



**WEBFORCE**  
BE THE CHANGE



## RÉSUMÉ THÉORIQUE - FILIÈRE DÉVELOPPEMENT DIGITAL M107 - DÉVELOPPER DES SITES WEB DYNAMIQUES



77 heures



# SOMMAIRE



## 1. INTRODUIRE LA NOTION CLIENT/SERVEUR

- Approfondir la notion client/serveur
  - Introduire le langage PHP
- Préparer l'environnement de développement

## 2. PROGRAMMER EN PHP

- Maîtriser le langage PHP
- Traiter les données en PHP
- Utiliser l'orientée objet en PHP

## 3. MANIPULER LES DONNÉES

- Écrire des scripts d'accès aux données
  - Sécuriser les données

## 4. RÉALISER UN SITE WEB AVEC L'ARCHITECTURE MVC

- Développer des sites dynamiques avec MVC en mode natif
  - Découvrir les Web services et les API REST

# MODALITÉS PÉDAGOGIQUES



1

## LE GUIDE DE SOUTIEN

Il contient le résumé théorique et le manuel des travaux pratiques



2

## LA VERSION PDF

Une version PDF est mise en ligne sur l'espace apprenant et formateur de la plateforme WebForce Life



3

## DES CONTENUS TÉLÉCHARGEABLES

Les fiches de résumés ou des exercices sont téléchargeables sur WebForce Life



4

## DU CONTENU INTERACTIF

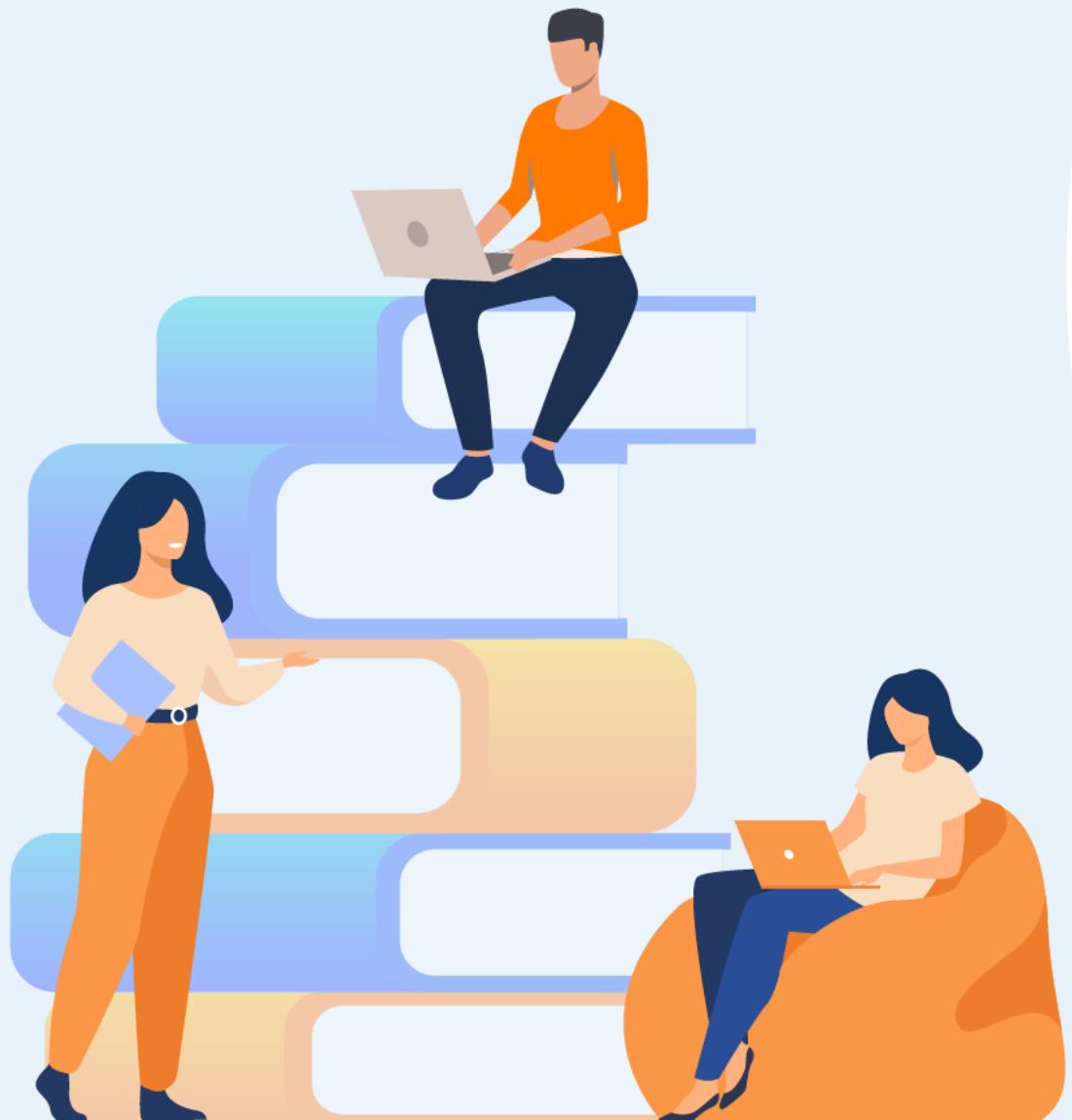
Vous disposez de contenus interactifs sous forme d'exercices et de cours à utiliser sur WebForce Life



5

## DES RESSOURCES EN LIGNES

Les ressources sont consultables en synchrone et en asynchrone pour s'adapter au rythme de l'apprentissage



## PARTIE 1

### Approfondir la notion Client/Serveur

Dans ce module, vous allez :

- Identifier les différents types de serveurs web
- Acquérir une bonne connaissance de l'architecture client/serveur
- Maîtriser l'environnement de développement



12 heures

# CHAPITRE 1

## Approfondir la notion Client/Serveur



Ce que vous allez apprendre dans ce chapitre :

- Principe Client/serveur
- Architectures 2-tiers
- Types de serveurs web



04 heures



# CHAPITRE 1

## Approfondir la notion Client/Serveur

1. **Principe Client/serveur**
2. Architectures 2-tiers
3. Types de serveurs web

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur

### Introduction

À partir du moment où les architectures matérielles peuvent être interconnectées, nous pourrons réaliser une **architecture client/serveur**.

L'architecture client-serveur s'appuie sur **un poste central, le serveur**, qui envoie des données aux machines clientes. Les programmes qui accèdent au serveur sont appelés **programmes clients** (client FTP, client mail, navigateur).

Cette architecture est basée sur l'utilisation de deux types de logiciels : **un logiciel serveur et un logiciel client** s'exécutant normalement sur 2 machines différentes.

L'élément important dans cette architecture est l'utilisation de mécanismes de communication entre deux applications.

Le logiciel client-serveur est indépendant des plateformes matérielles et logicielles dite hétérogénéité.

Le logiciel client-serveur masque aux clients la localisation du serveur.

Les services internet sont conçus selon cette architecture.

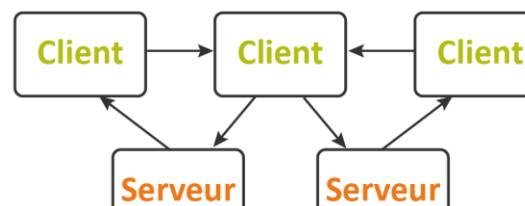
Chaque application est composée de logiciel serveur et logiciel client.

Présentation de <https://perso.univ-lyon1.fr/olivier.gluck> :



Requête/Réponse

Fig. : Un client, un serveur



Le serveur traite plusieurs requêtes simultanées

Fig. : Plusieurs clients, un serveur



Le serveur contacté peut faire appel à un service sur un autre serveur (ex.SGBD)

Fig. : Un client, plusieurs serveurs

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur

### C'est quoi un Client ?



Un client est un consommateur de services.



Le client déclenche la demande de service.



Une requête est un appel de fonction, la réponse éventuelle pouvant être synchrone ou asynchrone  
(le client peut émettre d'autres requêtes sans attendre)



Les arguments et les réponses sont énoncés dans un protocole



Le respect du protocole entre les deux processus communicants est obligatoire.  
Ce protocole étant décrit dans un RFC (Request For Comment).

