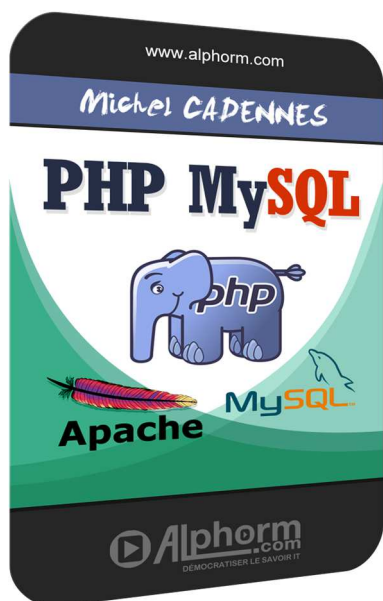




**Alphorm**.com

## Formation PHP & MySQL

Site : <http://www.alphorm.com>

Blog : <http://blog.alphorm.com>



**Michel CADENNES**

Formateur et Consultant indépendant  
Web, Gestion des connaissances

Formation PHP MySQL

alphorm.com™ ©



## Plan

- Présentation du formateur
- Le cursus global autour de PHP
- Le plan du cours
- Les objectifs
- A qui s'adresse ce cours ?
- Les pré-requis



Formation PHP MySQL

alphorm.com™ ©

## Présentation

### Michel CADENNES

- Développeur et architecte d'information indépendant
- Orientation : gestion des connaissances, intelligence artificielle, web sémantique
- Mes profils :
  - LinkedIn : <https://www.linkedin.com/in/michel-cadennes-2a287726>
  - Hopwork : <https://www.hopwork.fr/profile/michelcadennes>
  - Twitter : <https://www.twitter.com/tchevengour>
  - Github : <https://github.com/Septentrion>



## Pourquoi ce cours ?

- Le web est très majoritairement organisé autour de la plate-forme appelée LAMP (Linux, Apache, MySQL, PHP)
- La maîtrise de la plate-forme LAMP donne accès à la compréhension de tout l'écosystème d'applications web écrites en PHP
- Ces connaissances permettent d'arbitrer la question : doit-on faire des sites « statiques » ou des sites « dynamiques » ?
- Maîtriser les outils de publication sur Internet demande un effort technique assez limité

## Objectifs

- L'objectif de ce cours est de vous permettre d'appréhender l'architecture globale de ce qu'on appelle la plate-forme LAMP :
  - Approcher les ressources du système d'exploitation (Linux, OS X) qui vous permettront d'installer (relativement) facilement les logiciels nécessaires
  - Examiner les solutions prêtes à l'emploi de type XAMPP ou MAMP
  - Comprendre les bases du langage PHP
  - Comprendre le fonctionnement d'une base de données
  - Ecrire un premier programme

## Coursus PHP sur Alphorm



## Plan du cours

---

### 1. Installation d'un serveur web

1. Apache, PHP, MySQL, FTP
2. Configuration du serveur
3. Mise en route du serveur

### 2. Utilisation de XAMPP comme serveur

1. Installation de XAMPP
2. Configuration du serveur
3. Questions de sécurité
4. Mise en œuvre du serveur

### 3. Les bases de PHP

1. Le fonctionnement de PHP
2. Les requêtes
3. Les types
4. Les structures de contrôle
5. Les objets

### 4. Les bases de MySQL

1. Le calcul relationnel
2. Les bases du langage SQL
3. Utiliser phpMyAdmin
4. Le lien entre PHP et MySQL
5. Les résultats d'une requête SQL

## Application

---

- Pour appliquer concrètement les techniques dont nous parlerons, nous construirons une interface pour gérer une liste de tâches à faire.
- Pour cela nous aurons besoin de :
  - définir ce que c'est qu'une tâche
  - représenter les tâches au moyen d'un schéma de base de données
  - construire les pages dynamiques qui afficheront les tâches
  - ajouter et modifier les tâches



## A qui s'adresse ce cours ?

---

- A tous ceux qui, même avec une connaissance réduite des systèmes d'exploitation, des réseaux et de la programmation veulent apprendre à mettre en œuvre un site web



## Pré-requis

---

- Une connaissance minimale de GNU/Linux
- Une familiarité avec l'algorithmique de base
- Une bonne pratique de l'écriture de pages HTML



# Go on !



## Installer un serveur web

- Le serveur HTTP

## Plan

---

- Installer Apache
- Mise en route du serveur
- Les fichiers de configuration
- Autres serveurs HTTP
  - NGINX
  - lighthttpd
  - PHP



## Installer Apache

---

- Installer Apache se fait très simplement grâce à **apt-get**

**apt-get install apache2**

## Premier test

---

- Une fois Apache est installé, vous pouvez accéder au site par défaut via l'adresse IP de votre serveur.
- Si vous utilisez un serveur local, l'adresse sera localhost ou 127.0.0.1

```
curl xxx.yyy.zzz.ttt
```





## Activation du module userdir

- La première chose à faire est d'activer le module **userdir**, qui permet de définir le répertoire racine de l'espace web pour un utilisateur
- La commande **a2enmod** permet d'activer tel ou tel module d'Apache

```
a2enmod userdir
```

- Il est ensuite nécessaire de redémarrer le serveur Apache

```
/etc/init.d/apache2 restart
```



## envvars

- Le fichier **envvars** contient les variables d'environnement du serveur
- Il faut juste vérifier que l'utilisateur « propriétaire » du serveur et le groupe associé sont bien configurés

```
export APACHE_RUN_USER=www-data  
export APACHE_RUN_GROUP=www-data
```

## dir.conf

---

- Dans le dossier **mods-enabled**, le fichier **dir.conf** permet de configurer quels sont les fichiers par défaut que le serveur doit chercher lorsque l'on envoie une requête

**DirectoryIndex** index.html index.php index.xhtml

## userdir.conf

---

- Dans le dossier **mods-available**, le fichier **dir.conf** permet de déterminer si les utilisateurs sont bien autorisés à se connecter.
- On vérifie :

**UserDir** public\_html



## Activer PHP pour les utilisateurs

- Dans le dossier **mods-enabled**, le fichier **php5.conf** permet d'activer l'exécution de PHP pour tous les utilisateurs
- On commente la ligne :

```
# php_admin_value engine Off
```



Autres serveurs

## Autres serveurs

---

- Apache est le serveur de référence, néanmoins d'autres possibilités existent, comme :
  - **nginx**, serveur réputé pour ses performances et assez utilisé avec **Ruby on Rails** notamment
  - **lighttpd**, serveur léger fonctionnant en mode CGI ou FastCGI mais ne supportant pas les fichiers **.htaccess**
  - PHP possède son propre serveur HTTP, qui peut être démarré via le shell

## Serveur interne de PHP

---

- Démarrer le serveur

```
> cd /var/www/site_web_exemple  
> php -S localhost:8000
```

Racine du site web à servir

Port sur lequel le serveur écoute les requêtes



## Ce qu'on a couvert

---

- Installer Apache
- Mise en route du serveur
- Les fichiers de configuration
- Autres serveurs HTTP
  - NGINX
  - lighttpd
  - PHP



## Installer un serveur web

---

- MySQL

## Plan

---

- Installer MySQL
- Configuration de MySQL
- Règles de sécurité
- Accéder à la base de données par la ligne de commande
- Autres bases de données
  - MariaDB
  - PostgreSQL



## Installer MySQL

---

- Installer MySQL se fait par la commande **apt-get**

```
apt-get install mysql-server
```

- Il est aussi possible d'installer d'autres ressources

```
apt-get install mysql-client
```

## Configuration de MySQL

---

- Le fichier de configuration se trouve dans le répertoire /etc/mysql

```
nano /etc/mysql/my.cnf
```

- On peut notamment y paramétrer la langue, les jeux de caractères, les buffers, etc.
- Ne pas oublier de relancer le serveur après modification

```
etc/init.d/mysql reload
```

## Questionnaire de sécurité

---

- Si vous voulez sécuriser votre installation, il existe un script qui permet de modifier un certain nombre de caractéristiques

```
mysql_secure_installation
```

## Accéder à MySQL

- Une fois le serveur installé, il est possible de faire tout un tas d'opérations depuis le shell avec la commande `mysql`

```
mysql -u<user> -p<motdepasse>
```

- avec les identifiants de l'utilisateur désiré



# Autres bases de données

Formation PHP MySQL

alphorm.com™©

## MariaDB

---

- MySQL ayant été racheté par Oracle, la communauté a recréé un « fork », une nouvelle branche de développement, entièrement libre, qui s'appelle **MariaDB**
- Les deux bases sont des clones, mais ont des voies de développement différentes depuis leur séparation

## PostgreSQL

---

- PostgreSQL est une base de données libre qui a repris la suite du moteur Ingres, une base de données professionnelle qui a arrêté son développement et libéré le code
- PostgreSQL possède beaucoup d'avantages sur MySQL.
- Son seul défaut est de ne pas être installé par défaut dans les configurations de serveurs web.

## Des formations de BDD sur Alphorm

- <http://www.alphorm.com/formations/base-de-donnees>

|                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                   |                                                                                                                                                                                                                                                                                            |                                                                                                                                                                                                                                                                                                                          |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  <p>Le langage PL/SQL</p> <p>★★★★★(10 votes)<br/>Maîtriser les différents concepts de développement du langage PL/SQL avec les bases de données Oracle</p> <p>88€ Acheter</p> <p>18h11min</p> |  <p>PostgreSQL, la haute disponibilité</p> <p>★★★★★(2 votes)<br/>Apprenez les bonnes méthodes de la mise de la HD sous PostgreSQL</p> <p>49€ Acheter</p> <p>7h27min</p>                          |  <p>Le langage SQL</p> <p>★★★★★(40 votes)<br/>Maîtriser le langage SQL en environnement SGBDR. SQL n'aura plus de secrets pour vous !</p> <p>51€ Acheter</p> <p>12h11min</p>                             |  <p>MySQL (1Z0-883)</p> <p>★★★★★(5 votes)<br/>Apprenez à administrer MySQL d'une manière efficace et sécurisée</p> <p>97€ Acheter</p> <p>24h07min</p>                                                                                 |
|  <p>MongoDB, administration</p> <p>★★★★★(5 votes)<br/>Maîtriser le système de gestion de base de données MongoDB</p> <p>49€ Acheter</p> <p>9h48min</p>                                        |  <p>Oracle Database 11g DBA 1 (1Z0-052)</p> <p>★★★★★(19 votes)<br/>Devenez DBA Oracle et préparez votre certification Oracle Certified Associate 1Z0-052</p> <p>124€ Acheter</p> <p>31h51min</p> |  <p>PostgreSQL, Administration</p> <p>★★★★★(11 votes)<br/>Apprenez les bonnes techniques pour maîtriser et administrer votre serveur de base de données PostgreSQL</p> <p>45€ Acheter</p> <p>9h25min</p> |  <p>SQL Server 2012 (70-462)</p> <p>★★★★★(17 votes)<br/>Apprenez à administrer et maintenir un environnement de base de données Microsoft SQL Server 2012 et préparez votre certification MCSA</p> <p>99€ Acheter</p> <p>21h23min</p> |

## Ce qu'on a couvert

- Installer MySQL
- Configuration de MySQL
- Règles de sécurité
- Accéder à la base de données par la ligne de commande
- Autres bases de données
  - MariaDB
  - PostgreSQL



## Installer un serveur web

---

### • PHP

## Plan

---

- PHP : Quelle version ?
- Installer PHP
- Installer le connecteur `php5-mysql`
- Installer phpMyAdmin
  - Accéder à l'administration des bases de données



## Installer PHP

---

- Installer PHP se fait par la commande **apt-get**

```
apt-get install php5
```

- Il ne faut pas oublier d'installer le connecteur à MySQL

```
apt-get install php5-mysql
```

## phpMyAdmin

---

- Plutôt que d'utiliser la ligne de commande mysql, on installe généralement l'interface d'administration phpMyAdmin

```
apt-get install phpmyadmin
```

- On a intérêt à créer un alias pour utiliser phpmyadmin depuis le web

```
sudo ln -s /usr/share/phpmyadmin /var/www/phpmyadmin
```

## Utilisation de phpMyAdmin

---

- Maintenant que l'outil d'administration est installé, il suffit de l'interroger depuis un navigateur web

`http://<adresse_IP>/phpmyadmin`

- Vous devrez vous identifier comme utilisateur enregistré de la base de données

## Ce qu'on a couvert

---

- PHP : Quelle version ?
- Installer PHP
- Installer le connecteur php5-mysql
- Installer phpMyAdmin
  - Accéder à l'administration des bases de données





## Installer un serveur web

---

- FTP



## Plan

---

- Installer le démon VSFTPD
- Configurer le démon
- Autoriser des utilisateurs à accéder au système de fichiers



## Installation

---

- Le « démon » FTP s'installe via la commande **apt-get**

```
apt-get install vsftpd
```

- VSFTPD est l'acronyme de Very Secure File Transfer Protocol Daemon
- C'est un serveur FTP très bien sécurisé

## Configuration

---

- VSFTPD propose plusieurs mode de paramétrage.
- Comme l'on souhaite généralement plusieurs comptes FTP par domaine, on optera pour le mode « utilisateur virtuel »
- Le fichier à modifier est :

```
nano /etc/vsftpd.conf
```

## Configurer les utilisateurs

---

- Pour chaque utilisateur nous allons devoir définir quels sont ses droit d'accès aux répertoires du serveur.
- Pour cela, nous devons créer un fichier de configuration dans un dossier particulier

```
mkdir /etc/vsftpd/vsftpd_user_conf  
nano /etc/vsftpd/vsftpd_user_conf/<user>
```

- Il est nécessaire de redémarrer le serveur après modification

```
/etc/init.d/vsftpd restart
```

## Ajouter un utilisateur

---

- Pour ajouter un nouvel utilisateur :

```
# db4.8_load -T -t hash -f /etc/vsftpd/login.txt /etc/vsftpd/login.db
```

- Il faut ensuite répéter la configuration de l'utilisateur
- Et enfin relancer le serveur FTP

## chroot

---

- Les utilisateurs sont dits « **chrootés** », c'est-à-dire que le répertoire défini dans la configuration apparaît comme la racine du système de fichiers
- Ainsi, on évite les problèmes de sécurité

## chown

---

- Si vous rencontrez des problèmes pour modifier, effacer, ou créer des fichiers, c'est sans doute que vous n'avez pas les droits.
- Il faut alors vérifier à qui appartiennent les fichiers en question et, au besoin, modifier cela

```
chown -R www-data:www:data ./
```



## Ce qu'on a couvert

---

- Installer le démon VSFTPD
- Configurer le démon
- Autoriser des utilisateurs à accéder au système de fichiers



## Installer un serveur web

---

- Administrer Apache

## Plan

---

- Le fichier httpd.conf
- Le fichier .htaccess
- Lier un nom de domaine





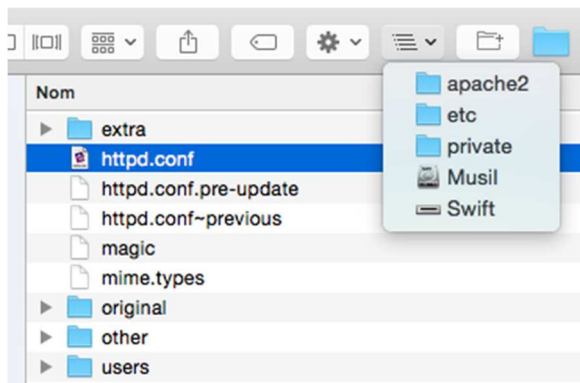
httpd.conf

Formation PHP MySQL

alphorm.com™©

## httpd.conf

- Le fichier **httpd.conf** est le fichier de configuration du serveur Apache.
- Il est souvent situé dans les dossiers :



/etc/apache2/httpd.conf

## httpd.conf

- Le fichier **httpd.conf** a beaucoup d'options pour paramétrer le serveur Apache.
- En particulier :
  - les ports sur lesquels écouter les requêtes (par défaut 80)
  - les répertoires dans lesquels sont installés les sites web sur votre serveur
  - les liaisons avec des noms de domaine

## Port

---

- Le port sur lequel écoute le serveur est modifié avec la directive **Listen**

```
1 | Listen 80
```

