>3</td> <td>essai</td> <td>essai</td> <td>essai</td> <td>essai</td> <td>ei@ei.fr</td> <td>f</td> <td>Lyon</td> <td>69</td> <td>Par ici</td> <td>0</td> <td>X</td> </tr> </tbody> </table> - Création d'un nouveau compte administrateur <b>4</b>	<u>id_membre</u>	Pseudo	Mdp	Nom	Prenom	Email	Sexe	Ville	Cp	adresse	statut	supprimer	1	test	test	test	test	test@test.fr	f	Paris	75	Par ici	0	X	2	admin	admin	admin	admin	a@a.fr	m	Paris	75	Par ici	1		3	essai	essai	essai	essai	ei@ei.fr	f	Lyon	69	Par ici	0	X
<u>id_membre</u>	Pseudo	Mdp	Nom	Prenom	Email	Sexe	Ville	Cp	adresse	statut	supprimer																																					
1	test	test	test	test	test@test.fr	f	Paris	75	Par ici	0	X																																					
2	admin	admin	admin	admin	a@a.fr	m	Paris	75	Par ici	1																																						
3	essai	essai	essai	essai	ei@ei.fr	f	Lyon	69	Par ici	0	X																																					
Mentions légales   C.G.V   Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>																																																

Description :

**3** correspond à l'affichage des membres du site.

**4** correspond à l'affichage d'un nouveau formulaire d'inscription.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3**, nous avons l'affichage des membres (il s'agit du contenu de la table membres).

Pour chaque membre, l'administrateur dispose d'un lien sous forme d'une croix représentée par un « X » qui peut lui permettre de supprimer le compte d'un membre.

**4**, L'administrateur à la possibilité de créer un nouveau compte « administrateur » à l'un de ses collaborateurs. En cliquant sur ce lien, un formulaire d'inscription apparaît afin qu'il puisse rentrer les informations qui permettront de créer ce nouveau compte.

#### Exercice 4 : Back - Gestion des commandes (gestion\_commandes.php)

<a href="#">logo / bannière / slogan</a> <b>1</b>																		
+ Gestion des salles	+ Gestion des produits	+ Gestion des membres	+ Gestion des commandes															
<a href="#">+ Gestion des avis</a> <a href="#">+ Gestion codes promo</a> <a href="#">+ Statistiques</a> <a href="#">+ Envoyer la newsletter</a>																		
> Gestion des commandes																		
<table border="1"> <thead> <tr> <th><i>id_commande</i></th><th><i>id_membre</i></th><th><i>montant</i>▲▼</th></tr> </thead> <tbody> <tr> <td><a href="#">313225</a></td><td>28</td><td>516.22</td></tr> <tr> <td><a href="#">321542</a></td><td>16</td><td>328.60</td></tr> <tr> <td><a href="#">321559</a></td><td>7</td><td>150.43</td></tr> <tr> <td><a href="#">327699</a></td><td>16</td><td>713.31</td></tr> </tbody> </table>				<i>id_commande</i>	<i>id_membre</i>	<i>montant</i> ▲▼	<a href="#">313225</a>	28	516.22	<a href="#">321542</a>	16	328.60	<a href="#">321559</a>	7	150.43	<a href="#">327699</a>	16	713.31
<i>id_commande</i>	<i>id_membre</i>	<i>montant</i> ▲▼																
<a href="#">313225</a>	28	516.22																
<a href="#">321542</a>	16	328.60																
<a href="#">321559</a>	7	150.43																
<a href="#">327699</a>	16	713.31																
Le Chiffre d'affaires (CA) de notre société est de : <a href="#">1708.56 €</a>																		
<b>3</b>																		
<a href="#">Mentions légales</a>   <a href="#">C.G.V</a>   <a href="#">Plan du site</a>   <a href="#">Imprimer la page</a>   <a href="#">S'inscrire à la newsletter</a>   <a href="#">Contact</a> <b>5</b>																		

Description :

**3** correspond à l'affichage des commandes passées (et payées) sur notre site.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3**, nous avons l'affichage des commandes (il s'agit du contenu de la table commandes).

Nous affichons à titre informatif le chiffre d'affaires (CA).

Les données sous la colonne « id\_commande » sont sous forme de lien cliquable et permettent d'afficher le détail d'une commande (sur la même page).