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur

### C'est quoi un Serveur ?

- Un serveur est un ordinateur ou un système qui met des ressources, des données, des services ou des logiciels à la disposition d'autres ordinateurs, qualifiés de « clients », sur un réseau.
- Les machines serveurs sont généralement dotées de capacités supérieures à celles des ordinateurs personnels en ce qui concerne la puissance de calcul, les entrées-sorties et les connexions réseau, afin de pouvoir répondre de manière efficace à un grand nombre de clients.
- Pour pouvoir offrir les services d'un serveur en permanence, le serveur doit être sur un site avec accès permanent sans interruption de connexion qui pourra interrompre la communication ainsi ses services. Dans le schéma par exemple c'est grâce à une connexion Internet.
- Pour un logiciel serveur, plusieurs logiciels clients sont utilisés dans des environnements différents : Unix, Mac, PC... par exemple.
- **Un serveur itératif** : désigne une implémentation qui traite une seule requête à la fois.
- **Un serveur parallèle** : fonctionne en mode concurrent, désigne une implémentation capable de gérer plusieurs tâches en apparence simultanées.
- **Service avec état** : le serveur conserve localement un état pour chacun des clients connectés : informations sur le client, les requêtes précédentes, ...
- **Service sans état** : le serveur ne conserve aucune information sur l'enchaînement des requêtes...

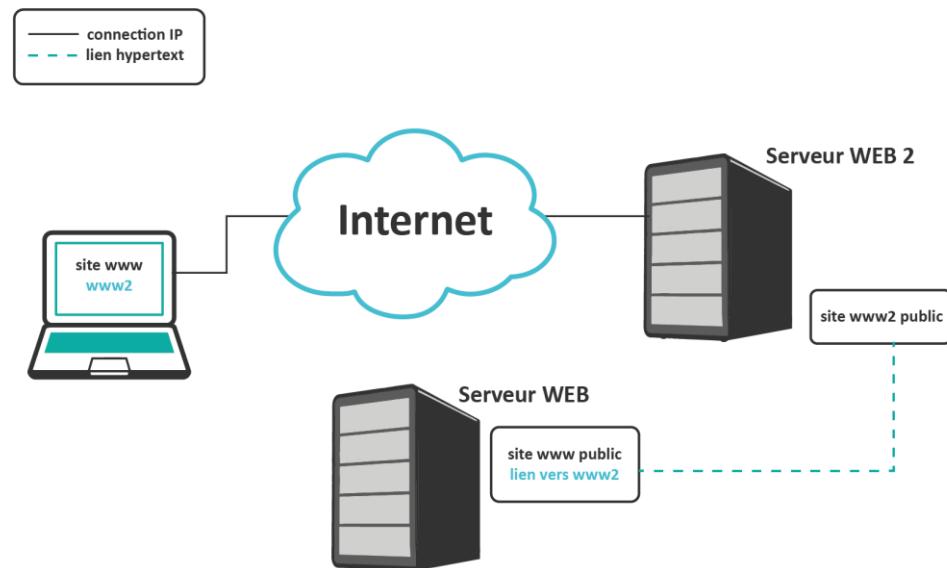


Fig : Internet connexion client serveur. Source: <https://www.ipgp.fr/>

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur

### C'est quoi un Middleware ?

- La liaison entre le client et le serveur qui se charge de toutes les communications entre les processus est appelée Middleware.
- Un middleware est un logiciel médiateur ou intergiciel qui se charge de la liaison.

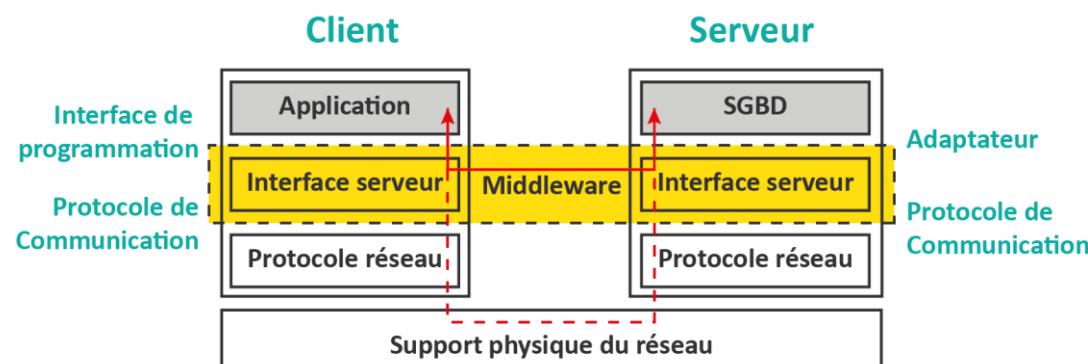


Fig : Représentation Middleware selon Gartner Group

### Les fonctions d'un Middleware

- Procédure d'établissement de connexion
- Exécution des requêtes
- Récupération des résultats
- Procédure de fermeture de connexion
- Initiation des processus sur différents sites
- Services de répertoire (nommage)
- Accès aux données à distance
- Gestion des accès concurrents
- Sécurité et intégrité
- Monitoring
- Terminaison des processus
- Mise en cache des résultats et des requêtes

### Les services d'un Middleware

- Conversion
- Adressage
- Sécurité
- Communication

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur



WEBFORCE  
BE THE CHANGE

### Avantages et Inconvénients

AVANTAGES	INCONVÉNIENTS
<ul style="list-style-type: none"><li>Des ressources centralisées, tout en garantissant la cohérence des données.</li><li>Une meilleure sécurité, puisque le nombre de points d'entrée permettant l'accès aux données est connus.</li><li>Une administration au niveau serveur.</li><li>Un réseau évolutif, on peut supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modifications majeures.</li><li>Une interface utilisateur riche.</li><li>Une appropriation des applications par l'utilisateur.</li><li>Une notion d'interopérabilité.</li></ul>	<ul style="list-style-type: none"><li>Déploiement coûteux : un coût élevé dû à la technicité du serveur</li><li>Trafic réseau important : tout le réseau est architecturé autour du serveur</li><li>Déploiement difficile : le poste client doit constamment être mis à jour pour répondre au besoin</li></ul>

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur



### Fonctionnement

Fonctionnement d'un réseau client /serveur :

- Le client, pour recevoir des informations du serveur, lui émet une requête passant par un port du PC (exemple : port 25 pour les mails, port 80 pour le web et port 21 pour le FTP)
- Le serveur envoie ensuite les informations grâce à l'adresse IP de la machine cliente
- Le client traite et affiche les informations en provenance du serveur

La communication peut être établie de deux manières entre deux appareils ou plus, qui sont orientés connexion (mode connecté) et sans connexion (mode non connecté).

**Le mode non connecté ne garantit pas :**

- l'intégrité des données
- l'ordonnancement des données
- la non-duplication des données

Le mode connecté garantit les propriétés ci-dessus qui ne sont pas garanties par l'application dans le mode non connecté. D'autant plus, le mode connecté :

- Implique une diminution des performances brutes par rapport au mode non-connecté
- Peut constituer une contrainte
- Permet une implémentation asynchrone des échanges

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur



### Types d'architectures

Le découpage et la répartition des trois niveaux d'abstraction, d'une application informatique, permettent de distinguer plusieurs types d'architecture à savoir :

**Architecture 1-tiers : Architecture Centralisée.**

**Architecture 2-tiers : Architecture Client/serveur.**

**Architecture 3-tiers : Architectures Distribuées.**

**Architecture n-tiers : Architectures Distribuées.**

**L'architecture 1-tiers** : Les trois couches sont intimement liées et s'exécutent sur la même machine. On parle donc de l'informatique centralisée.

**L'architecture 2-tiers** encore appelée client-serveur de première génération ou client-serveur de données, le poste client se contente de déléguer la gestion des données à un service spécialisé.

**L'architecture 3-tiers** est l'extension du modèle client/serveur. Ce type d'architecture est le plus courant des architectures multi-tiers. Il est également appelé client-serveur de deuxième génération ou client-serveur distribué.

**L'architecture n-tiers** est une généralisation de l'architecture 3-tiers à partir d'un modèle de composants. Les liens entre les composants sont rendus possibles par l'existence d'un bus logiciel.

D'après la source <http://www-igm.univ-mlv.fr/~dr/XPOSE2001/perrot/Intro-Comparatif.htm> on dispose d'un tableau comparatif entre l'architecture 2-tiers et 3 et n-tiers, qui nous permet de recenser les avantages et les limitations de chacun des processus (voir tableau).