- Par défaut, le port pour HTTP est 80, mais vous pouvez changer à votre guise

## Les systèmes de fichiers

---

- Les conteneurs **<Directory>** et **<Files>** permettent d'appliquer des directives à certaines parties du système de fichiers

```
<Directory "/var/web/dir1">  
    Options +Indexes  
</Directory>
```

```
<Directory "/var/web/dir1">  
    <Files "private.html">  
        Require all denied  
    </Files>  
</Directory>
```

## Les directives sur les sites

---

- Le conteneur <Location> permet de configurer spécifiquement certaines parties des sites web

```
<LocationMatch "/private">  
    Require all denied  
</LocationMatch>
```

## Les conditions

---

- Il est possible de tester l'état du système avec les conteneurs <If> <IfDefine> <IfModule> et <IfVersion>
- <IfModule> est très utilisé pour tester les extensions activées du serveur

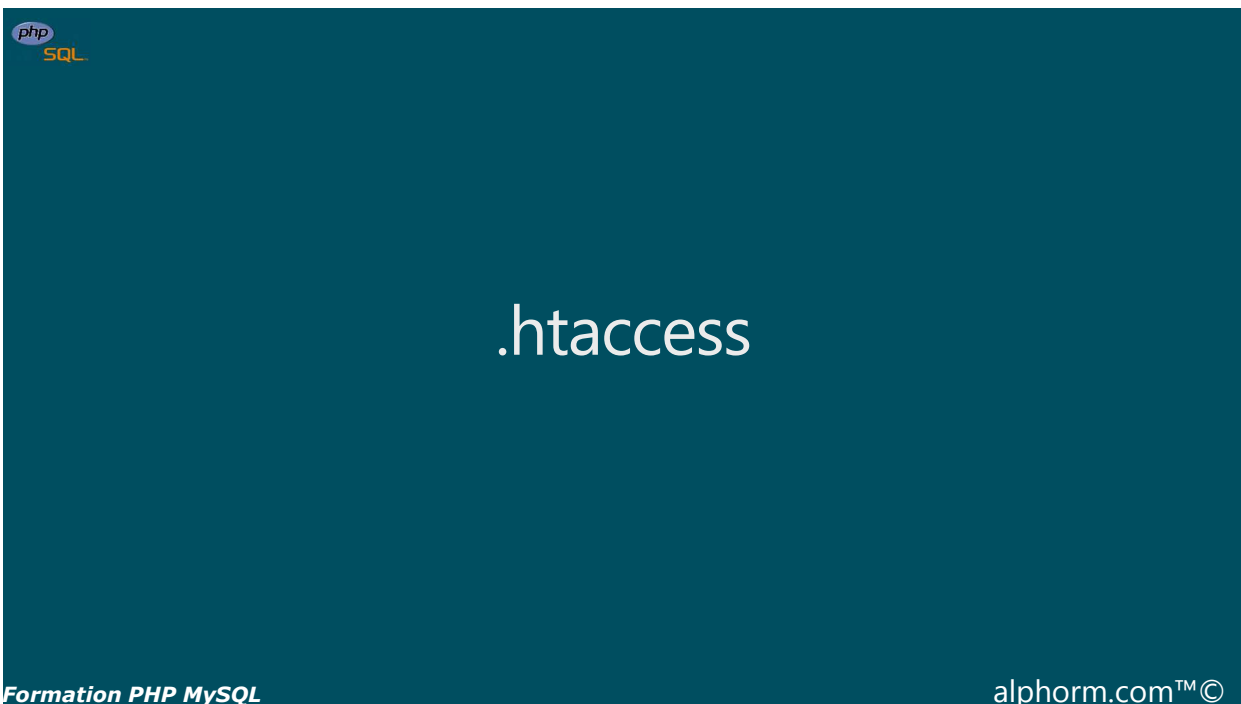
```
<IfModule mime_module>  
    #  
    # TypesConfig points to the file containing the list of mappings from  
    # filename extension to MIME-type.  
    #  
    TypesConfig /private/etc/apache2/mime.types
```

## Les erreurs

---

- httpd.conf permet également de configurer les réponses en cas d'erreur

```
#  
# Customizable error responses come in three flavors:  
# 1) plain text 2) local redirects 3) external redirects  
#  
# Some examples:  
#ErrorDocument 500 "The server made a boo boo."  
#ErrorDocument 404 /missing.html  
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"  
#ErrorDocument 402 http://www.example.com/subscription_info.html  
#
```



.htaccess

## .htaccess

---

- Le fichier **.htaccess** est un complément de configuration de **httpd.conf**, dont il recouvre en partie les possibilités.
- Il est souvent utilisé lorsque vous n'avez pas les droits d'accès sur les fichiers d'accès d'Apache
- Principalement il joue un rôle en matière de :
  - sécurité
  - SEO
  - performance
  - serveur
  - expérience utilisateur (UX)

## Sécurité

---

- Il existe de nombreux moyens pour protéger l'accès au site.
- Par exemple :

```
<Files secure.php>  
AuthType Basic  
AuthName "Prompt"  
AuthUserFile /home/path/.htpasswd  
Require valid-user  
</Files>
```

## SEO

---

- Un certain nombre d'options sont également très utiles, le forçage vers un domaine canonique

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^yourdomain.com$ [NC]
RewriteRule ^(.*)$ http://www.yourdomain.com/$1 [R=301,L]
```

## Performance

---

- L'optimisation des sites passe aussi par des techniques comme l'utilisation :
  - des ETags
  - de la compression des fichiers
  - de la mise en cache
  - de la configuration des types de media disponibles

## Performance

---

- Il est également intéressant de :
  - limiter éventuellement le nombre d'accès simultanés
  - de spécifier les fuseaux horaires du serveur

## Expérience utilisateur

---

- Personnaliser les pages d'erreurs, rediriger en cas de maintenance :

```
#custom error docs
ErrorDocument 404 /introuvable.php
ErrorDocument 403 /nonautorise.php
ErrorDocument 500 /erreur.php
```

```
RewriteEngine on
RewriteCond %{REQUEST_URI} !/maintenance.html$
RewriteCond %{REMOTE_ADDR} !^123.123.123.123
RewriteRule $ /maintenance.html [R=302,L]
```



# Lier un nom de domaine

Formation PHP MySQL alphorm.com™©



## VirtualHost

---

- Le conteneur <VirtualHost> permet de lier une adresse IP + un port à un dossier du système de fichiers.

```
<VirtualHost 192.168.0.1:80>  
DocumentRoot /var/www1  
ServerName server1  
</Virtualhost>
```



## Lier le nom de domaine à l'adresse IP

---

- Une fois défini le point d'entrée du site, il faut encore faire pointer le nom de domaine vers l'adresse IP
- Pour cela, il faut ajouter une entrée **A** dans la **zone DNS** de votre serveur (généralement chez l'hébergeur qui s'occupe de votre nom de domaine)



## Ce qu'on a couvert

---

- Le fichier httpd.conf
- Le fichier .htaccess
- Gérer les utilisateurs et les droits
- Lier un nom de domaine



## Installer un serveur web

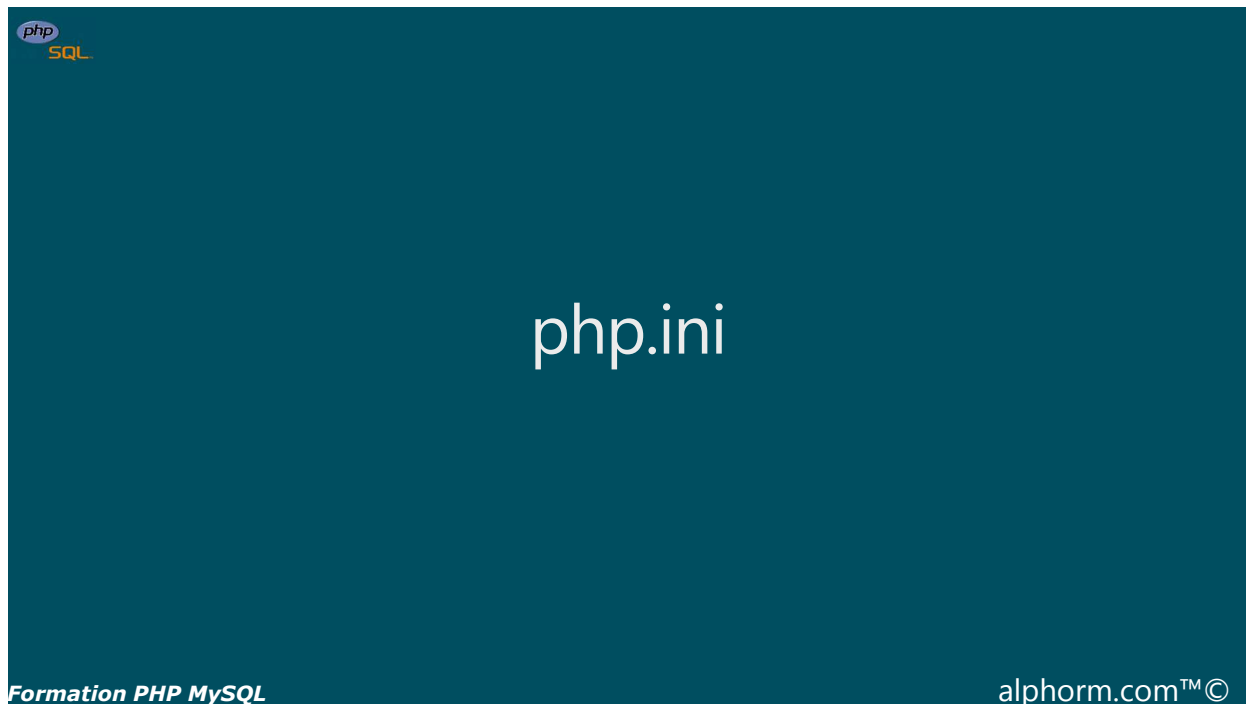
---

- Administrer PHP

## Plan

---

- Le fichier php.ini
- Installer une extension pour PHP
- L'environnement de test PHPUnit



## Introduction

- **php.ini** est le fichier qui détermine la configuration d'exécution de PHP
- Il est souvent logé dans le répertoire :  
**/etc/php5/apache2/php.ini**
- On peut connaître sa localisation exacte via PHP grâce à la fonction **phpinfo()** qui donne tous les détails de l'installation de PHP :

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

## Que faire ?

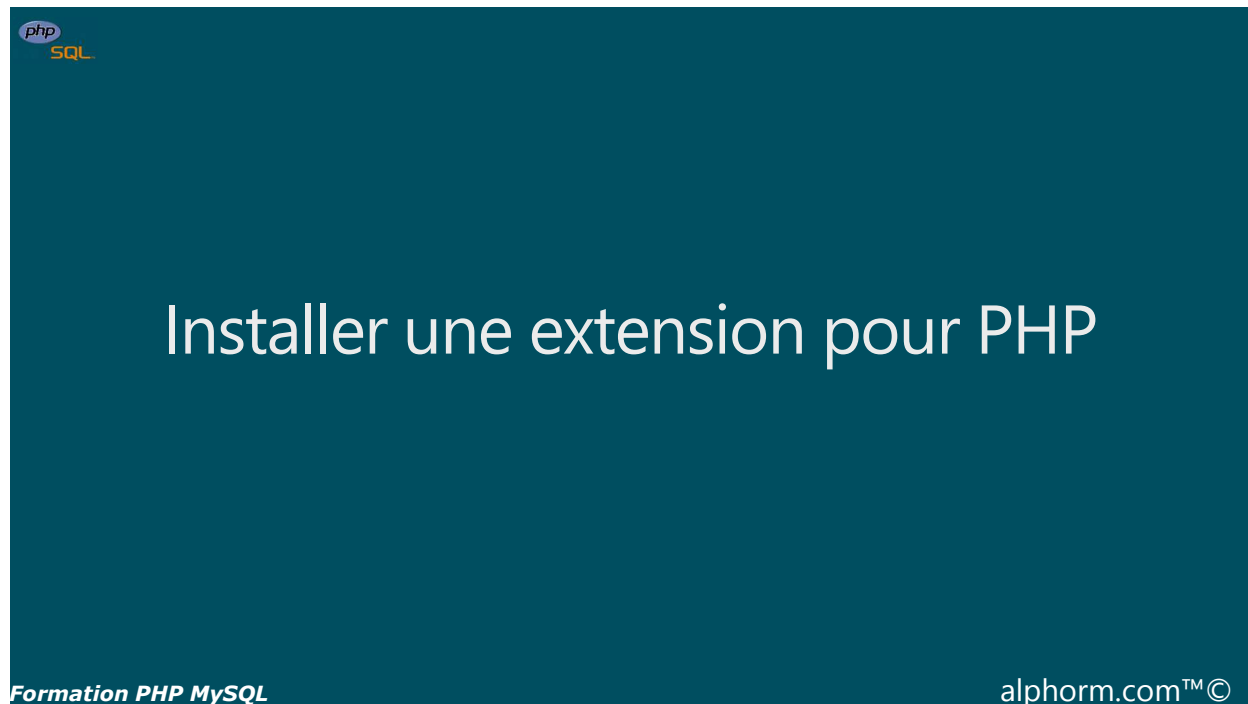
---

- Parmi les modifications les plus courantes qu'il est possible de faire dans le fichier **php.ini** :
  - La configuration du fuseau horaire
  - L'affichage et l'archivage (log) des erreurs
  - La restriction de certaines fonctions sensibles
  - La taille maximale des fichiers en upload
  - Le temps maximal d'exécution d'un script
  - Les répertoires d'inclusion

## Sécurité

---

- La configuration de **php.ini** peut avoir une grande importance sur la sécurité des applications.
- Notamment :
  - **allow\_url\_fopen**
  - **expose\_php**



## Un exemple Imagick

- ImageMagick est une application très répandue de traitement d'image, supérieure et plus facile à utiliser que la bibliothèque gd installée par défaut.
- Pour activer la bibliothèque dans PHP5, il faut installer le module **imagick** correspondant.

```
> apt-get install imagemagick  
> apt-get install php5-imagick
```

- Et relancer le serveur Apache

```
> service apache2 restart
```



- Il est aussi possible de passer par la commande **pecl**, qui accède directement au dépôt PEAR des modules PHP

```
> pecl install memcache
```

- ... et relancer le serveur



PHPUnit

## Introduction

---

- PHPUnit est un outil indispensable pour le développement d'applications PHP
- C'est un logiciel autonome qui permet de tester la validité du code écrit
- Les tests sont une composante essentielle du travail de développement
- Le fonctionnement de PHPUnit sera abordé plus tard dans le cours

## Installation

---

- Pour installer PHPUnit, il suffit de télécharger le logiciel via la ligne de commande :

```
> wget https://phar.phpunit.de/phpunit.phar
```

- Puis le rendre exécutable `> chmod +x phpunit.phar`
- Et le déplacer dans un répertoire référencé dans \$PATH

```
> mv phpunit.phar /usr/local/bin/phpunit
```

## Utilisation

---

- Une fois installé, PHPUnit s'utilise par la ligne de commande. Par exemple :