Exemple :

<i>id_commande</i>	<i>montant</i> ▲▼	<i>date</i> ▲▼	<i>id_membre</i>	<i>Pseudo</i>	<i>id_produit</i>	<i>id_salle</i>	<i>ville</i> ▲▼
313225	300.12	08/05/2012	28	Marie	167	56	Paris
313225	216.10	08/05/2012	28	Marie	39	107	paris

**Exercice 4 : Back - Gestion des avis (gestion\_avis.php)**

<a href="#">logo / bannière / slogan</a> <b>1</b>																								
+ Gestion des salles	+ Gestion des produits	+ Gestion des membres	+ Gestion des commandes																					
<a href="#">+ Gestion des avis</a> <a href="#">+ Gestion codes promo</a> <a href="#">+ Statistiques</a> <a href="#">+ Envoyer la newsletter</a>																								
>> Gestion des avis																								
<table border="1"> <thead> <tr> <th><i>id_avis</i></th><th><i>id_membre</i></th><th><i>id_salle</i></th><th>commentaire</th><th><i>note</i>▲▼</th><th><i>date</i>▲▼</th><th>supprimer</th></tr> </thead> <tbody> <tr> <td>1</td><td>7</td><td>5</td><td>Magnifique !</td><td>9</td><td>2011-03-25</td><td>X</td></tr> <tr> <td>2</td><td>6</td><td>11</td><td>A recommander...</td><td>7</td><td>2011-03-28</td><td>X</td></tr> </tbody> </table>				<i>id_avis</i>	<i>id_membre</i>	<i>id_salle</i>	commentaire	<i>note</i> ▲▼	<i>date</i> ▲▼	supprimer	1	7	5	Magnifique !	9	2011-03-25	X	2	6	11	A recommander...	7	2011-03-28	X
<i>id_avis</i>	<i>id_membre</i>	<i>id_salle</i>	commentaire	<i>note</i> ▲▼	<i>date</i> ▲▼	supprimer																		
1	7	5	Magnifique !	9	2011-03-25	X																		
2	6	11	A recommander...	7	2011-03-28	X																		
<b>3</b>																								
<a href="#">Mentions légales</a>   <a href="#">C.G.V</a>   <a href="#">Plan du site</a>   <a href="#">Imprimer la page</a>   <a href="#">S'inscrire à la newsletter</a>   <a href="#">Contact</a> <b>5</b>																								

Description :

**3** correspond à l'affichage des avis laissés sur le site.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3**, nous avons l'affichage des avis laissés sur le site (il s'agit du contenu de la table avis).

Pour chaque avis, l'administrateur dispose d'un lien sous forme d'une croix représentée par un « X » qui peut lui permettre de supprimer l'avis d'un membre s'il le juge incorrect.

#### Exercice 4 : Back - Gestion des codes promo (gestion\_promos.php)

logo / bannière / slogan <b>1</b>
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b> + Gestion des salles + Gestion des produits + Gestion des membres + Gestion des commandes + Gestion des avis + Gestion codes promo + Statistiques + Envoyer la newsletter
>> Gestion des codes de promotion
<b>3</b> - Ajouter un code promo - Affichage des codes promo
Mentions légales   C.G.V  Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>

##### Description :

2 correspond au menu quand un administrateur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous sommes dans le cas administrateur.

3, Les deux liens pointent sur la même page : « gestion\_promos.php » et déclenchent une action.

##### Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

3, dans cette zone nous avons un premier lien, si l'administrateur clique sur « ajouter un code promo » un formulaire apparaît en dessous et il doit pouvoir entrer des informations pour ajouter un code promo (ces informations sont ensuite sauvegardées dans la table promotions).

3, nous avons également un second lien, si l'administrateur clique dessus « affichage des codes promo » il doit pouvoir observer tous les codes promo dont il dispose (il s'agit du contenu de la table promotions).

L'administrateur doit également pouvoir supprimer des codes de promotion (il faudra alors porter une attention particulière aux produits qui y sont éventuellement associés).

L'administrateur enverra les codes de promotion qu'il souhaite diffuser lors de l'envoi de la newsletter.