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur



	2-tiers	3 et n tiers
<b>Administration du système</b>	Complexe (la couche application est physiquement répartie sur plusieurs postes clients)	Moins complexe (les applications peuvent être gérées centralement sur le serveur)
<b>Sécurité</b>	Faible (sécurité au niveau des données)	Elevée (raffinée au niveau des services ou des méthodes)
<b>Encapsulation des données</b>	Faible (les tables de données sont directement accessibles)	Elevée (le client fait appel à des services ou méthodes)
<b>Performance</b>	Faible (plusieurs requêtes SQL sont transmises sur le réseau, les données sélectionnées doivent être acheminées vers le client pour analyse)	Bonne (seulement les appels de service et les réponses sont mises sur le réseau)
<b>Extensibilité</b>	Faible (gestion limitée des liens réseaux avec le client)	Excellente (possibilité de répartir dynamiquement la charge sur plusieurs serveurs)
<b>Réutilisation</b>	Faible (application monolithique sur le client)	Excellente (réutilisation des services et des objets)
<b>Facilité de développement</b>	Elevée	En progression (des outils intégrés pour développer la partie du client et du serveur)

Tab : Comparatif architecture 2-tiers Vs 3 et n-tiers

# 01 - Approfondir la notion client/serveur

## Principe Client/serveur



	2-tiers	3 et n tiers
Lien Serveur-serveur	Non	Oui (via le middleware Serveur/Serveur)
Intégration des systèmes déjà en place	Non	Oui (via des passerelles encapsulées par les services ou objets)
Soutien Internet	Faible (les limitation de la bande passante pénalisent le téléchargement d'application de type « fat-client »)	Excellente (les applications de type « thin-client » sont facilement téléchargeable et les appels aux services repartissent la charge sur un ou plusieurs serveurs)
Sources de données hétérogènes	Non	Oui (les applications 3-tier peuvent utiliser plusieurs bases de données dans la même transaction)
Choix de communication de type « riche »	Non (synchrone et RPC)	Oui (gestion asynchrone de message, files de livraison, publication et abonnement, « broadcast »)
Flexibilité d'architecture matériel	Limitée	Excellente (possibilité de faire résider les couches 2 et 3 sur une ou plusieurs machines)
Relève en cas de pannes	Faible	Excellente (possibilité d'avoir la couche du centre « middle-tier » sur plusieurs serveurs)

Tab : Comparatif architecture 2-tiers Vs 3 et n-tiers



## CHAPITRE 1

### Approfondir la notion Client/Serveur

1. Principe Client/serveur
2. **Architectures 2-tiers**
3. Types de serveurs web

### Introduction

- L'architecture 2-tiers (aussi appelée client-serveur de première génération, ou encore client-serveur de données) **caractérise les systèmes clients/serveurs pour lesquels le client à un besoin et le serveur le lui fournit directement**, en utilisant ses propres ressources.
- Le serveur traite la demande du client **sans recours à des applications tierces**.
- Ce type d'architecture permet de bénéficier pleinement de la puissance des ordinateurs déployés en réseau pour **fournir à l'utilisateur une interface riche**, tout en garantissant la cohérence des données, qui reste gérée de façon centralisée.
- La gestion des données est **prise en charge par un SGBD** (Système de gestion de base de données) centralisé, s'exécutant le plus souvent sur un serveur dédié.
- **Le langage de requête SQL** (Structured Query Language) est la méthode la plus utilisée pour consulter la base de données.
- Le dialogue entre client et serveur se résume donc à l'envoi de requêtes et au retour des données correspondant aux requêtes.

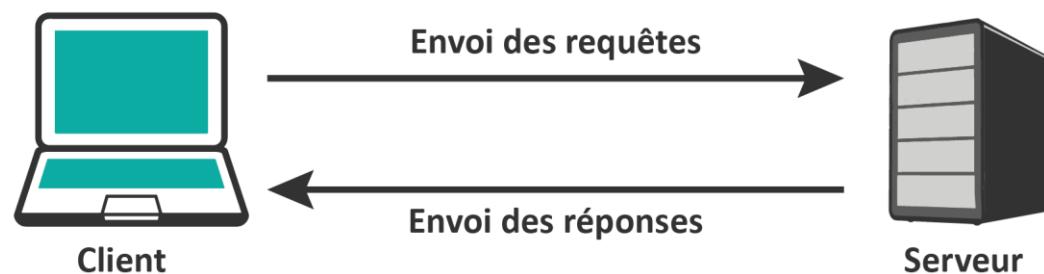


Fig. : Architecture 2 tier

# 01 - Approfondir la notion client/serveur

## Architectures 2-tiers

### C'est quoi un client ?

Le client demande un service au serveur.

Un client lourd est une application où les traitements sont principalement effectués sur la machine locale dite cliente.

#### Caractéristiques d'un client :

- Il est actif.
- Il envoie des requêtes au serveur.
- Il attend et reçoit les réponses du serveur.

### C'est quoi un serveur ?

Le serveur exécute la requête et renvoie le résultat au client.

Les performances du serveur sont atteintes rapidement après la sollicitation d'un nombre important de clients.

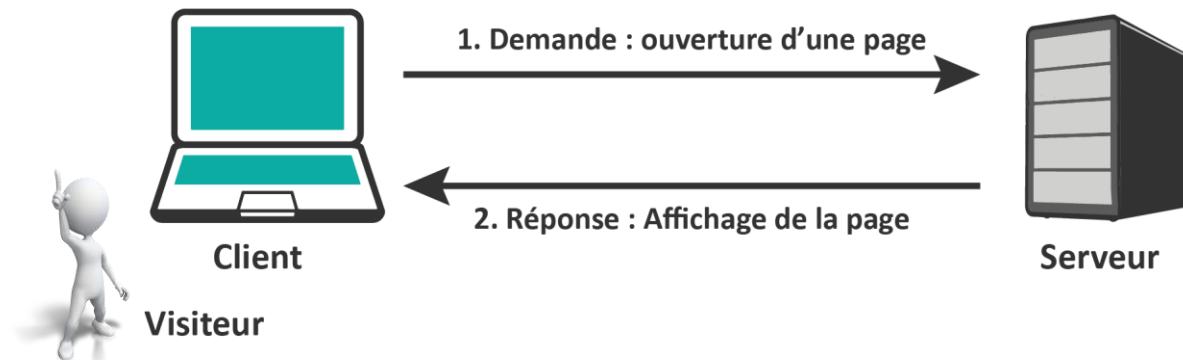


Fig. : Exemple du mécanisme de requête

### Niveau d'abstraction

Une application informatique peut être découpée en trois niveaux d'abstraction distincts :

#### Couche Présentation (Couche IHM : Interface Homme-Machine)

- Elle permet l'interaction d'application avec l'utilisateur.
- Elle contrôle en effet les saisies au clavier et à la souris ainsi que la présentation à l'écran.
- Elle doit être conviviale et ergonomique

#### Couche Traitement (Logique Applicative ou couche métier)

- Elle décrit les traitements à exécuter par l'application afin de répondre aux requêtes clients.
- Les traitements locaux : tiennent compte les contrôles effectués au niveau du dialogue avec l'IHM (formulaires, champs, boutons,...).
- Les traitements globaux : représentent les règles de l'application appelées aussi logique métier (Business Logique).

#### Couche Données (couche Persistance)

- Elle prend en charge les actions liées aux accès aux données.
- Regroupe l'ensemble des mécanismes permettant la gestion des informations stockées par l'application.
- Garantie souvent les fonctions classiques d'un SGBD (Définition de données, Manipulation de données, Sécurité de données, gestion des transactions, ...).

# 01 - Approfondir la notion client/serveur

## Architectures 2-tiers

- Gartner Group a proposé un découpage en six vues distinctes montrant les différentes possibilités de répartition entre clients et serveur des trois couches logicielles.