```
> phpunit —version
```

- PHPUnit attend un code structuré de manière à pouvoir le reconnaître automatiquement.

## Ce qu'on a couvert

---

- Le fichier php.ini
- Installer une extension pour PHP
- L'environnement de tests PHPUnit



## Installer un serveur web

---

- Premiers pas avec les serveurs LAMP

## Plan

---

- Utiliser un éditeur de code
- Se connecter au serveur avec FTP
- Une page HTML
- Un script PHP
- Vérifier l'état de la base de données



## Un éditeur de code

- Pour se faciliter la tâche, il est nécessaire de choisir un bon éditeur.
- Pour les plus techniciens, il existe des outils comme **vim**, **emacs**, **nano** qui sont le plus souvent accessibles par la ligne de commande
- On a également souvent besoin d'éditeurs élaborés (avec coloration syntaxique, auto-complétion, etc.).
  - Voir alors : **Atom**, **Eclipse**, **Sublime Text**, **NetBeans**

## Connexion au serveur FTP

- Grâce à des outils libres comme **FileZilla**, nous allons pouvoir travailler et échanger des fichiers avec le serveur web



## Une page HTML

---

- Pour commencer, nous allons écrire une simple page HTML que nous allons ensuite déposer sur le serveur
- Nous ferons ensuite un lien vers une autre page HTML locale située dans un sous-répertoire de notre hébergement
- Ainsi, nous verrons une première ébauche de ce qui deviendra une politique d'URL
- Les pages HTML sont reconnues par le serveur grâce au suffixe `.html` (et quelques autres)
- La liste des suffixes peut être modifiée dans la configuration d'Apache

## Un script PHP

---

- Maintenant que nous avons une page statique, nous allons aller un peu plus loin en écrivant un script PHP
- Les scripts PHP sont reconnus par le serveur grâce à leur suffixe `.php` (et quelques autres, mais rarement utilisés)
- Le script PHP va être simplement chargé de décorer un contenu HTML avec des éléments génériques, comme un entête ou un pied de page, de feuilles de style

## Tester MySQL

---

- Pour terminer ce premier tour d'horizon de la plate-forme LAMP, nous allons vérifier que PHP est bien capable de dialoguer avec MySQL
- Pour cela, nous allons simplement tenter de nous connecter au serveur de base de données avec la fonction `mysqli_connect()`

## Ce qu'on a couvert

---

- Utiliser un éditeur de code
- Se connecter au serveur avec FTP
- Une page HTML
- Un script PHP
- Vérifier l'état de la base de données





## • Installer XAMPP



- Installer XAMPP
- Qu'est-ce que XAMPP ?
- Exploration rapide



## Télécharger l'application

- XAMPP est une application que l'on peut télécharger en version française ici

<https://www.apachefriends.org/fr/index.html>



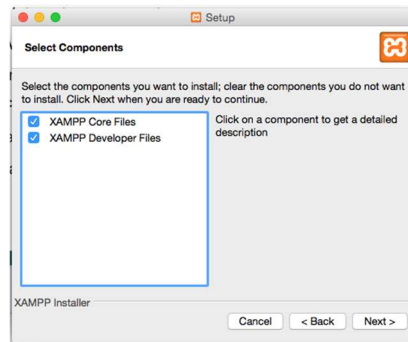
## Préparation

- Sous Linux, vous aurez sans doute à rendre le fichier `.run` exécutable en cochant la case dans les propriétés du fichier
- Pour l'installer, il faudra être **super-administrateur** du système et lancer la commande depuis la ligne de commande suivante dans le répertoire Téléchargements :

```
sudo ./<nom-du-fichier-téléchargé>.run
```

## Installation

- Fenêtre d'installation : Choix des fichiers à installer



- Sous Linux, l'application est hébergée dans le dossier :

**/opt/lampp**

## Qu'est-ce que XAMPP ?

- XAMPP est une plat-forme permettant d'exécuter un environnement complet de serveur Internet, comprenant :
  - un serveur web **Apache**
  - une base de données **MariaDB** (remplaçant MySQL depuis 2015)
  - un serveur FTP **ProFTPD**
  - un serveur de mails
  - PHP (et PHP 7 dans la dernière version)
  - Perl

## Première page

---

- Une fois l'environnement démarré, la page d'accueil de votre hébergement web local est à l'URL :

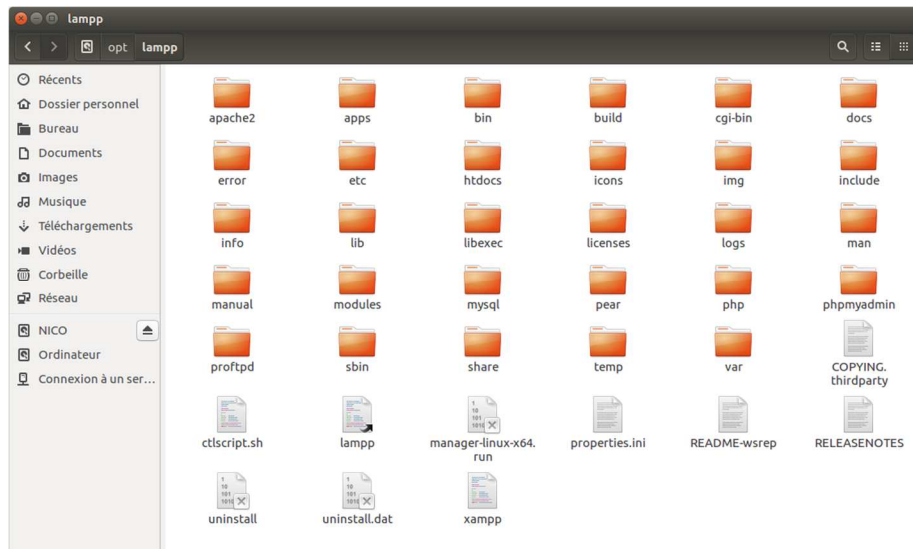
`http://localhost`

## Organisation de l'installation

---

- Sous Linux, l'installateur a créé un dossier **lampp** à l'intérieur du dossier **opt** qui est lui-même à la racine du système de fichiers
- Sous OS X, tous les dossiers sont encapsulés dans **XAMPP.app** qui est en fait un dossier « **caché** »

## Système de dossiers



## Exploration

- Dans les répertoires **lampp**, on remarque :
  - des dossiers liés aux applications du serveur : *php*, *proftpd*, *mysql*, *etc*.
  - des dossiers d'exécutables et de bibliothèques diverses
  - un dossier *cgi-bin* pour les scripts en mode CGI
  - un dossier *apps* pour les outils web supportés par XAMPP
  - des dossiers de documentation *docs*, *man*, *manual*



## Ce qu'on a couvert

---

- Installer XAMPP
- Qu'est-ce que XAMPP ?
- Exploration rapide



## XAMPP

---

- Configurer Apache

## Plan

---

- Trouver la configuration
- Changer le répertoire racine par défaut
- Modifier le fichier httpd.conf
- Créer un fichier .htaccess



## Trouver la configuration

---

- Pour ce qui concerne le serveur HTTP (Apache), le répertoire racine des pages HTML se trouve ici :

`/opt/lampp/htdocs`

- Et le fichier de configuration se trouve dans le dossier :

`/opt/lampp/apache2/conf`

## La page par défaut

- Le dossier htdocs contient en fait principalement une page PHP **index.php**, dont le rôle est de faire une simple redirection vers le dossier **dashboard**
- C'est la page qui s'affiche lorsque vous accédez à l'adresse IP du serveur (par ex. <http://127.0.0.1>)
- Si vous voulez gérer plusieurs sites web, il suffit d'ajouter de nouveaux dossiers dans htdocs

```
sudo adduser <identifiant> <groupe>
```

```
sudo adduser <identifiant>
```

## Le fichier de configuration httpd.conf

- Le fichier **httpd.conf**, comme tout serveur Apache traditionnel permet une grande variété d'options, entre autres :
  - La récupération de fichiers de configuration « locaux »
- La définition d'alias pour les routes des sites
- La définition de droits d'accès pour les différents répertoires

```
Include /opt/lampp/apps/<config>
```

```
Alias /cours-xampp /opt/lampp/htdocs/cours/xampp
```

```
Allow from all
```

## Le fichier .htaccess

---

- En dehors du fichier **httpd.conf**, il est conseillé de créer pour chaque dossier correspondant à un site ou une application un fichier **.htaccess** qui vient renforcer la configuration et la protection
- Les fichiers **.htaccess** sont plus accessibles que les **httpd.conf**
- On vient notamment y mettre tout ce qui a trait à la gestion de la compression, du cache et de l'optimisation du service des réponses à l'utilisateur

## Ce qu'on a couvert

---

- Trouver la configuration
- Changer le répertoire racine par défaut
- Modifier le fichier **httpd.conf**
- Créer un fichier **.htaccess**





## • Configurer ProFTPD



## Plan

- Trouver la configuration
- Créer un utilisateur FTP
  - Créer un nouvel utilisateur Linux
  - Rediriger l'utilisateur



## Créer un utilisateur

---

- Pour créer un utilisateur système, on utilise la commande **adduser** :

```
sudo adduser <identifiant>
```

- Il faut ensuite associer ce nouvel utilisateur à un groupe :

```
sudo adduser <identifiant> <groupe>
```

## Associer l'utilisateur à un répertoire

---

- Il faut maintenant modifier le fichier **proftpd.conf** pour limiter l'accès de l'utilisateur à une partie du système :

```
DefaultRoot /opt/lampp/htdocs <identifiant>
```

## Utilisation

---

- On peut maintenant utiliser un client ftp comme Filezilla ou encore Transmit (OS X) pour vérifier la configuration.
- L'adresse du serveur sera **127.0.0.1** si vous êtes sur la même machine ou **192.168.x.y** si vous êtes sur une autre machine du réseau local

## Ce qu'on a couvert

---

- Trouver la configuration
- Créer un utilisateur FTP
  - Créer un nouvel utilisateur Linux
  - Rediriger l'utilisateur





## •Éléments de sécurité



- Introduction
- Sécuriser
  - l'accès au répertoire de pages web
  - l'accès à la base de données
- phpMyAdmin
- le FTP



## Introduction

---

- XAMPP met à votre disposition un script pour paramétrer la sécurité de votre installation
- Pour cela, il faut passer par la ligne de commande (ouvrir le terminal) :

```
sudo /opt/lampp/lampp security
```

sous Linux

## Sécurisation des pages web

---

- A priori, votre machine est à l'intérieur d'un réseau local (derrière votre box, en particulier) mais si vous avez ouvert un port pour être accessible depuis Internet, ou si vous souhaitez protéger votre installation d'autres machines de votre réseau

```
XAMPP: Your XAMPP pages are NOT secured by a password.  
XAMPP: Do you want to set a password? [yes]
```

- L'installation est alors protégée par un fichier .htaccess



## Protection de la base de données

- De la même manière, il vaut mieux faire en sorte que la base de données ne soit pas accessible.

```
XAMPP: MySQL is accessible via network.  
XAMPP: Normally that's not recommended. Do you want me to turn it off? [yes]
```

- Et lui attribuer un mot de passe

```
XAMPP: MySQL has no root password set!!!  
XAMPP: Do you want to set a password? [yes]
```



## Protection du serveur FTP

- On protégera également le serveur FTP de manière à ce que n'importe qui ne puisse pas modifier les fichiers

```
The FTP password is still set to ####.  
XAMPP: Do you want to change the password? [yes]
```

## Protection phpMyAdmin

---

- Enfin, il sera également nécessaire de restreindre l'accès à l'interface d'administration de vos données

```
XAMPP: The MySQL/phpMyAdmin user pma has no password set!!!  
XAMPP: Do you want to set a password? [yes]
```

## Ce qu'on a couvert

---

- Introduction
- Sécuriser
  - l'accès au répertoire de pages web
  - l'accès à la base de données
- phpMyAdmin
- le FTP





## • Installer un module



## Plan

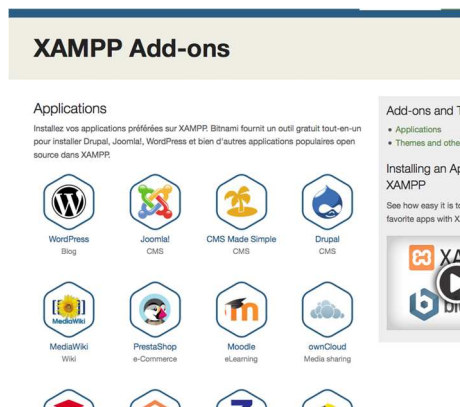
- Télécharger un module pour XAMPP
- Installer le module
  - Un exemple avec MediaWiki
- Vérifier l'installation et la structure des fichiers
- Mettre en œuvre le module



## Télécharger un module

- XAMPP est accompagné de toute une série d'extensions (ou modules) qui permettent d'utiliser des outils web très courants, comme des CMS, des plates-formes de commerce en ligne, etc.

- WordPress
- Drupal
- Prestashop
- Moodle
- MediaWiki
- etc.



## Un module

- Chaque module est en fait un installeur qu'il suffit de télécharger

### MediaWiki

MediaWiki is a wiki package originally written for Wikipedia. MediaWiki is an extremely powerful, scalable software and a feature-rich wiki implementation.



| Version                   | Platform     | Size  | Checksum             |                          |             |
|---------------------------|--------------|-------|----------------------|--------------------------|-------------|
| MediaWiki Module 1.26.2-2 | Windows      | 32 MB | <a href="#">show</a> | <a href="#">Download</a> | Recommended |
| MediaWiki Module 1.26.2-2 | Linux 64-bit | 31 MB | <a href="#">show</a> | <a href="#">Download</a> |             |
| MediaWiki Module 1.26.2-2 | Linux        | 29 MB | <a href="#">show</a> | <a href="#">Download</a> |             |
| MediaWiki Module 1.26.2-2 | OS X 64-bit  | 33 MB | <a href="#">show</a> | <a href="#">Download</a> | Recommended |

## Installer un module

- Installer le module consiste simplement à exécuter le programme que vous avez téléchargé



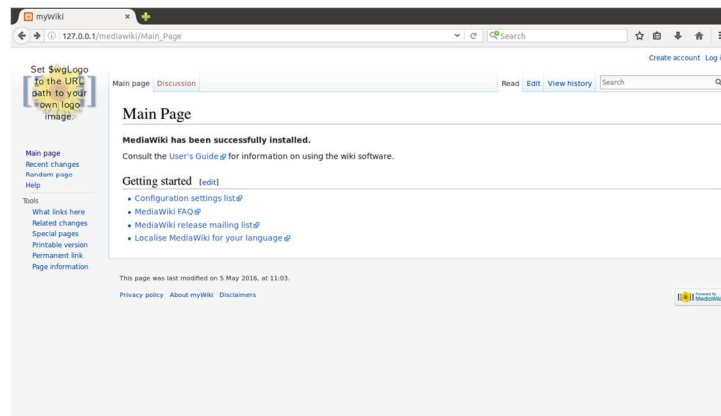
## Vérifier l'installation

- Une fois l'installation finie, on peut vérifier :
  - que les fichiers existent bien dans les répertoires
  - que la base de données a bien été créée
  - que l'application est bien accessible

## Lancer l'application

- Une fois le processus terminé, lancer l'application du module à partir de la racine des applications, ici :

127.0.0.1/mediawiki



## Ce qu'on a couvert

- Télécharger un module pour XAMPP
- Installer le module
  - Un exemple avec WordPress
- Vérifier l'installation et la structure des fichiers
- Mettre en œuvre le module



## XAMPP

---

- Premiers pas avec XAMPP

## Plan

---

- Trouver la racine du site
- Une première page HTML
- Un premier script PHP



## La racine du site

---

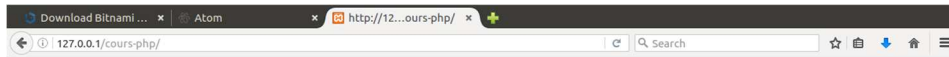
- Sous Linux répertoire racine se trouve dans :  
`/opt/lampp/htdocs`
- Sous OS X :  
`XAMPP/opt/xamppfiles/htdocs`
- A partir de là, nous pouvons créer les répertoires que nous voulons pour héberger des sites différents (et éventuellement créer des hôtes virtuels)

## Une première page HTML

---

- Après avoir créé un sous-répertoire **cours-php** dans htdocs, nous allons créer une page **index.html** dans ce sous-répertoire.
  - **index.html** fait partie des pages cherchées automatiquement par le serveur web lorsque l'on n'indique aucune ressource

`http://127.0.0.1/cours-php/`



Salut tout le monde

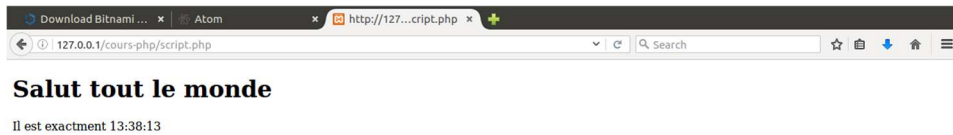
Firefox automatically sends some data to Mozilla so that we can improve your experience. Choose What I Share



## Un premier script PHP

- De la même manière, il est possible de créer un script dynamique
  - **index.php** fait partie des pages cherchées automatiquement par le serveur web lorsque l'on n'indique aucune ressource

<http://127.0.0.1/cours-php/index.php>



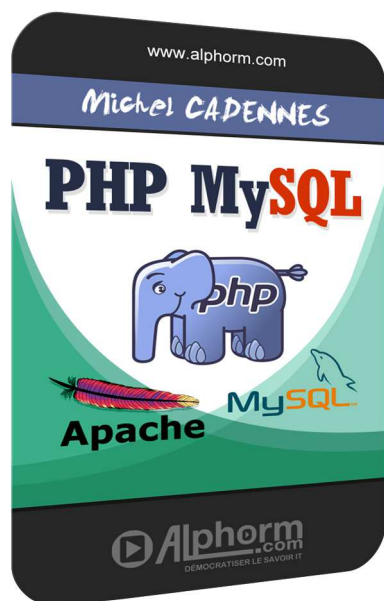
Firefox automatically sends some data to Mozilla so that we can improve your experience. Choose What I Share x



## **Ce qu'on a couvert**

- Trouver la racine du site
- Une première page HTML
- Un premier script PHP





**Alphorm**.com

## Découverte de PHP

### Qu'est-ce que PHP ?

Site : <http://www.alphorm.com>

Blog : <http://blog.alphorm.com>



**Michel CADENNES**

Formateur et Consultant indépendant  
Web, Gestion des connaissances

Formation PHP MySQL

alphorm.com™ ©

## Plan

- L'historique de PHP
- Comment insérer un fragment de code PHP dans une page web
- Afficher un message à l'écran
- Introduire une variable
- Séparer le squelette HTML du programme PHP



Formation PHP MySQL

alphorm.com™ ©

## Histoire de PHP

---

- **1994** : Création de « Personal Home Page Tools » par **Rasmus Lerdorf**
  - Routines en C pour la gestion de statistiques sur des pages web
- **1998** : Publication de PHP 3.0 par **Andi Gutmans** et **Zeev Suraski**
  - Refonte de la syntaxe, premières implémentations objet, facilité d'extension du langage
- **2000** : PHP 4.0
- **2004** : PHP 5.0
  - Extension du modèle objet de PHP
- **2009 – 2013** : PHP 5.3 à 5.6
  - Nombreuses additions fonctionnelles au langage
- **2015** : PHP 7
  - Publication du projet **HHVM** par [Facebook](#)

## PHP et HTML

---

- L'originalité de PHP (comme ASP) est d'insérer le code des scripts à l'intérieur du code HTML des pages web.
- Pour cela on utilise des balises spéciales :