#### Exercice 4 : Back - Statistiques (statistiques.php)

logo / bannière / slogan <b>1</b>
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b> + Gestion des salles + Gestion des produits + Gestion des membres + Gestion des commandes + Gestion des avis + Gestion codes promo + Statistiques + Envoyer la newsletter
>> Statistiques
<b>3</b> - Top 5 des salles les mieux notés - Top 5 des salles les plus vendues - Top 5 des membres qui achète le plus (en termes de quantité). - Top 5 des membres qui achète le plus cher (en termes de prix)
Mentions légales   C.G.V  Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>

##### Description :

2 correspond au menu quand un administrateur arrive (attention il est évolutif suivant l'internaute : visiteur, membre, administrateur). Ci-dessus nous

sommes dans le cas administrateur.

**3**, Les trois liens pointent sur la même page : « statistiques.php » et déclenchent une action.

Recommandations :

Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3**, dans cette zone nous avons trois liens permettant à l'administrateur d'avoir des statistiques selon les différentes informations qu'il demande.  
L'affichage des résultats doit se faire sur la même page.

**Exercice 4 : Back - Envoyer la newsletter (envoi\_newsletter.php)**

logo / bannière / slogan <b>1</b>					
Accueil - Réservation - Recherche - Se déconnecter - Voir votre Profil - Voir votre Panier <b>2</b> + Gestion des salles      + Gestion des produits      + Gestion des membres      + Gestion des commandes + Gestion des avis      + Gestion codes promo      + Statistiques      + Envoyer la newsletter					
>> Envoyer la newsletter					
Nombre d'abonné à la newsletter : 31.					
Expéditeur <input type="text"/>					
Sujet <input type="text"/>					
Message <input type="text"/>					
Envoi de la newsletter aux membres <b>3</b>					
Mentions légales   C.G.V   Plan du site   Imprimer la page   S'inscrire à la newsletter   Contact <b>5</b>					

Description :

**3** correspond à l'affichage du formulaire d'envoi d'email

Recommandations :

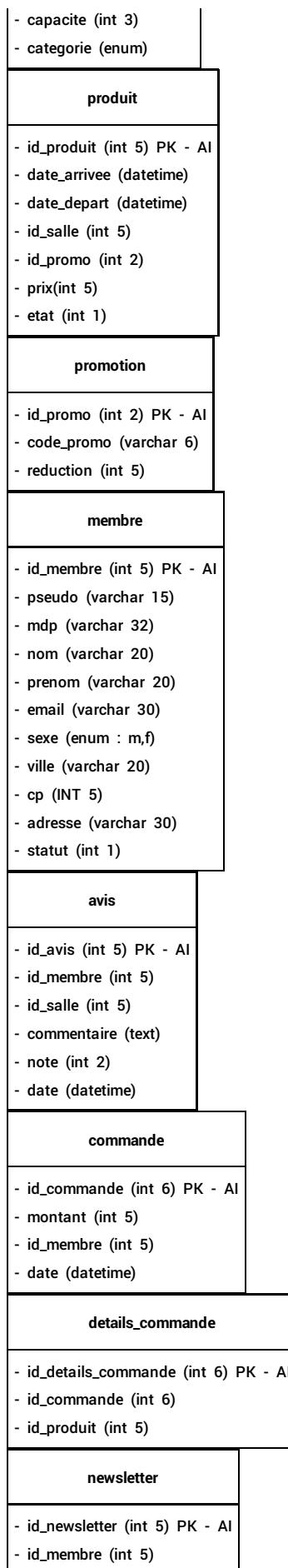
Cette page n'est disponible que si l'on est administrateur (vous devez penser à verrouiller l'accès dans tous les autres cas).

**3**, Nous pouvons choisir un message et ce message doit s'envoyer à l'ensemble des membres « abonné » à la newsletter (autrement dit, il faut faire une correspondance entre la table newsletter, et la table membres).

**Exercice 4 : Back - La base de données**

Voici notre modélisation :

salle
- id_salle (int 5) PK - AI
- pays (varchar 20)
- ville (varchar 20)
- adresse (text)
- cp (varchar 5)
- titre (varchar 200)
- description (text)
- photo (varchar 200)



© Copyright Aucune reproduction, même partielle (textes, documents, images, etc.), ne peut être faite sans l'accord de son auteur.