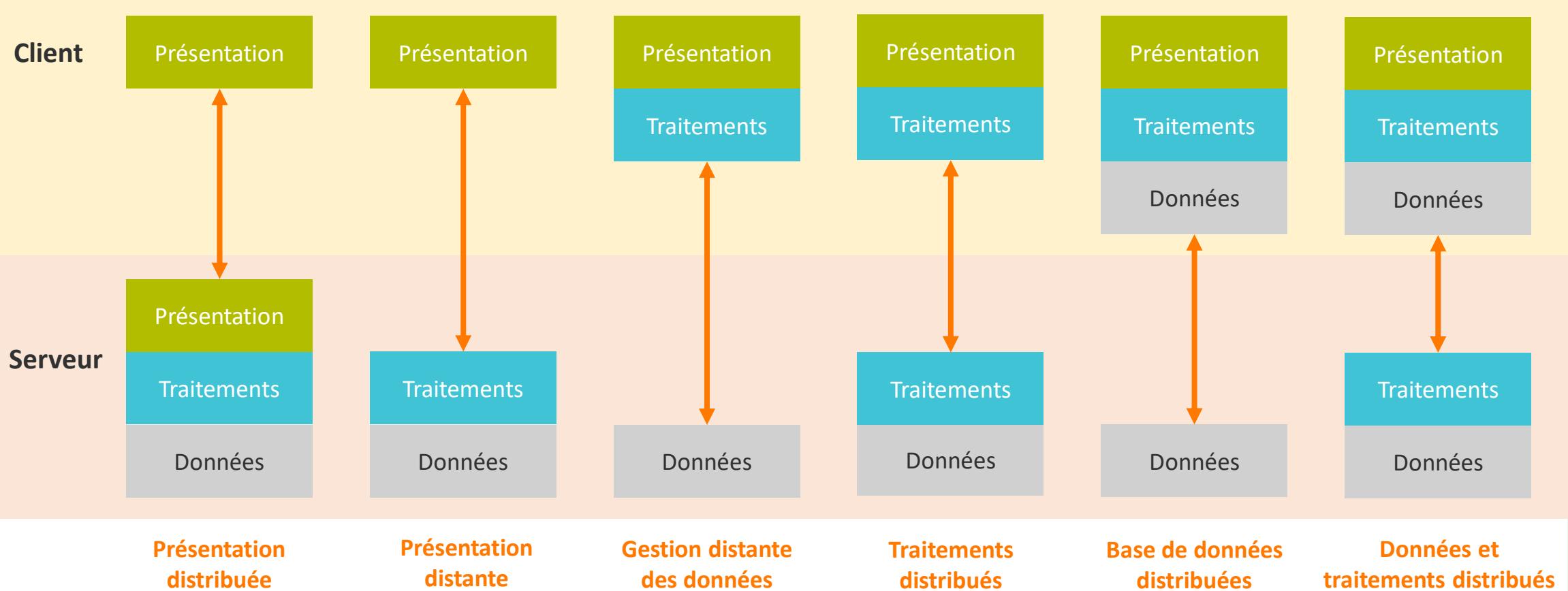


Fig : Modèle de Gartner pour les systèmes à deux niveaux (2-tiers)



## CHAPITRE 1

### Approfondir la notion Client/Serveur

1. Principe Client/serveur
2. Architectures 2-tiers
3. **Types de serveurs web**

# 01 - Approfondir la notion client/serveur

## Types de serveurs web



### Définition

#### Un serveur web

- Un serveur Web (aussi appelé serveur http), est tout type de serveur qui permet de diffuser des contenus Web sur Internet ou Intranet. C'est un service logiciel utilisé pour communiquer entre deux appareils sur un réseau.
- Un serveur web sert à rendre accessible des pages web sur internet via le protocole HTTP.
- Un serveur web répond par défaut sur le port 80.
- Pour qu'un site Web soit accessible à tout moment, le serveur Web sur lequel il est hébergé doit être connecté à Internet en permanence
- Un serveur Web en architecture 3 tiers est composé d'un système d'exploitation, un serveur HTTP, un langage serveur et un système de gestion de base de données (SGBD), cela constituant une plate-forme.

# 01 - Approfondir la notion client/serveur

## Types de serveurs web



### Définition

#### Un serveur web statique

- (aussi appelé une pile) est composé d'un ordinateur (matériel) et d'un serveur HTTP (logiciel).
- Il est appelé « statique » car le serveur envoie les fichiers hébergés « tels quels » vers le navigateur.

#### Un serveur web dynamique

- Possède d'autres composants logiciels, certains qu'on retrouve fréquemment dont un serveur d'applications et une base de données.
- Il est appelé « dynamique » car le serveur d'applications met à jour les fichiers hébergés avant de les envoyer au navigateur via HTTP.

# 01 - Approfondir la notion client/serveur

## Types de serveurs web

### Qui sont les clients ?

Les navigateurs web jouent le rôle de clients :



**Brave**

- Respect et protection de la vie privée
- Interface et ergonomie soignée
- Le plus rapide du marché



**Chrome**

- Très bonnes performances
- Simple et agréable à utiliser
- Un navigateur bien sécurisé



**Firefox**

- Un des plus innovants du marché
- Fonctionnalités pour optimiser l'UX
- Gère les derniers standards vidéo AV1



**Edge**

- Performances correctes
- Plus léger, rapide et moderne
- L'intégration à l'écosystème Windows/Microsoft...



**Opéra**

- Stable et performant
- Débits rapides
- Sécurité accrue avec outils de chiffrement efficaces



**Vivaldi**

- Bonne alternative aux navigateurs du marché
- Niveau de personnalisation
- Fonctionnalités intuitives et originales



**UC Browser**



**Next**



**Safari**



**Maxthon**



**UR Browser**

# 01 - Approfondir la notion client/serveur

## Types de serveurs web



### Qui est le Serveur ?

- Le serveur : machine qui exécute les requêtes et envoie les réponses. Liste des serveurs web :

- |                          |                                       |                                     |                   |
|--------------------------|---------------------------------------|-------------------------------------|-------------------|
| • AOLserver              | • IBM HTTP Server                     | • NCSA HTTPd                        | • Apache Tomcat   |
| • Apache HTTP Server     | • Internet Information Services (IIS) | • Nginx                             | • TUX web server  |
| • Boa Web Server         | • Jetty                               | • Node js                           | • Wakanda Server  |
| • Caudium                | • Lighttpd                            | • OpenLink Virtuoso                 | • Xitami          |
| • Cherokee Web Server    | • LiteSpeed Web Server                | • Oracle HTTP Server                | • Yaws            |
| • HTTP File Server (HFS) | • Mongrel                             | • Oracle iPlanet Web Server Perlbal | • Zeus Web Server |
| • Hiawatha Web Server    | • NaviServer                          | • Thttpd                            | • ...             |

Le protocole spécifie comment le client et le serveur communiquent.

- Protocole utilisé : HyperText Transfer Protocol (HTTP)

Les demandes contiennent l'Uniform Ressource Locator (URL) de la page à afficher. L'URL est l'extension de la notion de nom de fichier sur un réseau

- Syntaxe d'une URL : protocole://adresse/fichier (exemple : <https://www.ofppt.ma/fr/nos-formations>)

- Protocole : HTTP, FTP, news,...
- Adresse : Spécifique au protocole et en général @ IP du serveur HTTP (ou le nom, qui sera résolu par appel au serveur DNS)
- Fichier : nom du fichier à récupérer, facultatif (si omis, page par défaut : index.htm / index.html)

# 01 - Approfondir la notion client/serveur

## Types de serveurs web



### Qui est le Serveur ?

Une fois que le serveur a reçu la requête, et si celle-ci est valide, il effectue le traitement.