```
1  
2 |<?php /* code PHP */ ?>  
3
```

## Premier exemple

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>Bonjour tout le monde !</title>
6 </head>
7 <body>
8   <h1>Bonjour tout le monde !</h1>
9   <?php echo "Ceci est affiché par PHP" ?>
10 </body>
11 </html>
```

Insertion d'un script PHP dans  
une page web

## Séquence d'instructions

```
1 <!DOCTYPE html>
2 <?php $x = 10; ?>
3 <html>
4 <head>
5   <meta charset="utf-8">
6   <title>Bonjour tout le monde !</title>
7 </head>
8 <body>
9   <?php $y = 20; ?>
10  <h1>Bonjour tout le monde !</h1>
11  <?php echo $x + $y ?>
12 </body>
13 </html>
```

La séquence des instructions  
PHP est indépendante de sa  
fragmentation dans la page.  
Les différents éléments sont  
considérés comme un script.

## Premiers éléments syntaxiques

```
1 <!DOCTYPE html>
2 <?php $x = 10; ?>
3 <html>
4 <head>
5 <meta charset="utf-8">
6 <title>Bonjour tout le monde !</title>
7 </head>
8 <body>
9 <?php $y = 20; ?
10 <h1>Bonjour tout le monde !</h1>
11 <?php echo $x + $y ?>
12 </body>
13 </html>
```

Les variables sont signalées  
par le caractère préfixe \$

Le séparateur d'instructions est  
le caractère ;  
Il est obligatoire sauf pour la  
dernière instruction (ici avant la  
fermeture de la balise)

## Modularisation du code

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Bonjour tout le monde !</title>
6 </head>
7 <body>
8 <h1>Bonjour tout le monde !</h1>
9 <?php include "script.php" ?>
10 </body>
11 </html>
```

En général, le code PHP est  
placé dans des fichiers  
externes qui sont appelés par  
la directive include  
(des versions plus élaborées  
des imports existent aussi)

## script.php

---

```
1 <?php
2
3 $x = 10;
4 $y = 20;
5
6 echo $x + $y;
```

Pour les fichiers qui ne contiennent que du code PHP, la convention est de ne pas fermer la balise de script.

## Ce qui a été couvert

---

- L'historique de PHP
- Comment insérer un fragment de code PHP dans une page web
- Afficher un message à l'écran
- Introduire une variable
- Séparer le squelette HTML du programme PHP



## Découverte de PHP

---

- Les types de base en PHP

## Plan

---

- Le système de types de base de PHP
  - les types simples ou scalaires
  - les types composés
  - les types spéciaux
- Comment convertir une variable d'un type en un autre



## Les types

---

- PHP possède un système de types assez simples, avec :
  - 4 types scalaires : integer, float, string, boolean
  - 2 types composés : array, object
  - 2 types spéciaux : resource, null



# Les Types scalaires

Formation PHP MySQL

alphorm.com™©

## Les booléens

- Une variable booléenne peut prendre deux valeurs : **true** (vrai) et **false** (faux)
- La valeur **false** s'exprime elle-même de plusieurs manières différentes :
  - 0 (entier)
  - 0.0 (décimal)
  - La chaîne vide et la chaîne '0'
  - Un tableau vide
  - NULL

## Les entiers

- Les entiers peuvent être exprimés en PHP dans plusieurs bases différentes : **décimale** (bien sûr) mais aussi **binaire**, **octal**, **hexadécimal**

```
1 <?php
2
3 $a = 1234; // un nombre décimal
4 $a = -123; // un nombre négatif
5 $a = 0123; // un nombre octal (équivalent à 83 en décimal)
6 $a = 0x1A; // un nombre hexadécimal (équivalent à 26 en décimal)
7 $a = 0b11111111; // un nombre binaire (équivalent à 255 en decimal)
8
9 ?>
```

## Les entiers

---

- Sur les plates-formes 64 bits, la valeur maximale d'un entier est de l'ordre de **9e+18** et elle peut être connue par la constante **PHP\_INT\_MAX**

## Les nombres à virgule flottante

---

- PHP ne connaît qu'un seul type de nombres réels qui sont le type **float**
- Les nombres entiers dépassant la limite sont automatiquement transtypés en float
- La constante **NAN** sert à représenter les valeurs indéfinies dans le cadre d'opérations numériques sur les réels.

## Les chaînes de caractères

---

- Les chaînes de caractères peuvent s'exprimer sous quatre formes :
  1. entourées par des guillemets simples
  2. entourées par des guillemets doubles
  3. avec la syntaxe **Heredoc**
  4. avec la syntaxe **Nowdoc** (depuis PHP 5.3)

## Les guillemets simples

---

- Les guillemets simples permettent de définir une chaîne de caractères entièrement statique, donc aucun élément n'est interprété.
  - Néanmoins, pour insérer un guillemet simple à l'intérieur d'une chaîne de caractères statique, il faut le protéger avec le caractère `\` (antislash)

## Les guillemets doubles

- Une chaîne de caractères comprise à l'intérieur de guillemets doubles peut contenir des éléments variables qui seront interprétés avant d'être affichés.

```
1
2 $x = 'la variable';
3 "Dans cette chaîne, $x sera remplacée par sa valeur";
```

- Ces chaînes reconnaissent également un certain nombre de caractères spéciaux, protégés par l'antislash

## Heredoc

- La syntaxe Heredoc offre les mêmes caractéristiques fonctionnelles que les guillemets doubles, mais sans recourir aux caractères spéciaux
  - Elle s'ouvre par la séquence <<<

```
1
2 $chaine : <<<F00
3 Ceci est une
4 chaîne disposée sur
5 trois lignes.
6 F00;
```

## Nowdoc

- Nowdoc est aux chaînes statiques (guillemets simples) ce que Heredoc est aux chaînes dynamiques. La différence syntaxique est que le label doit être lui-même entouré de guillemets simples.

```
2 $chaine : <<<'F00'  
3 Ceci est une  
4 chaîne disposée sur  
5 trois lignes.  
6 F00;
```



## Le types composés

## Les tableaux

- Un tableau est une variable qui, au lieu de contenir une valeur simple, contient une liste ordonnée de valeurs.
  - Les valeurs peuvent être ordonnées selon un index numérique (celui-ci est généralement implicite)
  - Les valeurs peuvent être associées à des mot-clefs, on parle alors de tableau associatif.
  - Les tableaux n'ont pas de dimensions, toute valeur peut être un tableau, ce qui conduit à la représentation d'arbres

## Création de tableau

- Un tableau peut se créer avec deux syntaxes :

```
1 <?php
2
3 $t = array('a', 'b', 'c') // syntaxe historique
4 $t2 = ['a', 'b', 'c'] // depuis PHP 5.4
5
6 $t3 = ['a' => 1, 'b' => 2, 'c' => 3] //tableau associatif
```

## Modifier et interroger un tableau

- Pour modifier ou consulter un élément du tableau, il suffit d'indiquer son index entre crochets :

```
8  $t[] = 'd'; // tableau simple
9  $t3['d'] = 4; // tableau associatif
10
11  $x = $t[2];
12  $y = $t3['a'];
```

## Les tableaux

- Il existe de nombreuses fonctions en PHP pour travailler sur les tableaux
- Par ailleurs la bibliothèque standard de PHP (SPL) fournit de nombreux outils plus spécifiques pour représenter des listes, des arbres, des files d'attentes, etc.

## Les objets

---

- PHP fournit depuis la version 5 une représentation objet extensive.
- Le modèle objet de PHP se base sur le modèle classe/instance
- Une formation sera consacrée à l'utilisation des objets et des classes en PHP



## les types spéciaux

## Les ressources

---

- Une ressource est une variable spéciale, contenant une référence vers une ressource externe. Les ressources sont créées et utilisées par des fonctions spéciales.
- Les ressources représentent généralement des fichiers, des flux (streams), des connexions à des bases de données, des images, etc.

## NULL

---

- La valeur spéciale **NULL** représente une variable sans valeur. **NULL** est la seule valeur possible du type **NULL**.
- Une variable est considérée comme **NULL** si :
  - elle s'est vue assigner la constante **NULL**.
  - elle n'a pas encore reçu de valeur.
  - elle a été effacée avec la fonction **unset()**

## Manipulation de types

- Il est possible de convertir, sous certaines conditions, une variable d'un type dans un autre. c'est l'opération de transtypage (cast).

```
<?php
$foo = 10;           // $foo est un entier
$bar = (boolean) $foo; // $bar est un booléen
?>
```

## Ce qu'on a couvert

- Le système de types de base de PHP
  - les types simples ou scalaires
  - les types composés
  - les types spéciaux
- Comment convertir une variable d'un type en un autre



## Découverte de PHP

---

- Les variables

## Plan

---

- Caractéristiques des variables
- Les variables pré-définies
- Qu'est-ce que la portée ?
- Qu'est-ce qu'un bloc ?
- Les variables 'global'
- Et pour les imports ?
- Les variables dynamiques



## Les variables

---

- En PHP, les variables sont reconnues par le caractère \$
- Les noms des variables ne peuvent commencer que par une lettre ou un '\_' (souligné)
  - Ils ne peuvent contenir que des caractères ou des chiffres
  - Les conventions sont que les noms de variables comment par une lettre minuscules.

## Variables pré-définies

---

- PHP possède un certain nombre de variables pré-définies, dont :
  - les variables dites « **super-globales** »
  - **\$argc** et **\$argv** qui contiennent les arguments passés au script PHP lorsque celui-ci est exécuté en ligne de commande



## La portée

---

- En informatique, la portée est l'espace du code au sein duquel un symbole (souvent une variable) est défini, accessible et lié à une valeur.
- Cette étendue est inséparable de la notion de bloc.
- La portée peut être définie de deux manières, selon les langages :
  - **lexicale** : la portée est définie par la structure du code source
  - **dynamique** : la portée est définie par l'état du programme au moment de son exécution

## Les blocs

---

- Un bloc est une séquence d'instructions délimitée, éventuellement récursive.
  - Dans beaucoup de langages, les blocs sont repérés par des accolades. Ainsi, définissent notamment des blocs :
    - les fonctions
    - les conditionnelles
    - les boucles

## Les portées en PHP

---

- PHP n'admet qu'une seule déclaration de symboles (variables, fonctions, etc.)
- Il n'y a donc que deux portées possibles :
  - le bloc où est définie la variable
  - la totalité du programme

## Le mot clef 'global'

- Les variables globales sont une mauvaise méthode de programmation et on cherchera à les éviter au maximum. Pour faire référence à une variable globale, on utilise le mot-clef : 'global' :

```
1 <?php
2
3 $a = 20;
4
5 function f () {
6     return $a ** 2;
7 }
8
9 echo f();
10 // Avertissement E_NOTICE : la variable $a n'est pas définie dans la portée de f
11
12 function g () {
13     global $a;
14     return $a ** 2;
15 }
16
17 echo g();
18 // 40
```

## Les variables dynamiques

## Les variables dynamiques

- En PHP, les symboles peuvent être variables.
- Il est tout à fait licite d'écrire :

```
1 <?php
2
3 $a = 'x';
4 $x = 50;
5 echo $$a; // Indirection affichant la valeur de $x;
6
7 function hello () {
8     return "Hello !";
9 };
10 $f = 'hello';
11 echo $hello(); // Nom de fonction dynamique
```

## Ce qu'on a couvert

- Caractéristiques des variables
- Les variables pré-définies
- Qu'est-ce que la portée ?
- Qu'est-ce qu'un bloc ?
- Les variables 'global'
- Et pour les imports ?
- Les variables dynamiques



## Découverte de PHP

---

- Les structures de contrôle

## Plan

---

- La conditionnelle : if
- La conditionnelle multiple : switch
- Les boucles itératives : for, while, foreach
- Les interruptions d'itération : break, continue
- Les imports : include, require





- **if** est une des instructions fondamentales des langages de programmation car elle permet de n'exécuter du code que si quelque chose (une expression) est **vraie**.

```
1 <?php
2
3 $somme = 4 + 8;
4
5 if ($somme > 10) {
6     echo $somme;
7 }
```

Expression conditionnant  
l'exécution du code à  
l'intérieur des accolades.

## if ... else

- Si l'expression conditionnelle est fausse, on peut indiquer le code à exécuter alternativement avec **else**.

```
1 <?php
2
3 $somme = 4 + 8;
4
5 if ($somme > 10) {
6     echo $somme;
7 } else {
8     echo "Caramba ! Encore raté...";
9 }
```

Sinon...

## if ... else if ... else

- PHP accepte également la construction **elseif**, mais on lui préférera en général l'instruction **switch**, plus lisible.

```
1 <?php
2
3 $somme = 4 + 8;
4
5 if ($somme > 10) {
6     echo $somme;
7 } else {
8     echo "Caramba ! Encore raté...";
9 }
```

Sinon...

## switch

- switch est une instruction qui permet de tester différentes valeurs prises par une expression

```
1 <?php
2
3 if ($reponse) {
4     case "oui":
5         echo "accord";
6         break;
7     case "non":
8         echo "désaccord";
9         break;
10    case "abstention":
11        echo "ne choisit pas";
12        break;
13    default:
14        echo "pas d'avis";
15 }
```

Fin de traitement  
du cas →

Valeur attendue →

La valeur n'est pas parmi  
celles qui sont attendues →

## les structures itératives

## while

- while permet de répéter une séquence d'instructions tant que la valeur d'une expression est **vraie**

```
1 <?php
2
3 $i = 0;
4 while ($i <= 10) {
5     echo "Le carré de $i vaut : " . ($i*$i);
6     $i += 1;
7 }
```

## foreach ... as

- foreach est une structure itérative qui permet de parcourir un objet « itérable » (i.e. Traversable) comme, typiquement, un tableau

```
1 <?php
2
3 $t = [1,4,12,15,29,45,234,52358,1234589];
4 foreach ($t as $nombre) {
5     if ($nombre % 2 == 0) {
6         echo "$nombre est pair";
7     }
8 }
```

\$nombre est la variable de boucle

## do ... while

- Alternativement à while, il est possible d'utiliser **do ... while** pour que le test de validité soit effectué à la fin de l'itération (au moins une exécution garantie)

```
1 <?php
2
3 $i = 0;
4 do {
5     echo "Le carré de $i vaut : " . ($i*$i);
6     $i += 1;
7 } while ($i <= 10);
```

## for

- Il est toujours possible d'utiliser la boucle **for** traditionnelle, même si sa syntaxe lourde fait qu'elle n'est pas privilégiée

```
1 <?php
2
3 for ($i=0; $i <=10 ; $i++) {
4     echo "Le carré de $i vaut : " . ($i*$i);
5     $i += 1;
6 }
```



## break

- **break** est une instruction qui permet d'interrompre l'exécution d'une ou plusieurs boucles imbriquées.
- L'argument numérique indique combien de niveaux d'imbrication doivent être court-circuités.

```
1 <?php
2
3 while (true) {
4     while (true) {
5         while (true) {
6             break 3;
7         }
8     }
9 }
```

Les trois boucles  
infinies sont  
interrompues

## continue

- **continue** est une variante de **break**, permettant de sauter directement à la fin de l'itération courante.
- Là aussi, un nombre d'imbrications peut être précisé.

```
1 <?php
2
3 $i = 0;
4 while ($i <= 10) {
5     echo $i;
6     if ($i != 5) continue;
7     echo ("Cinq !");
8 }
```

← Cette ligne n'est exécutée que si i = 5

## Les imports

## include (et include\_once)

---

- L'instruction de langage **include** inclut et exécute le fichier spécifié en argument.
  - Le code le composant hérite de la portée des variables de la ligne où l'inclusion apparaît.
  - Toutes les variables disponibles à cette ligne dans le fichier appelant seront disponibles dans le fichier appelé, à partir de ce point.
  - Cependant, toutes les fonctions et classes définies dans le fichier inclus ont une portée globale.
  - **include\_once** a le même comportement mais le fichier ne sera inclus qu'une seule fois.

## require (et require\_once)

---

- L'instruction **require** a le même comportement que **include**, à la différence qu'en cas d'erreur, une erreur fatale est déclenchée, alors que **include** ne déclenche qu'un avertissement (**E\_WARNING**)
- **require\_once** a le même comportement que **include\_once**



## Ce qu'on a couvert

---

- La conditionnelle : if
- La conditionnelle multiple : switch
- Les boucles itératives : for, while, foreach
- Les interruptions d'itération : break, continue
- Les imports : include, require



## Découverte de PHP

---

- Les opérateurs

## Plan

---

- Qu'est-ce qu'une expression ?
- Qu'est-ce qu'un opérateur ?
- Différents types principaux d'opérateurs
  - Les affectations
  - Les opérateurs arithmétiques, sur les chaînes
  - Les opérateurs logiques, de comparaison



## Les expressions

---

- Une expression est un objet qui peut être défini, de manière récursive, comme une formule dont l'évaluation produit une valeur.
  - Les éléments simples du langage, comme des constantes ou des variables sont des expressions
  - La combinaison d'expressions par des opérateurs est elle-même une expression
  - La précedence des opérateurs peut être modifiée grâce aux parenthèses

## Les opérateurs

- Un opérateur est un élément du langage qui prend plusieurs expressions ou valeurs (généralement 1 ou 2) et calcule une nouvelle valeur
- Les opérateurs sont dits **préfixes**, **infixes** ou **suffixes** selon leur position dans une expression
- Les opérateurs ont une **précédence** qui détermine l'ordre de leur exécution

```
1  <?php
2
3  echo 3 + 4 * 2;    // 11
4  echo (3 + 4) * 2; // 14
```

## L'affectation

- L'affectation ('=') est un opérateur (infixe) très spécial car il permet :
  - de modifier la valeur de la variable qui est dans le membre gauche avec la valeur calculée dans le membre droit
  - de retourner une valeur, pouvant être transmise à une autre expression

## Les références

- PHP permet de désigner des valeurs « par référence » grâce au symbole &
- Une référence peut être considérée comme un « alias », un moyen de dire que plusieurs variables désignent en fait la même valeur, comme contenu logique de la mémoire.

```
1 <?php
2
3 $a = "Un exemple de texte";
4 $b = &$a;
5 echo $b;
6 $a = $a." ... modifié";
7 echo $b;
```

Définition de \$b  
comme référence à  
\$a

## Les opérateurs arithmétiques

- Les opérateurs arithmétiques suivants permettent les opérations sur les nombres :

| Exemple    | Nom            | Résultat                                                                 |
|------------|----------------|--------------------------------------------------------------------------|
| -\$a       | Négation       | Opposé de \$a.                                                           |
| \$a + \$b  | Addition       | Somme de \$a et \$b.                                                     |
| \$a - \$b  | Soustraction   | Différence de \$a et \$b.                                                |
| \$a * \$b  | Multiplication | Produit de \$a et \$b.                                                   |
| \$a / \$b  | Division       | Quotient de \$a et \$b.                                                  |
| \$a % \$b  | Modulo         | Reste de \$a divisé par \$b.                                             |
| \$a ** \$b | Exponentielle  | Résultat de l'élevation de \$a à la puissance \$b. Introduit en PHP 5.6. |

## Les opérateurs sur les chaînes de caractères

- Il n'existe qu'un seul opérateur sur les chaînes :
  - La concaténation, symbolisée par le caractère '.'
- La concaténation peut être combinée avec l'affectation en utilisant le raccourci '.='

## Opérateurs de comparaison

| Exemple                        | Nom                   | Résultat                                                                                                                                                                      |
|--------------------------------|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>\$a == \$b</code>        | Egal                  | <b>TRUE</b> si <code>\$a</code> est égal à <code>\$b</code> après le transtypage.                                                                                             |
| <code>\$a === \$b</code>       | Identique             | <b>TRUE</b> si <code>\$a</code> est égal à <code>\$b</code> et qu'ils sont de même type.                                                                                      |
| <code>\$a != \$b</code>        | Différent             | <b>TRUE</b> si <code>\$a</code> est différent de <code>\$b</code> après le transtypage.                                                                                       |
| <code>\$a &lt;&gt; \$b</code>  | Différent             | <b>TRUE</b> si <code>\$a</code> est différent de <code>\$b</code> après le transtypage.                                                                                       |
| <code>\$a !== \$b</code>       | Différent             | <b>TRUE</b> si <code>\$a</code> est différent de <code>\$b</code> ou bien s'ils ne sont pas du même type.                                                                     |
| <code>\$a &lt; \$b</code>      | Plus petit que        | <b>TRUE</b> si <code>\$a</code> est strictement plus petit que <code>\$b</code> .                                                                                             |
| <code>\$a &gt; \$b</code>      | Plus grand            | <b>TRUE</b> si <code>\$a</code> est strictement plus grand que <code>\$b</code> .                                                                                             |
| <code>\$a &lt;= \$b</code>     | Inférieur ou égal     | <b>TRUE</b> si <code>\$a</code> est plus petit ou égal à <code>\$b</code> .                                                                                                   |
| <code>\$a &gt;= \$b</code>     | Supérieur ou égal     | <b>TRUE</b> si <code>\$a</code> est plus grand ou égal à <code>\$b</code> .                                                                                                   |
| <code>\$a &lt;=&gt; \$b</code> | Opérateur combiné     | Un entier inférieur, égal ou supérieur à zéro lorsque <code>\$a</code> est respectivement inférieur, égal, ou supérieur à <code>\$b</code> . Disponible depuis PHP 7.         |
| <code>\$a ?? \$b ?? \$c</code> | Opérateur d'union nul | Retourne la première opérande depuis la gauche vers la droite qui existe, et différent de <b>NULL</b> . Si rien n'existe, <b>NULL</b> sera retourné. Disponible depuis PHP 7. |

## Remarque

- Noter la différence entre les opérateurs ('==') qui ne comparent que la valeur et les opérateurs qui tiennent compte du type de la valeur

```
1 <?php
2
3 '3' == 3 // vrai
4 '3' === 3 //faux (chaîne <-> entier)
```

## Les opérateurs logiques

- Les opérateurs logiques permettent de déterminer la valeur de vérité d'une expression. Elles entrent en particulier dans la composition de l'opérateur ternaire <si>?<alors>:<sinon> :

| Exemple     | Nom       | Résultat                                                      |
|-------------|-----------|---------------------------------------------------------------|
| \$a and \$b | And (Et)  | TRUE si \$a ET \$b valent TRUE.                               |
| \$a or \$b  | Or (Ou)   | TRUE si \$a OU \$b valent TRUE.                               |
| \$a xor \$b | XOR       | TRUE si \$a OU \$b est TRUE, mais pas les deux en même temps. |
| ! \$a       | Not (Non) | TRUE si \$a n'est pas TRUE.                                   |
| \$a && \$b  | And (Et)  | TRUE si \$a ET \$b sont TRUE.                                 |
| \$a    \$b  | Or (Ou)   | TRUE si \$a OU \$b est TRUE.                                  |

```
1 <?php
2
3 $i = 0;
4 $a = ($i == 0) ? 'valeur nulle' : 'valeur non nulle';
```



## Ce qu'on a couvert

---

- Qu'est-ce qu'une expression ?
- Qu'est-ce qu'un opérateur ?
- Différents types principaux d'opérateurs
  - Les affectations
  - Les opérateurs arithmétiques, sur les chaînes
  - Les opérateurs logiques, de comparaison



## Découverte de PHP

---

- Les fonctions

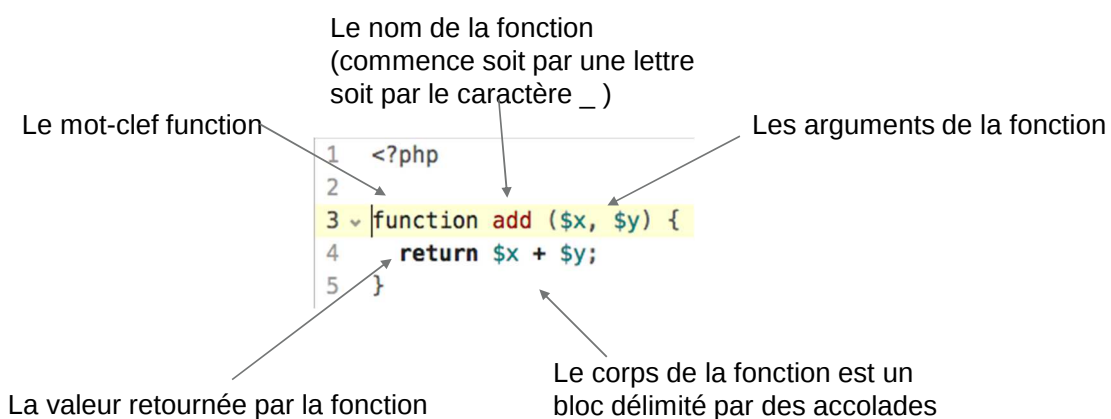
## Plan

- Comment déclarer une fonction ?
- Les propriétés des arguments de fonction
- Les valeurs de retour
- Les appels de fonctions dynamiques



## Déclarer une fonction

- Une fonction est déclarée par le mot-clef 'function'



## Les arguments des fonctions

---

- Les arguments de fonction possèdent plusieurs caractéristiques :
  - Ils peuvent contenir des valeurs par défaut
  - Ils peuvent être typés (de manière incomplète avant PHP7)
  - Ils peuvent être regroupés par l'opérateur 'reste' : ...\$reste
  - Ils peuvent être passés par référence

## Argument par défaut

---

- Pour passer un argument implicite, il suffit de lui donner une valeur dans la signature de la fonction.
- Les arguments potentiellement implicites doivent être accumulés à la fin de la signature

```
1  <?php
2
3  function add ($x, $y = 1) {
4      return $x + $y;
5  }
6
7  echo add(5);
```

## Argument typé

- Dans PHP 5, un argument peut admettre un type, à condition que celui-ci soit une **classe** (cf. POO) ou un **tableau** (array)
- Avec PHP 7, un argument admet n'importe quel type (integer, bool, ...)

```
1 <?php
2
3 function add (array $t) {
4     return implode(' ', $t);
5 }
6
7 echo add([1, 2, 3]);
```

L'argument \$t doit être un tableau, sinon une erreur est déclenchée

## L'opérateur « reste »

- Une fonction peut admettre un nombre indéfini de paramètres. Ceux-ci sont alors regroupés dans un tableau.

```
1 <?php
2
3 function add ($x, ...$y) {
4     $somme = $x;
5     foreach ($y as $i) {
6         $somme = $somme + $i;
7     }
8     return $somme;
9 }
10
11 echo add(1,2);
12 echo add(1,2,3,4,5);
```

L'opérateur ... qui permet regrouper une série de paramètres dans un tableau

## Le passage par référence

- Les paramètres sont généralement passés « par valeur »
- On peut les passer « par référence », dans ce cas, la variable locale est un alias de la variable externe, pointant vers le même contenu.

```
1 <?php
2
3 $s = 0;
4
5 function add (&$x, $y) {
6     $x = $x + $y;
7     return $x;
8 }
9
10 $resultat = add ($s, 1);
11 echo $s;
```

Le caractère & indique que  
l'argument est une  
référence (un alias) à une  
autre variable

La variable dénotée par la  
référence est modifiée.  
(effet de bord)

## La valeur de retour

- L'instruction 'return' interrompt l'exécution de la fonction et retourne la valeur qui lui est associée.
- On ne peut retourner qu'une seule valeur, mais une fonction peut très bien contenir plusieurs points de sortie (pas toujours recommandé)

La valeur retournée par la fonction

```
1 <?php
2
3 function add ($x, $y) {
4     return $x + $y;
5 }
```

## Les fonctions dynamiques

- Comme les variables, les fonctions peuvent être dynamiques.

```
1 <?php
2
3 function add ($x, $y) {
4     $x = $x + $y;
5     return $x;
6 }
7
8 $f = 'add';
9 echo $f(4,5);
```

La fonction est choisie  
dynamiquement au  
moment de l'exécution  
du code

## Ce qu'on a couvert

- Comment déclarer une fonction ?
- Les propriétés des arguments de fonction
- Les valeurs de retour
- Les appels de fonctions dynamiques



## Découverte de PHP

---

- Les objets

## Plan

---

- Introduction à la « programmation orientée objet »
- Notion de classe
- Notion d'instance
- Propriétés et méthodes



## Programmation Orientée Objet

- La programmation orientée objet (*POO*) est un paradigme qui s'est progressivement étendu à la très large majorité des langages de programmation
- Elle vise :
  - à diminuer les risques de conflits de nommage en établissant une nouvelle forme de portée (l'objet)
  - à représenter un programme comme un écosystème d'agents (les objets) communiquant entre eux par des envois de messages.

## Les classes

- En PHP, les objets sont bâtis sur des modèles que l'on appelle des classes :

```
1  <?php
2
3  class Personne {
4
5      $prenom;
6      $nom;
7      $date_de_naissance;
8      $métier;
9      $amis = [];
10     $hobbys = [];
11
12     function connait ($quelqu_un) {
13         // ...code...
14     }
15
16     function homonymes () {
17         // ...code...
18     }
19 }
```

## Les instances

- Une instance est un objet construit sur un certain modèle :

```
1  <?php
2
3  > class Personne {
20
21  $michel = new Personne();
```

L'opérateur new crée le nouvel objet de type Personne

## Les propriétés d'un objet

- Une propriété est une variable associée à un objet.
- On peut demander « le nom de cette personne » et non le nom « en général.