Les réponses contiennent des documents au format HyperText Markup Langage

Les types MIME (Multipurpose Internet Mail Extensions) permettent de préciser le type de documents transmis lors d'une communication. Repris dans HTTP/1.0 pour que les clients aient connaissance du type de document qui leur est renvoyé

**Le serveur envoie la réponse :**

- une ligne de statut (version du protocole)
- Un code de succès (ou d'échec)
- Informations (relatives au serveur et au message)
- Corps du message

**Les Codes de retour permettent d'informer sur le succès ou l'échec de la requête :**

- Information : 100 et 101 (demande de la suite de la requête, changement de protocole)
- Succès : 200 à 206
- Redirection : 300 à 305 + 307
- Erreurs du client : 400 à 417
- Erreurs du serveur : 500 à 505

# 01 - Approfondir la notion client/serveur

## Références et ressources



### Références et ressources

#### Architecture serveur/Client :

- Daniel MARTIN. Architecture des applications réparties. <http://worldserver2.oleane.com/>, Octobre 1999. /dmartin/Architecture applications reparties.htm
- <http://mariepascal.delamare.free.fr/IMG/pdf/leClientServeur1.pdf>
- <https://mrproof.blogspot.com/2011/03/larchitecture-client-serveur.html>

#### Architecture 2tiers :

- <http://mariepascal.delamare.free.fr/IMG/pdf/leClientServeur1.pdf>

#### Services Web :

- <https://pub.phyks.me/sdz/sdz/les-services-web.html>
- [http://www-inf.it-sudparis.eu/cours/WebServices/Docs/Bob\\_WS-1.pdf](http://www-inf.it-sudparis.eu/cours/WebServices/Docs/Bob_WS-1.pdf)

## CHAPITRE 2

### Introduire le langage Php



**Ce que vous allez apprendre dans ce chapitre :**

- Définition du PHP
- Généralités sur les fonctionnalités du PHP

 02 heures



## CHAPITRE 2

### Introduire le langage Php

1. Définition du PHP
2. Généralités sur les fonctionnalités du PHP

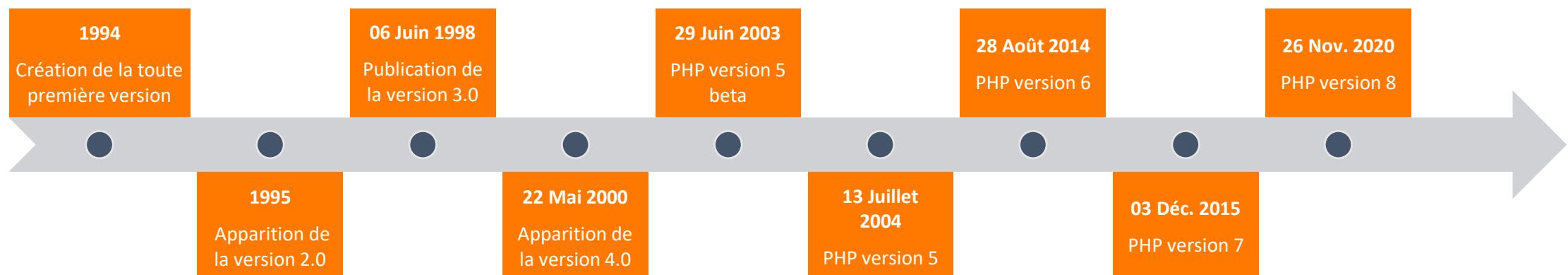
## 02 - Introduire le langage PHP

### Définition du PHP



### Un peu d'histoire

- PHP est l'acronyme de Hypertext Preprocessor
- Mis au point en **1994** par Rasmus Lerdorf pour publier son CV en ligne mais surtout conserver la trace des visiteurs qui venaient sur son site consulter son CV
- Grâce au succès de son code sympathique, et suite aux demandes multiples des internautes, Rasmus mis en ligne 1995 une première version de son programme qu'il nomma Personal Sommaire Page Tools puis Personal Home Page v1.0 (d'où le nom PHP)
- **1995**, apparition de la version 2.0 (Intégration des structures des structures plus avancées : conditions, boucles, intégration des formulaires ...)
- La version 2.0 permettra ainsi au développeur d'intégrer des instructions de programmation puissantes directement dans du code HTML.



## 02 - Introduire le langage PHP

### Définition du PHP



#### Définition

##### PHP

- PHP a permis de créer un grand nombre de sites web célèbres tels que Facebook, YouTube, Wikipedia, ...
- PHP est un langage interprété orienté Web. Syntaxiquement, c'est un mélange de C, Java et de Perl. Les scripts PHP sont lus et interprétés par le moteur PHP.
- Il a été conçu spécifiquement pour la création dynamique des pages HTML. Ainsi, il s'intègre dans une page HTML.
- PHP est un produit Open Source : chacun a le droit d'utiliser, d'étudier, de modifier, de dupliquer, de donner et de vendre cette application sans contrepartie.
- PHP est distribué sous licence GNU GPL

#### Le logo



### Définition

#### Plus de 500 fonctions

- Accès aux bases de données.
- Fonctions d'images.
- Sockets.
- Protocoles Internet divers
- ...

#### Les Principaux atouts de PHP

- La gratuité et la disponibilité du code source.
- La simplicité d'écriture de scripts.
- La possibilité d'inclure le script PHP au sein d'une page HTML.
- La simplicité d'interfaçage avec les principaux systèmes de gestion de bases de données.
- L'intégration au sein de nombreux serveurs web (Apache, Microsoft IIS, ...).
- Indépendance vis-à-vis des OS et serveurs même si l'environnement Linux est sa plateforme de prédilection.

### Langage de script

PHP a considérablement évolué, ces dernières années, d'un langage de script non organisé à un langage de programmation backend orienté objet.

Un script est un ensemble d'instructions de programmation qui est interprété à l'exécution.

Les scripts sont généralement intégrés dans d'autres environnements logiciels.

**Les langages de script sont différents des langages de programmation :**

- Les langages de script sont spécifiques à une plateforme, tandis que les langages de programmation sont multiplateformes car ils ont la capacité de s'exécuter eux-mêmes.
- Les langages de script sont souvent interprétés, les langages de programmation sont compilés. Il existe certains langages de script qui sont à la fois compilés et interprétés, comme Python et Groovy.
- Le langage de programmation est compilé grâce à son propre compilateur qui traduit la syntaxe en code machine avant l'exécution.
- Les langages de script sont interprétés ligne par ligne pendant l'exécution par l'interpréteur de la plateforme sur laquelle ils fonctionnent.
- Les compilateurs effectuent un traitement collectif des erreurs avant l'exécution, tandis que les interpréteurs évaluent le code ligne par ligne.

### Questions

#### Question 1 :

Quels sont les avantages des réseaux client/serveur?

1. Une sécurité centralisée
2. Un point de défaillance unique
3. Des sauvegardes centralisées
4. Une réduction des coûts d'exploitation

#### Question 2 :

Les quelles des caractéristiques suivantes s'applique à l'adresse IP?

1. Elle utilise une adresse de 32 bits
2. Elle dispose de conversion de nom d'hôte améliorée.
3. Cette adresse doit être unique sur le réseau.
4. Elle est fournie par le fournisseur.

#### Question 3 :

Un Middleware est un framework, comme eclipse, qui assiste un développeur à développer les composants de son architecture logicielle ?

1. Oui
2. Non

#### Question 4 :

Une application dite "distribuée" est une application logicielle dans lequel les données informatiques sont :

1. Centralisées dans un singleton créé dans un programme accessible par tous les composants du réseau
2. Réparties dans des Factory répartis sur le réseau

#### Question 5 :

L'architecture client/serveur, trouver l'affirmation fausse :

1. Il est possible de supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modifications majeures
2. Il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisée, afin d'éviter les problèmes de redondance
3. Les « clients » peuvent fonctionner entre eux de manière autonomes

#### Question 6 :

L'architecture 3-tiers est un cas particulier des architectures n-tiers ?

1. Oui
2. Non

### Questions

#### Question 7 :

La couche d'accès aux données est liée au serveur de base de données (SGBD) ?

1. Oui
2. Non

#### Question 8 :

Les coûts de développements d'une architecture 2-tiers sont plus élevés que pour du 3-tiers ?

1. Oui
2. Non

#### Question 9 :

L'architecture deux tiers est une architecture Client/Serveur ?

1. Oui
2. Non

#### Question 10 :

IP est un protocole

1. fonctionnant en mode connecté.
2. fonctionnant en mode non connecté.

#### Question 11 :

Pour envoyer un paquet de données d'un objet connecté à un serveur distant, il faut configurer l'objet. C'est-à-dire :

1. lui affecter une adresse IP, un masque réseau, une route vers le routeur du réseau et indiquer un serveur DNS.
2. lui affecter une adresse IP, une route par défaut vers le routeur du réseau.

#### Question 12 :

Que signifie DNS?

1. Dynamic Name System
2. Dynamic Nomination System
3. Domain Name System
4. Domain Name Save

### Questions

#### Question 13 :

A quoi sert le DNS ?

1. Permet de traduire une adresse logique en un nom de domaine
2. Permet de traduire un nom de domaine en une adresse logique
3. Permet de transplanter un système dans un autre système
4. Permet de traduire une adresse IP v4 en IP v6

#### Question 14 :

Quel est le protocole dédié à la navigation WEB ?