```
1  <?php
2
3  > class Personne {
20
21  $michel = new Personne();
22
23  echo $michel->nom;
```

Pas de caractère \$  
devant le nom de la  
propriété

La propriété nom est liée  
intrinsèquement à l'objet ; cette liaison  
est notée par la flèche

## Les méthodes d'un objet

- On appelle méthode une fonction associée à un objet

```
1  <?php
2
3  > class Personne {
20
21  $michel = new Personne();
22  $luc = new Personne();
23
24  if $michel->connait($luc) {
25      $michel->amis[] = $luc;
26  };
```

La fonction connait ne peut être exécutée que dans le contexte de l'objet.

On dit que l'on envoie le message connait à l'objet \$michel

## Ce qu'on a couvert

- Introduction à la « programmation orientée objet »
- Notion de classe
- Notion d'instance
- Propriétés et méthodes

*La POO est un domaine très vaste et une formation complète sera consacrée à son fonctionnement en PHP*





# Les formulaires

## Découverte de PHP

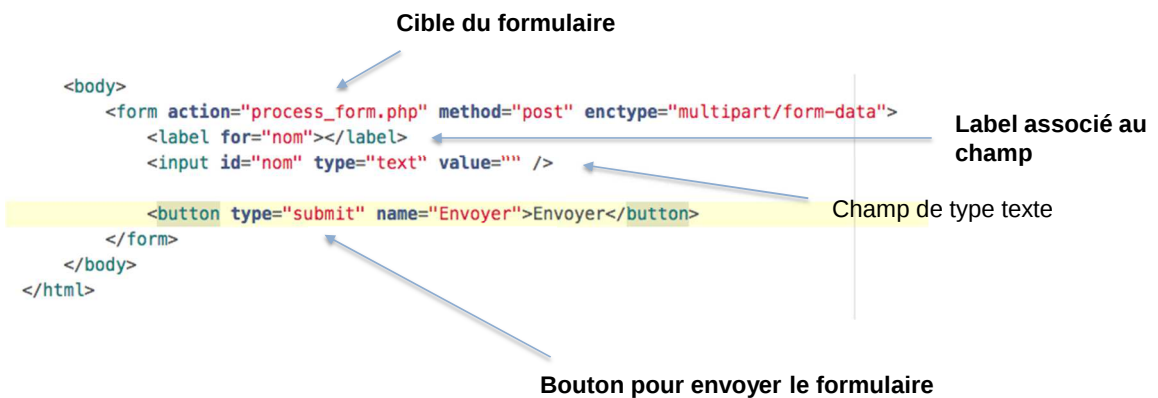


- Rappel sur les formulaires HTML
- La variable super-globale \$\_POST
- Les traitements des fichiers

## Les formulaires HTML

- Un formulaire est un élément HTML qui comprend des champs de différents types
- Un formulaire est envoyé au serveur par le biais d'un « événement » appelé submit.
- Le formulaire est envoyé via la méthode HTTP POST qui permet d'associer des données à la requête (hors de l'URL)
- Un formulaire a bien entendu une cible, l'URL où se trouve le programme qui traitera les données envoyées

## Exemple



```
<body>
  <form action="process_form.php" method="post" enctype="multipart/form-data">
    <label for="nom"></label>
    <input id="nom" type="text" value="" />
    <button type="submit" name="Envoyer">Envoyer</button>
  </form>
</body>
</html>
```

The diagram shows an HTML form structure with four annotations:

- Cible du formulaire**: Points to the `action="process_form.php"` attribute in the `<form>` tag.
- Label associé au champ**: Points to the `<label for="nom"></label>` tag.
- Champ de type texte**: Points to the `<input id="nom" type="text" value="" />` tag.
- Bouton pour envoyer le formulaire**: Points to the `<button type="submit" name="Envoyer">Envoyer</button>` tag.

## Les éléments de formulaire

- Il existe différents types de widgets pour les champs de formulaires HTML : *input*, *select*, *checkbox*, *radio*, etc.
- Chaque champ est repéré par son attribut « name » qui permettra au programme cible de le reconnaître
- Les attributs **name** peuvent être des tableaux, dont on a besoin en particulier pour les cases à cocher.

## Exemple : cases à cocher

Attribut name sous forme de tableau

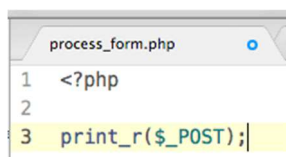
```
<label for="tag1">Travail</label>
<input type="checkbox" id="tag1" name="tags[]" value="travail" />
<label for="tag2">Personnel</label>
<input type="checkbox" id="tag2" name="tags[]" value="personnel" />
<label for="tag3">Autre</label>
<input type="checkbox" id="tag3" name="tags[]" value="autre" />
```

## Les types d'encodage

- On utilise deux types d'encodage différents pour les formulaires :
  - **application/x-www-form-urlencoded** : lorsque l'on souhaite envoyer des formulaires avec des données simples.
  - **multipart/form-data** : lorsque l'on a besoin d'envoyer des fichiers par le biais d'un champ de type file.

## Le tableau \$\_POST

- Du côté du serveur, on doit avoir un script à l'URL désignée pour traiter les données
- Dans notre exemple, ce script est contenu dans un fichier qui s'appelle **process\_form.php**
- En recevant la requête HTTP, le pré-processeur PHP enrichit automatiquement le tableau super-global **\$\_POST**



```
process_form.php
1 <?php
2
3 print_r($_POST);
```

Afficher le contenu du tableau  
super-global \$\_POST

## Le traitement des données

---

- Le tableau `$_POST` est associatif
- Chaque clef correspond à un name du formulaire HTML
- Les données sont toutes transmises sous forme de chaînes de caractères

## Le cas des fichiers

---

- Si l'on veut téléverser des fichiers sur le serveur via un formulaire, la technique est un peu différente
- Tous les fichiers transmis avec des champs de type file sont stockés dans un répertoire temporaire et leurs noms conservés dans le tableau super-global `$_FILES`
- Chaque clef du tableau correspond à l'attribut name du formulaire
- Chaque entrée a plusieurs paramètres :
  - *name, tmp\_name, size, type, error*

## Exemple

Une entrée de \$\_FILES correspondant au champ de nom 'pictures'

```
5 $uploads_dir = '/uploads';
6 foreach ($_FILES["pictures"]["error"] as $key => $error) {
7     if ($error == UPLOAD_ERR_OK) {
8         $tmp_name = $_FILES["pictures"]["tmp_name"][$key];
9         $name = $_FILES["pictures"]["name"][$key];
10        move_uploaded_file($tmp_name, "$uploads_dir/$name");
11    }
12 }
```

Le nom temporaire attribué par PHP au fichier téléversé

Pour être conservé, le fichier doit être explicitement déplacé dans un répertoire définitif



- Rappel sur les formulaires HTML
- La variable super-globale \$\_POST
- Les traitement des fichiers

## Découverte de PHP

---

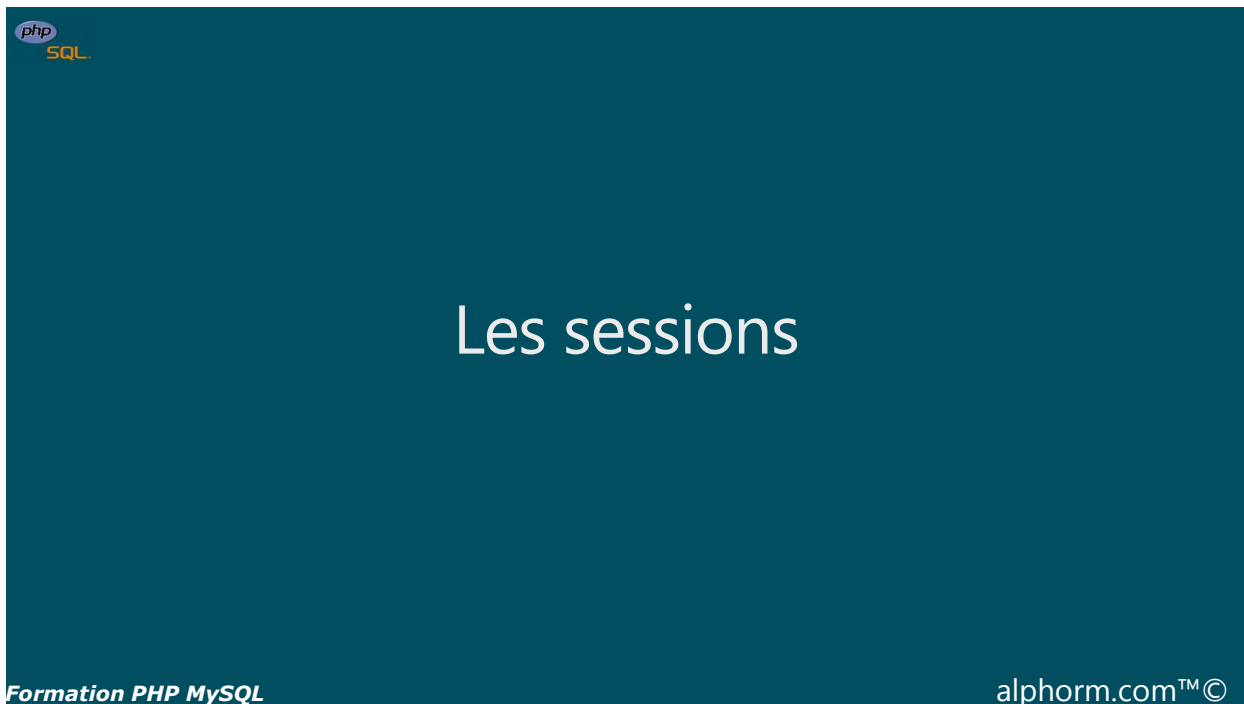
- Sessions et cookies

## Plan

---

- Le rôle des sessions
  - HTTP, un protocole sans états
- Manipuler les variables de session grâce à `$_SESSION`
- Session et sécurité
- Les cookies
  - Rôle des cookies
  - Gérer des cookies avec `$_COOKIE`





## Pourquoi les sessions ?

---

- HTTP est un protocole que l'on appelle « sans état » :
  - Chaque requête envoyée au serveur est indépendante des autres
  - Les données ne sont pas mémorisées
  - Les utilisateurs ne sont pas conservés
- Les sessions sont le premier étage de la gestion de la sécurité

## Créer une session

---

- Pour activer la session, on utilise généralement au début du programme la fonction **session\_start()**;
- **session\_start()** crée un identifiant pour la session
- D'autres fonctions permettent de terminer la session, voire de la détruire, etc.

## La variable super-globale \$\_SESSION

---

- **\$\_SESSION** est un tableau géré par PHP qui conserve les données pour chaque utilisateur entre deux requêtes.
- Une session est identifiée à un identifiant de session unique pour chaque utilisateur
- Le navigateur connaît la session car l'identifiant lui est transmis dans un cookie (cf. plus loin)
- La session peut également être indiquée dans l'URL avec l'option **PHPSESSID**

## Les données de session

---

- Un programme peut stocker n'importe quelle donnée dans la variable `$_SESSION`
- Comme tableau associatif, il suffit de créer un index correspondant

```
1  <?php
2
3  $user = "Hector";
4  $_SESSION['user'] = $user;
5
6  echo $_SESSION['user'];|
```

## Session et sécurité

---

- Les sessions peuvent être cibles de plusieurs types d'attaques :
  - Injection SQL
  - Cross-Site scripting (XSS)
  - Empoisonnement de session
  - Prise de contrôle de session
  - Fixation de session

## Sécurité et php.ini

---

- Ces attaques peuvent être contrées :
  - Par programme, en vérifiant systématiquement toutes les données envoyées par le navigateur
  - En vérifiant dans les fichiers de configuration de PHP que certaines options ont des valeurs correctes



# Les cookies

Formation PHP MySQL

alphorm.com™©

## Rôle des cookies

---

- Les cookies peuvent être considérés comme des variables (avec un nom, donc) passées au navigateur lors de l'envoi de la réponse HTTP et qui ont une durée de vie définie (potentiellement infinie)
- Les cookies peuvent être utilisés à de fins très diverses, éventuellement pour espionner les activités de l'utilisateur du navigateur
- Ce ne sont pas des informations fiables, car l'utilisateur peut les effacer à tout moment, voire les interdire.

## Le tableau super-global \$\_COOKIE

---

- Lors d'une requête HTTP, le navigateur envoie les cookies associés au domaine
- Les valeurs sont stockées dans le tableau \$\_COOKIE

## Créer un cookie

- Le serveur peut créer autant de cookies qu'il souhaite.
- Il suffit d'utiliser la fonction **setcookie()**

```
1 <?php
2
3 $expires = time() + 3600; // expire dans une heure
4
5 setcookie('exemple', 'PHP 2016', $expires, '/', 'cours.exemple.com', false, true)
```

## Ce qu'on a couvert

- Le rôle des sessions
  - HTTP, un protocole sans états
- Manipuler les variables de session grâce à **\$\_SESSION**
- Session et sécurité
- Les cookies
  - Rôle des cookies
  - Gérer des cookies avec **\$\_COOKIE**



## Découverte de PHP

---

- Organiser l'application

## Plan

---

- Passer d'une « page web » à une application
- Décomposer le cycle d'exécution en éléments « simples »
- Donner un point d'entrée unique à son application



## Une page HTML/PHP

- Le premier mode d'utilisation de PHP est celui qui enrichit des squelettes HTML statiques (c'est l'origine de PHP)

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title><?php $titre = "Hello World !"; echo $titre; ?></title>
6   </head>
7   <body>
8     <?php echo strtoupper($titre) ?>
9   </body>
10 </html>
```

## Une application PHP

- Le deuxième mode consiste à écrire des programmes PHP qui vont engendrer le code HTML à envoyer au navigateur en réponse

```
1 <?php
2 // Requête = http://exemple.com/page.php?op=mult&x=25&y=37
-
```

Il existe à priori un script  
page.php à la racine du site

## Des bibliothèques de fonctions

- Un programme est constitué d'une collection de fonctions qui constituent des bibliothèques

```
3
4 // Bibliothèque de fonctions
5 function mult ($x, $y) { return $x * $y; }
6 function add ($x, $y) { return $x + $y; }
```

Les fonctions sont regroupées par catégories, séparées du script et organisées en classes avec la POO

## L'examen de la requête

- L'examen de la requête (appelé aussi routage) permet au programme de décider de la suite des traitements

```
7
8 // Traitement des données de la requête (HTTP GET)
9 $fonction = $_GET['op'];
10 if (!in_array($fonction, ['mult', 'add'])) {
11     die("La fonction demandée n'existe pas");
12 }
13 $op1 = isset($_GET['x'])? $_GET['x'] : die("Vous devez donner une valeur à X");
14 $op2 = isset($_GET['y'])? $_GET['y'] : die("Vous devez donner une valeur à Y");
```

Les valeurs sont le tableau  
super-global \$\_GET

die interrompt le programme en cas d'erreur

## Les traitements

---

- Une fois les données recueillies, les fonctions appropriées peuvent calculer les sorties

```
15
16 // Délégation du traitement à la bibliothèque
17 $résultat = $fonction($op1, $op2);
```

## Construction de la réponse

---

- La phase suivante consiste à choisir et/ou construire les squelettes HTML. C'est la phase de rendu (confiée à des outils appelés 'vues')

```
18
19 // Construction de la réponse
20 $reponse = <<<HTML
21 <html>
22   <head></head>
23   <body>
24     <p>Le résultat de $op1 $fonction $op2 vaut $resultat</p>
25   </body>
26 </html>
27 HTML;
```

## Envoi de la réponse

- Une fois la réponse construite, il ne reste qu'à l'envoyer
- Eventuellement, on peut personnaliser les entêtes HTTP

```
28
29 // Envoi de la réponse
30 header('Content-Type', 'text/html');
31 echo $reponse;
```

## Mettre en œuvre la redirection HTTP

- Si vous voulez ne pas multiplier les pages, il est possible d'utiliser le fichier `.htaccess` pour demander au serveur web de servir une autre ressource

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ /index.php?path=$1 [NC,L,QSA]
```



## Ce qu'on a couvert

---

- Passer d'une « page web » à une application
- Décomposer le cycle d'exécution en éléments « simples »
- Donner un point d'entrée unique à son application



## Découverte de PHP

---

- Documenter  
ses programmes

## Plan

---

- Pourquoi documenter le code ?
- Les outils de documentation pour PHP
- La syntaxe PHPDoc
- Installer PHPDocumentor
- Un exemple



## Documenter son code

---

- Documenter son code est une des bonnes pratiques les plus importantes de la programmation :
  - Cela permet la transmission du code
  - Cela permet les pratiques de revue de code (par le pairs)
  - Cela facilite les tests
  - Cela permet la fabrication semi-automatique de documentation technique pour les programmes (et le utilisateurs :o)



- Il existe en PHP plusieurs outils de documentation. Et deux formats :
  - PHPDoc
  - Doxygen

## Commenter avec PHPDoc

- Créer une documentation **PHPDoc** consiste simplement à ajouter un bloc de commentaire devant chaque variable, fonction, objet que vous voulez documenter.

```
1 <?php
2
3  ▾ /** ← Bloc de commentaires dont la
4     * première ligne est toujours /**
5     */
6  ▾ function add ($x, $y) {
7     return $x + $y;
8 }
```

## Les principales annotations

- La documentation se construit à l'aide d'annotations.

auteur du code

arguments de la fonction

exemple d'utilisation

```
1 <?php
2
3 /**
4  * Addition de deux nombres (description courte)
5  *
6  * [...] Description longue de la fonction
7  *
8  * @author Michel Cadennes
9  * @version 1.0
10 *
11 * @param integer $x Premier nombre à additionner
12 * @param integer $y Second nombre à additionner
13 * @return integer Somme des deux nombres
14 *
15 * @example add(4,5)
16
17 * @see math.php Bibliothèque de fonctions mathématiques
18 */
19 function add ($x, $y) {
20     return $x + $y;
21 }
```

## Installer PHPDocumentor

- PHPDocumentor fonctionne avec des versions de PHP > 5.3.3
- Cela nécessite juste d'installer la bibliothèque **GraphViz** au préalable
- Il suffit ensuite d'installer l'archive **PHAR** du projet

## Créer la documentation

- Pour créer la documentation associée au projet, il suffit d'exécuter la ligne de commande :

```
php phpDocumentor.phar -d . -t docs|
```

chemin du répertoire racine de  
la documentation

## Ce qu'on a couvert

- Pourquoi documenter le code ?
- Les outils de documentation pour PHP
- La syntaxe PHPDoc
- Installer PHPDocumentor
- Un exemple



## Plan

---

# Découverte de PHP Sessions et cookies



## Plan

---

- Le rôle des sessions
  - HTTP, un protocole sans états
- Manipuler les variables de session grâce à \$\_SESSION
- Les cookies
  - Rôle des cookies
  - Gérer des cookies avec \$\_COOKIE



## Plan

---

# Les sessions



## Plan

---

- HTTP est un protocole que l'on appelle « sans état » :  
Pourquoi les sessions ?
  - Chaque requête envoyée au serveur est indépendante des autres
  - Les données ne sont pas mémorisées
  - Les utilisateurs ne sont pas conservés
- Les sessions sont le premier étage de la gestion de la sécurité





## Plan

- Pour activer la session, on utilise généralement au début du programme la fonction `session_start()`;  
**Créer une session**
- `session_start()` crée un identifiant pour la session
- D'autres fonctions permettent de terminer la session, voire de la détruire, etc.



## Plan

- `$ SESSION` est un tableau géré par PHP qui conserve les données pour chaque utilisateur entre deux requêtes.  
**La variable super-globale \$SESSION**
- Une session est identifiée à un identifiant de session unique pour chaque utilisateur
- Le navigateur connaît la session car l'identifiant lui est transmis dans un cookie (cf. plus loin)
- La session peut également être indiquée dans l'URL avec l'option `PHPSESSID`



## Plan

---

- Un programme peut stocker n'importe quelle donnée dans la variable `$_SESSION`
- Comme tableau associatif, il suffit de créer un index correspondant

```
1 <?php
2
3 $user = "Hector";
4 $_SESSION['user'] = $user;
5
6 echo $_SESSION['user'];|
```



## Plan

---

### Les cookies



## Plan

---

- Les cookies peuvent être considérés comme des variables (avec un nom, donc) passées au navigateur lors de l'envoi de la réponse HTTP et qui ont une durée de vie définie (potentiellement infinie)
- Les cookies peuvent être utilisés à de fins très diverses, éventuellement pour espionner les activités de l'utilisateur du navigateur
- Ce ne sont pas des informations fiables, car l'utilisateur peut les effacer à tout moment, voire les interdire.



## Plan

---

- Lors d'une requête HTTP, le navigateur envoie les cookies associés au domaine
- Les valeurs sont stockées dans le tableau `$_COOKIE`



## Plan

- Le serveur peut créer autant de cookies qu'il souhaite.  
Créer un cookie
- Il suffit d'utiliser la fonction setcookie()

```
1 <?php
2
3 $expires = time() + 3600; // expire dans une heure
4
5 setcookie('exemple', 'PHP 2016', $expires, '/', 'cours.exemple.com', false, true)
```



## Plan

- Le rôle des sessions
  - HTTP, un protocole sans états
- Manipuler les variables de session grâce à \$\_SESSION
- Les cookies
  - Rôle des cookies
  - Gérer des cookies avec \$\_COOKIE



## Introduction à MySQL

---

- Le modèle relationnel

## Plan

---

- Les tables
- Les attributs et les tuples
- Les relations
- Les clefs primaires et les index





## Différents modèles de bases de données

- Les bases de données les plus répandues aujourd'hui s'appuient sur ce que l'on appelle le « **calcul relationnel** »
- De nombreux autres modèles de bases de données existent :
  - bases de données hiérarchiques et graphe
  - bases XML (ou RDF)
  - bases de données objet
  - etc.



## Modèle relationnel

## Tables

---

- Une base de données relationnelle est composée de « **tables** » qui sont des tableaux à deux dimensions dont :
  - les colonnes sont les attributs qui définissent la structure (ou le schéma) de la table
  - les lignes sont les enregistrements qui constituent le contenu de la table

## Attribut

---

- Les attributs sont les éléments minimaux des bases de données relationnelles. Ils ont comme caractéristiques :
  - d'avoir un nom
  - d'avoir un domaine : l'ensemble de valeurs possibles
  - d'être (ou non) unique
  - d'être un index (ou une clef) et en particulier une clef primaire

## Relation

---

- Une relation est une extension de la notion de table, définie comme étant le sous-ensemble du produit cartésien d'une liste de domaines
  - une table est un cas particulier de relation

## Tuple

---

- Un tuple est une extension de la notion d'enregistrement qui se définit comme une collection de valeurs sur des attributs déterminés
  - Un enregistrement d'une table est un cas particulier de tuple

## Clef primaire

---

- Une clef primaire est un identifiant unique qui caractérise un enregistrement d'une table
  - Souvent, les clefs primaires sont créées automatiquement selon une stratégie d'auto-incrémentation de valeurs numériques

## Index

---

- En plus de la clef primaire, il est souvent avantageux de créer des index, qui permettent d'accélérer la recherche d'information dans la base
- Un index correspond souvent à un attribut, mais ce n'est pas systématique
  - un index peut correspondre à un fragment d'attribut
  - un index peut s'étendre sur plusieurs attributs

## Clef étrangère

---

- On appelle clef étrangère, un attribut dont la valeur correspond à un enregistrement dans une autre table
- Le but principal des clefs étrangères est de vérifier l'intégrité référentielle du modèle, c'est-à-dire le fait que l'information soit toujours cohérente

## Ce qu'on a couvert

---

- Les tables
- Les attributs et les tuples
- Les relations
- Les clefs primaires et les index



## Introduction à MySQL

---

- Le calcul relationnel

## Plan

---

- Qu'est-ce que calcul relationnel ?
- Notion d'algèbre relationnelle
- Les différents opérateurs
  - Projection
  - Restriction
  - Union
  - Jointure
  - Intersection
  - Différence relationnelle



## Calcul relationnel

---

- Le calcul relationnel est une notation logique, où les requêtes sont exprimées par la formulation de restrictions logiques que les tuples doivent satisfaire dans la réponse.

## Algèbre relationnelle

---

- L'Algèbre Relationnelle qui est une notation algébrique, où les requêtes sont exprimées en appliquant des opérateurs spécialisées pour les relations.



## Projection

- On appelle projection, l'opération qui consiste à ne sélectionner que certains attributs d'une relation
- Une projection peut conduire à l'apparition de doublons que l'on sera conduit à éliminer/fusionner

**SELECT titre, description FROM tache**

Champs déterminant la projection

## Projection

---

| id | titre | description | date | auteur |
|----|-------|-------------|------|--------|
|    |       |             |      |        |
|    |       |             |      |        |
|    |       |             |      |        |
|    |       |             |      |        |

## Restriction

---

- On appelle restriction l'opération qui consiste à ne sélectionner que certains enregistrements d'une relation sur un critère donné

```
SELECT * WHERE auteur = 'arthur'
```

La clause WHERE permet de définir des restrictions

## Restriction

| id | titre | description | date | auteur |
|----|-------|-------------|------|--------|
|    |       |             |      | john   |
|    |       |             |      | arthur |
|    |       |             |      | arthur |
|    |       |             |      | john   |

## Union

- L'union est l'opération ensembliste qui consiste à prendre la totalité des éléments de plusieurs relations ayant le même schéma.
- L'union peut conduire à l'apparition de doublons de tuiles, qui seront supprimés

```
SELECT auteur FROM tache WHERE auteur = 'arthur'  
UNION  
SELECT auteur FROM note WHERE id > 100
```

La clause UNION permet de définir des unions

## Union

| id  | auteur |
|-----|--------|
| 1   | john   |
| 42  | arthur |
| 123 | arthur |
| 256 | john   |

+

| id  | auteur  |
|-----|---------|
| 3   | john    |
| 28  | arthur  |
| 312 | claudio |

## Jointure

- La jointure est l'opération qui consiste à faire le produit cartésien de deux relations.
- Les jointures sont souvent faites sous des contraintes particulières

```
SELECT * FROM tache JOIN categorie
```

La clause JOIN (et ses relatives) permet de fusionner des relations

## Jointure

| id | titre     | categorie |   | id | nom     |
|----|-----------|-----------|---|----|---------|
| 1  | Séminaire | 1         | X | 1  | Travail |
| 2  | Cinéma    | 2         |   | 2  | Loisirs |
| 3  | Voyage    | 2         |   | 3  | Autre   |
| 4  | RV Claude | 1         |   |    |         |

## Intersection

- L'intersection est l'opération ensembliste qui ne consiste à retenir que les tuiles communs à plusieurs relations
- La clause **INTERSECT** n'est pas implémentée dans MySQL, mais on peut le faire grâce à des sous-requêtes

## Différence relationnelle

---

- La **différence** est l'opération ensembliste qui consiste à ne retenir que les tuiles apparaissant dans une relation mais pas dans l'autre
- La clause **MINUS** n'est pas implémentée dans MySQL, mais on peut le faire grâce à des sous-requêtes

## Ce qu'on a couvert

---

- Qu'est-ce que calcul relationnel ?
- Notion d'algèbre relationnelle
- Les différents opérateurs
  - Projection
  - Restriction
  - Union
  - Jointure
  - Intersection
  - Différence relationnelle



## Introduction à MySQL

---

- Les formes normales

## Plan

---

- Qu'appelle-t-on forme normale ?
- Les formes normales
  - 1FN (première forme normale)
  - 2FN (deuxième forme normale)
  - 3FN (troisième forme normale)
  - 4FN (quatrième forme normale)





## Qu'appelle-t-on forme normale ?

---

- Dans une base de données relationnelle, une forme normale désigne un type de relation particulier entre les entités.
- La normalisation des modèles de données permet de vérifier la robustesse de leur conception pour améliorer la modélisation (et donc obtenir une meilleure représentation) et faciliter la mémorisation des données en évitant la redondance et les problèmes sous-jacents de mise à jour ou de cohérence. La normalisation s'applique à toutes les entités et aux relations porteuses de propriétés.



## Première forme normale

---

- Toutes les données sont atomiques ;
- contiennent une valeur scalaire (les valeurs ne peuvent pas être divisées en plusieurs sous-valeurs dépendant également individuellement de la clé primaire) ;
- contiennent des valeurs non répétitives (le cas contraire consiste à mettre une liste dans un seul attribut) ;
- sont constants dans le temps (utiliser par exemple la date de naissance plutôt que l'âge)

## Deuxième forme normale

---

- Les attributs d'une relation sont divisés en deux groupes :
  - Le premier groupe est composé de la clef (un ou plusieurs attributs).
  - Le deuxième groupe est composé des autres attributs (éventuellement vide).
- La deuxième forme normale stipule que tout attribut du deuxième groupe ne peut pas dépendre d'un sous-ensemble (strict) d'attribut(s) du premier groupe.
- En d'autres termes : « Un attribut non-clef ne dépend pas d'une partie de la clef ».

## Troisième forme normale

---

- Les attributs d'une relation sont divisés en deux groupes :
  - Le premier groupe est composé de la clef (un ou plusieurs attributs).
  - Le deuxième groupe est composé des autres attributs (éventuellement vide).
- La troisième forme normale stipule que tout attribut du deuxième groupe ne peut pas dépendre que d'un sous-ensemble (strict) d'attribut(s) du deuxième groupe.
- En d'autres termes : « Un attribut non-clef ne dépend pas d'un ou plusieurs attributs ne participant pas à la clef ».

## Quatrième forme normale

---

- Pour toute relation de dimension **n** en forme normale de **Boyce-Codd**, les relations de dimension **n-1** construites sur sa collection doivent avoir un sens.
- Il ne doit pas être possible de reconstituer les occurrences de la relation de dimension **n** par jointure de deux relations de dimension **n-1**.
- Cette normalisation conduit parfois à décomposer une relation complexe en deux relations plus simples.

## Ce qu'on a couvert

---

- Qu'appelle-t-on forme normale ?
- Les formes normales
  - 1FN (première forme normale)
  - 2FN (deuxième forme normale)
  - 3FN (troisième forme normale)
  - 4FN (quatrième forme normale)



## Introduction à MySQL

---

- phpMyAdmin

## Plan

---

- Accéder à phpMyAdmin
- Créer une base de données
- Créer une table
- Définir les propriétés d'une colonne



## Accéder à l'interface de phpMyAdmin

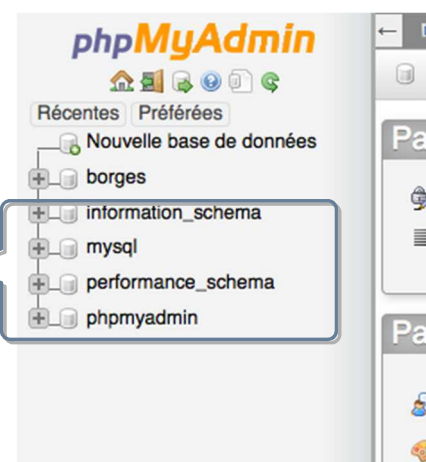
- Pour accéder à l'administration des bases de données de votre serveur, vous devez vous identifier comme administrateur de MySQL



## La liste des bases

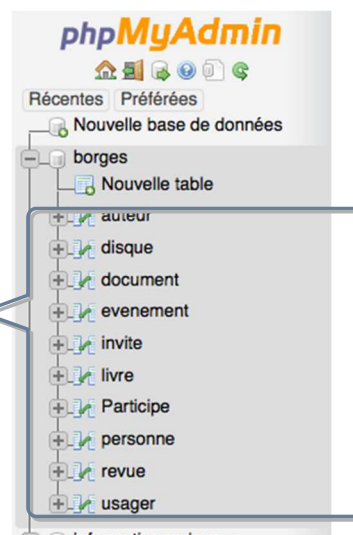
- Dans la colonne de gauche, la liste des bases de données

Bases de données  
« administratives »

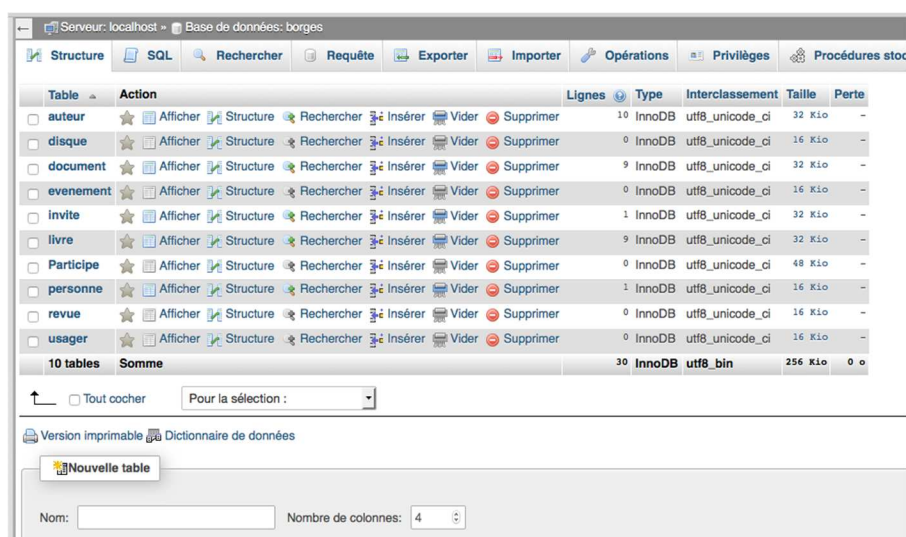


## phpMySQL Structure d'une base de données

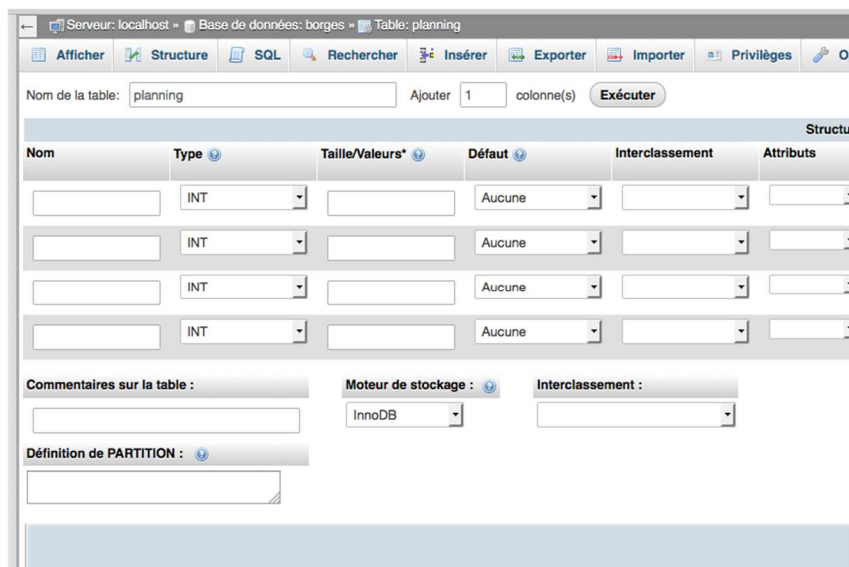
Les tables de la  
base de données  
borges



## phpMySQL L'interface de manipulation de la base



## Création d'une table



Structure

| Nom | Type | Taille/Valeurs* | Défaut | Interclassement | Attributs |
|-----|------|-----------------|--------|-----------------|-----------|
|     | INT  |                 | Aucune |                 |           |
|     | INT  |                 | Aucune |                 |           |
|     | INT  |                 | Aucune |                 |           |
|     | INT  |                 | Aucune |                 |           |

Commentaires sur la table :

Moteur de stockage : InnoDB

Interclassement :

Définition de PARTITION :

## Les propriétés d'une colonne

- Nom
- Type
- Taille ou valeurs
- Valeur par défaut
- Interclassement
- Attribut
- Vide
- Index
- Commentaires
- Type MIME
- Transformation
- Options de transformation

## Les moteurs de stockage

---

- CSV
- MRG\_MYISAM
- BLACKHOLE
- MEMORY
- MyISAM
- ARCHIVE
- InnoDB

## Ce qu'on a couvert

---

- Accéder à phpMyAdmin
- Créer une base de données
- Créer une table
- Définir les propriétés d'une colonne



## Introduction à MySQL

---

### • Clefs et index

## Plan

---

- Les clefs primaires
- Les index
  - Les différents types d'index
  - La recherche plein texte
- Les clefs étrangères



## Clef primaire

---

- La clé primaire d'une table est une **contrainte d'unicité**, composée d'une ou plusieurs colonnes, et qui permet d'**identifier de manière unique** chaque ligne de la table.
- Les clefs primaires sont presque toujours indispensables pour chaque table
- Les clefs primaires sont souvent numériques et auto-incrémentées

## Créer une clef primaire

---

- Pour ajouter une clef primaire à une table, on peut :
  - soit passer par l'interface phpMyAdmin
  - soit utiliser une **requête SQL**