1. HTTP
2. FTP
3. CTP
4. DNS

#### Question 15 :

Qu'est-ce qui caractérise une architecture Client/Serveur ?

1. Un terminal passif
2. Un navigateur
3. Une gestion locale de l'interface graphique
4. Un poste qui dispose d'un logiciel envoyant des requêtes à un serveur

#### Question 16 :

Une architecture logicielle est un document qui décrit l'expression du besoin sous la forme de spécifications textuelles précises ?

1. Oui
2. Non

#### Question 17 :

Une architecture logicielle est un document qui décrit les composants logiciels et leurs dépendances mutuelles ?

1. Oui
2. Non

### Questions

#### Question 18 :

Le logiciel qui joue le rôle du client dans le modèle client/serveur du Web s'appelle un :

1. Explorateur de fichier
2. Navigateur
3. Moteur de recherche

#### Question 19 :

Un logiciel de serveur Web peut être installé sur n'importe quel ordinateur ?

1. Oui
2. Non

#### Question 20 :

Complétez les blancs.

Le \_\_\_\_\_ HHTP sert à définir les règles du dialogue entre le \_\_\_\_\_ et le \_\_\_\_\_ dans un échange Web.

## 02 - Introduire le langage PHP

### Quizz (Correction)



### Réponses

#### Question 1 :

Quels sont les avantages des réseaux client/serveur?

1. Une sécurité centralisée
2. Un point de défaillance unique
3. Des sauvegardes centralisées
4. Une réduction des coûts d'exploitation

#### Question 2 :

Les quelles des caractéristiques suivantes s'applique à l'adresse IP?

1. Elle utilise une adresse de 32 bits
2. Elle dispose de conversion de nom d'hôte améliorée.
3. Cette adresse doit être unique sur le réseau.
4. Elle est fournie par le fournisseur.

#### Question 3 :

Un Middleware est un framework, comme eclipse, qui assiste un développeur à développer les composants de son architecture logicielle ?

1. Oui
2. Non

#### Question 4 :

Une application dite "distribuée" est une application logicielle dans lequel les données informatiques sont :

1. Centralisées dans un singleton créé dans un programme accessible par tous les composants du réseau
2. Réparties dans des Factory répartis sur le réseau

#### Question 5 :

L'architecture client/serveur, trouver l'affirmation fausse :

1. Il est possible de supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modifications majeures
2. Il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisée, afin d'éviter les problèmes de redondance
3. Les « clients » peuvent fonctionner entre eux de manière autonomes

#### Question 6 :

L'architecture 3-tiers est un cas particulier des architectures n-tiers ?

1. Oui
2. Non

## 02 - Introduire le langage PHP

### Quizz (Correction)



### Réponses

#### Question 7 :

La couche d'accès aux données est liée au serveur de base de données (SGBD) ?

1. Oui
2. Non

#### Question 8 :

Les coûts de développements d'une architecture 2-tiers sont plus élevés que pour du 3-tiers ?

1. Oui
2. Non

#### Question 9 :

L'architecture deux tiers est une architecture Client/Serveur ?

1. Oui
2. Non

#### Question 10 :

IP est un protocole

1. fonctionnant en mode connecté.
2. fonctionnant en mode non connecté.

#### Question 11 :

Pour envoyer un paquet de données d'un objet connecté à un serveur distant, il faut configurer l'objet. C'est-à-dire :

1. lui affecter une adresse IP, un masque réseau, une route vers le routeur du réseau et indiquer un serveur DNS.
2. lui affecter une adresse IP, une route par défaut vers le routeur du réseau.

#### Question 12 :

Que signifie DNS?

1. Dynamic Name System
2. Dynamic Nomination System
3. Domain Name System
4. Domain Name Save

## 02 - Introduire le langage PHP

### Quizz (Correction)



### Réponses

#### Question 13 :

A quoi sert le DNS ?

1. Permet de traduire une adresse logique en un nom de domaine
2. Permet de traduire un nom de domaine en une adresse logique
3. Permet de transplanter un système dans un autre système
4. Permet de traduire une adresse IP v4 en IP v6

#### Question 14 :

Quel est le protocole dédié à la navigation WEB ?

1. HTTP
2. FTP
3. CTP
4. DNS

#### Question 15 :

Qu'est-ce qui caractérise une architecture Client/Serveur ?

1. Un terminal passif
2. Un navigateur
3. Une gestion locale de l'interface graphique
4. Un poste qui dispose d'un logiciel envoyant des requêtes à un serveur

#### Question 16 :

Une architecture logicielle est un document qui décrit l'expression du besoin sous la forme de spécifications textuelles précises ?

1. Oui
2. Non

#### Question 17 :

Une architecture logicielle est un document qui décrit les composants logiciels et leurs dépendances mutuelles ?

1. Oui
2. Non

## 02 - Introduire le langage PHP

### Quizz (Correction)



#### Réponses

##### Question 18 :

Le logiciel qui joue le rôle du client dans le modèle client/serveur du Web s'appelle un :

1. Explorateur de fichier
2. Navigateur
3. Moteur de recherche

##### Question 19 :

Un logiciel de serveur Web peut être installé sur n'importe quel ordinateur ?

1. Oui
2. Non

##### Question 20 :

Complétez les blancs.

Le protocole HTTP sert à définir les règles du dialogue entre le **client** et le **serveur** dans un échange Web.



## CHAPITRE 2

### Introduire le langage Php

1. Définition du PHP
2. Généralités sur les fonctionnalités du PHP

## 02 - Introduire le langage PHP

### Généralités sur les fonctionnalités du PHP

#### 4 types de domaines différents



##### Langage de script côté serveur

- C'est l'utilisation la plus traditionnelle et la principale utilisation de PHP, nécessite trois composants pour l'utiliser :
  - un analyseur PHP (CGI « Common Gateway Interface » ou module serveur)
  - un serveur Web lié à PHP
  - un navigateur Web.
- Vous pouvez accéder au programme PHP à l'aide d'un navigateur Web.

##### Écrire des applications clientes graphiques

- PHP-GTK permet d'écrire de tels programmes.
- Vous pouvez également choisir d'écrire des applications très portables dans ce langage.
- PHP-GTK est une extension de PHP qui n'est pas inclus dans la distribution de base.

##### Langage de programmation en ligne de commande

- Vous pouvez écrire un script PHP et l'exécuter à partir de la ligne de commande sans utiliser de serveur Web ni de navigateur.
- Tout ce dont vous avez besoin est le fichier exécutable PHP.
- Cette utilisation convient très bien aux scripts qui s'exécutent régulièrement (avec cron sous Unix ou Linux) ou aux gestionnaires de tâches (sous Windows).
- Ces scripts peuvent également être utilisés pour effectuer des opérations sur des fichiers texte.

#### Définition

- C'est un **langage côté serveur** : c'est donc la machine qui héberge le site web qui va exécuter le PHP, et non la machine ou le navigateur du visiteur.
- **Indépendant de la plate-forme utilisée** puisqu'il est exécuté côté serveur et non côté client.
- C'est un **langage interprétré** : les instructions sont traitées directement par le serveur (pas de compilation)
- PHP supporte de nombreux protocoles **comme LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM** (sous Windows) et encore d'autres.
- Vous pouvez ouvrir des sockets réseau, et interagir avec n'importe quel autre protocole.
- **PHP supporte le format complexe WDDX**, qui permet de communiquer entre tous les langages web.
- En terme d'interconnexion, PHP supporte aussi les objets Java, et les utilise de manière transparente comme objets intégrés.
- PHP standardise toutes les extensions XML sur la solide base de libxml2, et étend le jeu de fonctionnalités en ajoutant le support de SimpleXML, XMLReader et XMLWriter.
- **PHP est devenu le langage le plus utilisé dans le développement Web**, en raison de son interaction avec la base de données MySQL et de son intégration facile avec le code XHTML.
- PHP permet de développer des scripts suivant le modèle procédural et/ou le modèle objet.
- La dernière version PHP8.1.1 publié le 17 déc. 2021

## 02 - Introduire le langage PHP

### Généralités sur les fonctionnalités du PHP



#### Syntaxe

- PHP est utilisable sur la majorité des systèmes d'exploitation, comme Linux, de nombreuses variantes Unix (incluant HP-UX, Solaris et OpenBSD), Microsoft Windows, macOS, RISC OS, .... Il fonctionne sous forme de module, ou comme processeur CGI. Un script ou programme CGI est un programme comme un autre mais dont la sortie est dédiée à un navigateur, par exemple un programme shell, C ou Perl qui sort du HTML.
- Une page PHP est tout simplement une page HTML. Cela signifie que celle-ci comporte des balises placées au sein d'une structure HTML classique.
- Les balises PHP sont incluses à l'intérieur du code HTML, mais elles sont repérées par les mentions suivantes : <?php et ?>
- L'extension d'un fichier contenant du code php doit être .php (si vous avez une page nommée index.htm par exemple et que vous y insérez du code PHP, il vous faudra ainsi la renommer en index.php)
- Il y a plusieurs formes de basiles pour interpréter un code PHP, mais une seule de ces formes est recommandée pour des raisons de compatibilité.