```
ALTER TABLE tache ADD PRIMARY KEY id
```

On modifie la table tache en ajoutant la colonne id comme clef primaire

## Index

---

- Les index sont des structures de données ajoutées à la base de données qui permettent de trouver plus vite l'information recherchée, exactement comme l'index d'un livre.
  - Les index sont très efficaces en lecture
  - En revanche, ils prennent davantage de temps lors des opérations d'écriture ou de modification
  - En conséquence, il faut utiliser des index, mais les utiliser à bon escient : sur des données qui serviront très souvent de base à la recherche

## Types d'index

---

- Il y a plusieurs types d'index :
  - Les **INDEX** « normaux »
  - Les **UNIQUE** qui forcent les colonnes à ne contenir que des valeurs uniques, comme les clefs primaires
  - Les **FULLTEXT** qui sont destinées à supporter la recherche « plein texte »
  - Les **SPATIAL** qui sont liées aux coordonnées spatiales et à OpenGIS

## Déclaration des index

---

- Les index peuvent être déclarés à tout moment :
  - Lors de la création de la table, dans une requête SQL de type **CREATE TABLE**
  - Via l'interface **phpMyAdmin**
  - Par une requête SQL de type **CREATE INDEX** après la création de la table concernée

```
CREATE INDEX titre_abrege ON tache (titre(10));
```

On crée un index sur les 10 premiers caractères du titre

## Les index FULLTEXT

---

- Les index de type FULLTEXT permettent de faire de manière très efficace des recherches à l'intérieur de colonnes contenant des textes longs. Trois modes de recherche sont possibles :
  - mode naturel : recherche tous les mots indiqués (pas nécessairement conjoints)
  - mode booléen : permet d'ajouter des options et des jokers aux mots recherchés
  - mode extension de requête : effectue automatiquement une deuxième passe en cherchant les mots issus de la première passe

## Exemple

On cherche les mots travail  
et urgent

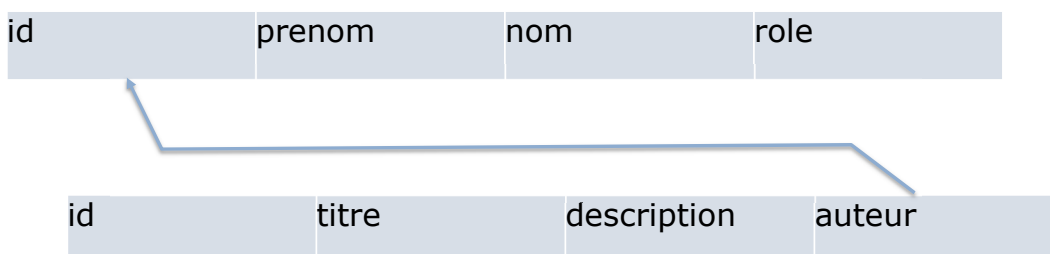
```
SELECT *  
FROM tache  
WHERE MATCH (description)  
AGAINST ('travail urgent' IN NATURAL LANGUAGE MODE)
```

```
SELECT *  
FROM tache  
WHERE MATCH (description)  
AGAINST ('-travail +urgent' IN BOOLEAN MODE)
```

On cherche les descriptions contenant le mot urgent  
mais pas travail

## Clefs étrangères

- Une clef étrangère est une clef unique (souvent primaire) dans une table qui sert de référence à la valeur d'une colonne dans une autre table



## Déclaration d'une clef étrangère

- Dans MySQL, il y a plusieurs manières de déclarer une clef étrangère :
  - dans la requête SQL qui crée la table
  - par une requête spécifique
  - par l'interface phpMyAdmin

## Exemple

```
ALTER TABLE tache  
ADD CONSTRAINT fk_auteur_tache  
FOREIGN KEY (auteur)  
REFERENCES tache(id)
```

L'identifiant de la clef  
étrangère

La colonne  
« source »

La table et la colonne de  
référence



## Ce qu'on a couvert

---

- Les clefs primaires
- les index
  - les différents types d'index
  - La recherche plein texte
- Les clefs étrangères



## Introduction à MySQL

---

- L'interface mysql

## Plan

---

- Les styles syntaxiques
- Créer une connexion à la base
- Des requêtes simples
- Les requêtes préparées
- Utiliser une procédure stockée



## mysqli

---

- mysqli est une API spécifique qui transcrit dans PHP les commandes de MySQL
- mysqli est l'une des nombreuses interfaces que PHP intègre pour accéder aux bases de données de divers constructeurs (Oracle, Postgresql, Informix, MongoDB, etc.)
- mysqli est venu remplacer l'ancienne interface mysql, devenue obsolète

## Les styles

---

- mysqli offre deux styles d'écriture :
  - un style procédural
  - et un style orienté objet

## Ouvrir une connexion sur une base

---

- On rend disponible une ressource « base de données » en instanciant la classe mysqli

```
$db = new mysqli('host', 'utilisateur', 'mot_de_passe', 'database');
```

## Exécution d'une requête

- Une requête SQL peut être exécutée avec la méthode `query`

```
$res = $db->query('INSERT INTO tache (auteur) VALUES (5)');
```

- Pour les requêtes de type **SELECT**, la méthode `query` renvoie un tableau comprenant tous les résultats trouvés

## Requête préparée

- Le processus de requête peut être décomposé en plusieurs temps :
  - La préparation

```
$res = $db->prepare('INSERT INTO tache (titre) VALUES (?)');
```

- La liaison des variables

```
$res = $db->bind_param('i', 'rendez-vous demain soir');
```

- L'exécution

```
$res = $db->execute();
```

## Les procédures stockées

---

- Les procédures stockées sont des formes de fonctions mémorisées dans la base de données et que l'on peut exécuter en leur passant des paramètres
- Pour créer une procédure stockée, nous avons le choix entre :
  - utiliser l'interface phpMyAdmin
  - utiliser la commande SQL : **CREATE PROCEDURE**

## Exemple

---

```
CREATE PROCEDURE p(IN id_val INT)
BEGIN
INSERT INTO tache(id) VALUES(id_val);
SELECT * FROM tache
END;
```



## Appler une procédure stockée

---

- L'appel de la procédure se fait exactement comme une requête simple :

```
$db->query("CALL p(1)")
```

- CALL appelle la procédure p



## Ce qu'on a couvert

---

- Les styles syntaxiques
- Créer une connexion à la base
- Des requêtes simples
- Les requêtes préparées
- Utiliser une procédure stockée



## Introduction à MySQL

---

- Requêtes avec PDO

## Plan

---

- Créer une connexion à la base
- Des requêtes simples
- Les requêtes préparées
- Le mode transactionnel



## PDO

---

- PDO (*PHP Data Objects*) est une couche d'abstraction pour SQL qui permet de se connecter à toute une variété de bases de données relationnelles
- On accède donc à la base de données au travers d'un pilote adapté

## Le DSN

---

- Le DSN représente le **Data Source Name**
- Il peut se présenter sous plusieurs formes mais par défaut il comportera 3 segments :

mysql:dbname=<nom\_de\_la\_base>:host=<adresse\_IP>

## Connexion a une base

- Contrairement à mysqli, PDO n'accepte que le style objet

```
$db = new PDO('mysql:host=127.0.0.1;dbname=<nom>', 'host', 'utilisateur');
```

 Déclaration du pilote pour MySQL

## Requêtes

- De la même manière que mysqli, PDO possède une méthode query qui permet d'exécuter une requête SQL

```
$res = $db->query('INSERT INTO tache (auteur) VALUES (5)');
```

- Il est également possible de préparer les requêtes

## Requête préparée

- Le processus de requête peut être décomposé en plusieurs temps:
  - La préparation

```
$sth = $db->prepare('SELECT * FROM tache WHERE id > :id');
```

- L'exécution de la requête

```
$err = $sth->execute( ['id' => 10 ] );
```

- Récupération des résultats

```
$res = $sth->fetchAll();
```

## Le mode transactionnel

- Le mode transactionnel permet de valider ou d'annuler une série de requêtes en fonction d'un certain signal.
- Il faut alors conclure manuellement la transaction :
  - soit en la validant par un **commit** ;
  - soit en l'annulant par un **rollback** ;

## Exemple

```
$dbh->beginTransaction();

$sth = $dbh->exec("DROP TABLE fruit");
$sth = $dbh->exec("UPDATE dessert SET name = 'hamburger'");

if (is_ok) {
    $dbh->commit();
} else {
    $dbh->rollBack();
}
```

## Ce qu'on a couvert

- Créer une connexion à la base
- Des requêtes simples
- Les requêtes préparées
- Le mode transactionnel



## Introduction à MySQL

---

- Requêtes simples avec PDO

## Plan

---

- Le formulaire HTML
- La traitement du formulaire dans PHP
- La création de la requête SQL
- Recherche des données modifiées
- Affichage des données dans une page HTML



## Le formulaire HTML

---

- Le formulaire HTML doit contenir :
  - Les champs définissant une nouvelle tâche : *un titre, une description, une date*
  - Un sélecteur permettant de savoir qui a défini cette tâche
    - Pour cela, une première requête donnant la liste des personnes sera exécutée

## Le traitement du formulaire

---

- Lorsque le formulaire est renvoyé au serveur (après validation par le navigateur lui-même), les données sont dans le tableau super-global **`$_POST`**
- La première étape est de préparer une requête qui servira à ajouter un nouvel enregistrement dans la table des tâches.

## Une requête d'insertion dans la base

---

- L'étape suivante consistera à exécuter la requête précédente en lui associant les variables issues du formulaire
- Il faut enfin vérifier que les requêtes n'ont pas provoqué d'erreur

## Une requête pour rechercher les données

---

- Nous souhaitons maintenant afficher la liste des tâches en cours, c'est-à-dire dont le statut est ouvert et la date limite non dépassée (par exemple) pour une personne donnée
- Nous avons là aussi préparé une requête et ensuite l'exécuter avec les paramètres qui conviennent



## Le renvoi d'une page HTML contenant les résultats

---

- La précédente requête de sélection nous rendra un **ResultSet**, c'es-à-dire un ensemble de résultats de recherche sous forme de tableau.
- Nous allons alors devoir itérer sur ce tableau pour en extraire les données et créer les éléments HTML correspondant pour renvoyer la page au navigateur.
- Le cycle d'exécution du cas d'utilisation sera alors achevé



## Ce qu'on a couvert

---

- Le formulaire HTML
- La traitement du formulaire dans PHP
- La création de la requête SQL
- Recherche des données modifiées
- Affichage des données dans une page HTML



## Conclusion

---

- Le mot de la fin

## Plan

---

- Rappel des objectifs du cours
- Une synthèse des points abordés dans les différents chapitres
- Ce qui n'a pas été couvert, mais le sera par la suite



## **Rappel des objectifs**

---

- L'objectif de ce cours est de vous permettre d'appréhender l'architecture globale de ce qu'on appelle la plate-forme LAMP :
  - Approcher les ressources du système d'exploitation (Linux, OS X) qui vous permettront d'installer (relativement) facilement les logiciels nécessaires
  - Examiner les solutions prêtes à l'emploi de type XAMPP ou MAMP
  - Comprendre les bases du langage PHP
  - Comprendre le fonctionnement d'une base de données
  - Ecrire un premier programme

## **Mettre en œuvre un serveur web**

---

- Installer les différentes briques de la plate-forme LAMP :
  - Apache 2
  - MySQL
  - PHP 5
  - VSFTPD
- Configurer les différents éléments
- Créer des utilisateurs
- Mettre en route le serveur et créer ses premières pages

## Utiliser XAMPP

---

- Comme alternative, nous avons utilisé une solution toute prête.
- Installation et configuration de XAMPP
- Démarrage et arrêt des services
- Organisation des sites hébergés par le serveur
- Lancement du serveur et création des premières pages

## Introduction à PHP

---

- Le lien en PHP et Apache et le fonctionnement du **pré-processeur** et le traitement des requêtes
- Comment écrire une page intégrant du PHP ?
- Les fondements du langage :
  - Les types de base
  - Les structures de contrôle
  - Les variables (et les variables dynamiques)
- La gestion des sessions et des cookies
- Le traitement des formulaires HTML
- Les bases de la programmation objet
- Les pratiques de documentation du code

## Introduction à MySQL

---

- Qu'est-ce qu'une « base de données relationnelle » ?
- Les fondements de la structure d'une base de données :
  - tables, colonnes, index, clef primaire, clef étrangère
- Les bases du calcul relationnel et les différents types de requêtes SQL
- La notion de « forme normale »
- Créer et gérer une base avec phpMyAdmin
- La connexion entre un programme PHP et la base de données
- L'utilisation des données avec PHP

## Coursus PHP sur Alphorm

---





## Ce qui reste à voir

---

- La structure et le vocabulaire du langage PHP
  - les fonctions, le système de fichiers, le traitement des chaînes de caractères, les tableaux, le traitement des bases de données
- La programmation orientée objet en PHP 5 et PHP 7
  - le système des classes, les méthodes magiques, les interfaces, les traits, les classes anonymes
- PHP avancé
  - les formats XML et JSON, la programmation fonctionnelle, la programmation réseau, les questions de sécurité



A bientôt !