<?php ... ?> // La forme recommandé  
<? ... ?>  
<script language="php"> ... </script>  
<% ... %>

## 02 - Introduire le langage PHP

### Généralités sur les fonctionnalités du PHP



#### Configuration

- PHP nécessite au moins **Windows 2008/Vista**. Soit 32-Bit soit 64-bit (AKA X86 or X64). PHP ne fonctionne pas sur Windows RT/WOA/ARM).
- À partir de PHP 7.2.0 Windows 2008 et Vista ne sont plus supporté.
- **PHP nécessite le Visual C runtime (CRT)**. CRT x86 pour la compilation PHP x86 et le CRT x64 pour la compilation PHP x64. L'installateur CRT supporte les options de ligne de commande /quiet et /norestart, aussi, vous pouvez les utiliser dans vos scripts.
- Le Microsoft Visual C++ Redistributable pour Visual Studio 2019 convient pour toutes les versions de PHP
- Pour vos développements des pages web dynamiques avec PHP, vous aurez besoin :
  - **Un OS** : Windows, Linux
  - **Un serveur web** avec un interpréteur PHP
  - **Un éditeur de texte** pour écrire des scripts
  - Éventuellement une **base de données**

## 02 - Introduire le langage PHP

### Généralités sur les fonctionnalités du PHP



#### Caractéristiques

Exemples d'application :

- Wiki (MediaWiki, WikiNi, DokuWiki...)
- Forum (phpBB, IPB, punBB...)
- Systèmes de gestion de blog (Dotclear, WordPress...)
- Systèmes de gestion de contenu (appelés aussi CMS) (Spip, ExpressionEngine, Drupal, Xoops...)
- Administration de bases de données (phpMyAdmin, phpPgAdmin, Eskuel...)
- Frameworks (Zend Framework, CodeIgniter, CakePHP, Symfony, etc.)

## 02 - Introduire le langage PHP

### Généralités sur les fonctionnalités du PHP



#### Caractéristiques

Dans le cas de PHP comme langage serveur, les combinaisons les plus courantes sont :

LAMP	WAMP	MAMP	LEMP	LAPP	XAMPP
Linux Apache MySQL PHP	Windows Apache MySQL PHP. WAMP s'installe généralement par le biais d'un seul logiciel qui intègre Apache, MySQL et PHP, comme par exemple EasyPHP, VertrigoServ, WampServer ou UwAmp	Mac OS Apache MySQL PHP	Linux Nginx(EngineX) MySQL PHP	Linux Apache PostgreSQL PHP	Apache MySQL Perl PHP ; le X indique que le logiciel est multiplate-forme

Il est également utilisé par de nombreux systèmes de gestion de contenu (CMS) populaires tels que WordPress, Drupal et Joomla.

## 02 - Introduire le langage PHP

### Références et ressources



### Références et ressources

PHP :

<https://www.php.net/manual/fr/index.php>

<https://www.techno-science.net/glossaire-definition/PHP.html>

Langage de Script :

<https://kinsta.com/fr/blog/langages-de-script/>

PHP Framework :

<https://geekflare.com/fr/php-frameworks/>



## CHAPITRE 3

### Préparer l'environnement de développement

Ce que vous allez apprendre dans ce chapitre :

- Choix des serveurs web
- Installation du serveur (Apache)
- Configuration du serveur (ports, services, ...)
- Installation des extensions nécessaires (VS Code)

03 heures



## CHAPITRE 3

### Préparer l'environnement de développement

1. Choix des serveurs web
2. Installation du serveur (Apache)
3. Configuration du serveur (ports, services, ...)
4. Installation des extensions nécessaires (VS Code)

## 03 - Préparer l'environnement de développement

### Choix des serveurs web

#### Classement

- Pourcentages de sites Web utilisant divers serveurs Web répartis par classement.
- Un site Web peut utiliser plusieurs serveurs Web.
- Les rapports inter-technologies n'incluent que les technologies avec plus de 1 % d'utilisation pour garantir la signification statistique des résultats.
- Apache est utilisé par 31.4% de tous les sites Web dont nous connaissons le serveur Web. Il est utilisé par 27.0% de tous les sites Web dont nous connaissons le serveur Web et qui se classent dans le top 1 000 000.

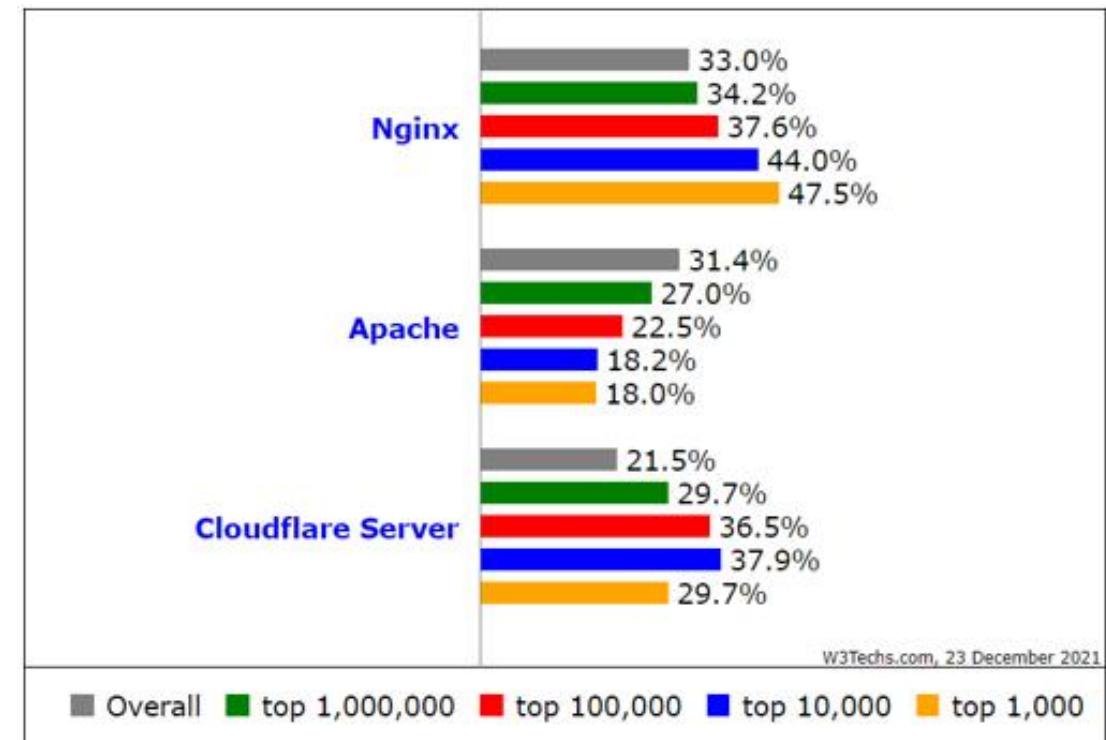


Fig. : Pourcentages de sites Web utilisant divers serveurs Web répartis par classement.

## 03 - Préparer l'environnement de développement

### Choix des serveurs web

#### Classement

- Ce schéma montre les pourcentages de sites Web utilisant différents serveurs Web.
- Un site Web peut utiliser plusieurs serveurs Web.
- Apache est utilisé par 31.4% de tous les sites Web dont nous connaissons le serveur Web.

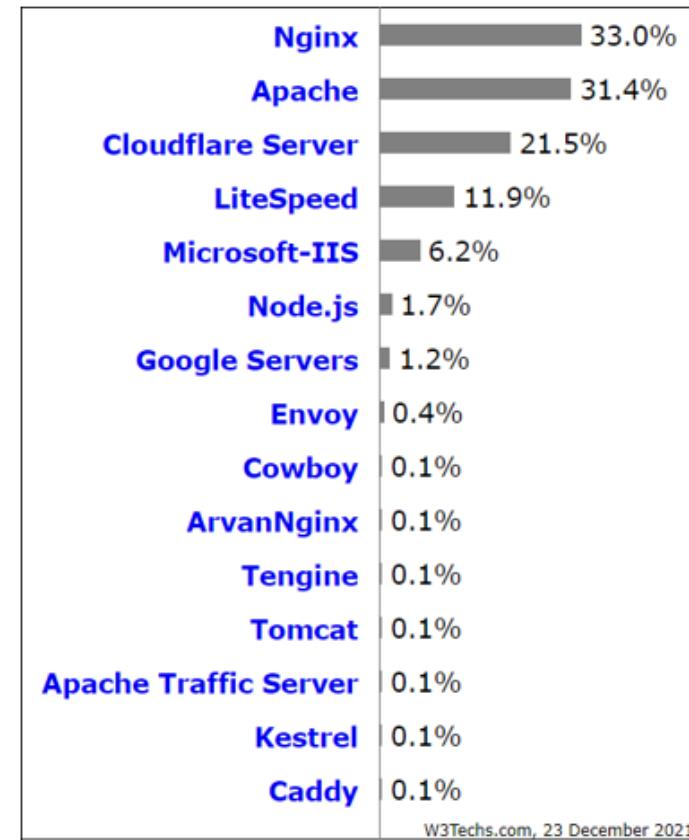


Fig. : Pourcentages de sites Web utilisant divers serveurs Web.

## 03 - Préparer l'environnement de développement

### Choix des serveurs web



#### Apache Vs NGINX

- La différence entre Apache et NGINX est présente dans le tableau ci-dessous :
- NGINX a été créé pour résoudre le problème des dix mille connexions simultanées. Cependant, dans notre cas on est loin d'atteindre ce chiffre. Apache offre des avantages par rapport à Nginx, tels que sa configuration simple, ses nombreux modules et son environnement convivial ainsi, nous allons choisir Apache comme serveur Web.

Apache	NGINX
Apache fonctionne sur tous les systèmes de type Unix tels que Linux, BSD, etc. et prend entièrement en charge Windows.	Nginx fonctionne sur des systèmes modernes de type Unix ; cependant, il a un support limité pour Windows.
Apache utilise une approche multi-thread pour traiter les demandes des clients.	Nginx suit une approche pilotée par les événements pour répondre aux demandes des clients.
Apache ne peut pas gérer simultanément plusieurs requêtes avec un trafic Web important.	Nginx peut gérer simultanément et efficacement plusieurs demandes de clients avec des ressources matérielles limitées.
Apache traite le contenu dynamique au sein du serveur Web lui-même.	Nginx ne peut pas traiter le contenu dynamique de manière native.
Apache est conçu pour être un serveur Web.	Nginx est à la fois un serveur Web et un serveur proxy.
Les modules sont dynamiquement chargés ou déchargés, ce qui le rend plus flexible.	Étant donné que les modules ne peuvent pas être chargés dynamiquement, ils doivent être compilés dans le logiciel de base lui-même.
Un seul thread ne peut traiter qu'une seule connexion.	Un seul thread peut gérer plusieurs connexions.
Les performances d'Apache pour le contenu statique sont inférieures à celles de Nginx.	Nginx peut exécuter simultanément des milliers de connexions de contenu statique deux fois plus rapidement qu'Apache et utilise un peu moins de mémoire.



## CHAPITRE 3

### Préparer l'environnement de développement

1. Choix des serveurs web
2. **Installation du serveur (Apache)**
3. Configuration du serveur (ports, services, ...)
4. Installation des extensions nécessaires (VS Code)

## 03 - Préparer l'environnement de développement

### Installation du serveur (Apache)



#### Téléchargement

- En mode local (sur une machine) ou en intranet, on peut utiliser des plateformes telles que Wamp, Lamp ou Easyphp, dans lesquelles Apache est intégré en tant que serveur web. Ces plateformes ont aussi MySQL (interfacé avec PhpMyAdmin) comme système de gestion de base de données, et PHP, comme langage de développement.
- Il s'appuie sur les protocoles HTTP ou HTTPS. Il existe sous deux formes binaires: avec et sans le support de (Open) SSL. La deuxième étant déconseillée si l'on compte mettre en place HTTPS.
- Apache peut être installé sous plusieurs systèmes d'exploitation: Windows, Mac, et la famille Unix (Linux, OpenBSD, Mac OS X, etc.)
- La plate-forme Windows de base pour l'exécution d'Apache 2.4 est Windows 2000 ou supérieur.
- Télécharger Apache La dernière version stable : 2.4.52 (publié le 2021-12-20) depuis le lien officiel : <https://httpd.apache.org/download.cgi>



## 03 - Préparer l'environnement de développement

### Installation du serveur (Apache)



#### Apache en tant que service

- Apache Service Monitor (Moniteur du service Apache) vous donne la possibilité de voir et gérer l'état de tous les services Apache installés sur toutes les machines du réseau.
- Pour pouvoir gérer un service Apache avec le moniteur, vous devez installer le service soit automatiquement au cours de l'installation, soit manuellement.
- Pour installer Apache en tant que service Windows NT, vous devez à partir de la ligne de commandes et depuis le sous-répertoire Apache bin écrire la commande :

**httpd.exe -k install**

- Vous pouvez installer plusieurs services Apache sur votre ordinateur, vous devrez spécifier le nom du service comme suivant :

**httpd.exe -k install -n "Nom-service"**

- Une fois vous changer le nom durant l'installation d'un service, vous devrez aussi le spécifier pour toute opération comportant l'option –k



## CHAPITRE 3

### Préparer l'environnement de développement

1. Choix des serveurs web
2. Installation du serveur (Apache)
- 3. Configuration du serveur (ports, services, ...)**
4. Installation des extensions nécessaires (VS Code)

## 03 - Préparer l'environnement de développement

### Configuration du serveur (ports, services, ...)



#### Introduction

- La configuration du serveur HTTP Apache s'effectue via des fichiers textes.
- La configuration d'Apache est enregistrée dans les fichiers du sous-répertoire conf.
- Le répertoire par défaut des fichiers de configuration est `/usr/local/apache2/conf` ou `/usr/local/httpd2/conf` selon les distributions.
- Le nom du fichier de configuration par défaut est en général `httpd.conf`, mais peut varier en fonction des distributions tierces du serveur.
- En général, sous Windows la configuration par défaut est suffisante.
- La plate-forme choisie permet de gérer l'édition du fichier de configuration.
- Pour la configuration, il est conseillé de se connecter en tant qu'utilisateur et non en tant qu'admin. Il faut accorder les droits de lecture et d'écriture pour la configuration.

## 03 - Préparer l'environnement de développement

### Configuration du serveur (ports, services, ...)



#### Modules

- Un module peut exporter ou importer certains symboles comme des variables, des fonctions ou des classes. Il désigne originellement un fichier de code de programmation ou un fichier de bibliothèque statique ou dynamique. (source Wikipédia)
- Liste des modules : <https://httpd.apache.org/docs/current/mod/>

Le module core	Le module mod_rewrite	Le module mod_ssl	Le module mod_proxy	Le module mod_actions	Le module mod_cgi
Fonctionnalités de base du serveur HTTP Apache toujours disponibles.	Ce module fournit un moteur de réécriture à base de règles permettant de réécrire les URLs des requêtes à la volée.	Chiffrement de haut niveau basé sur les protocoles Secure Sockets Layer (SSL) et Transport Layer Security (TLS).	Serveur mandataire / passerelle multi-protocole.	Exécution des scripts CGI (Common Gateway Interface) en fonction du type de média ou de la méthode de requête.	Exécution des scripts CGI. D'ailleurs l'exécution proprement dite des pages développées avec des langages de programmation tels que PHP est assurée par des modules ou des CGI.

## 03 - Préparer l'environnement de développement

### Configuration du serveur (ports, services, ...)



#### Quelques commandes

- Vous pouvez installer Apache en tant que service Windows NT à partir de la ligne de commandes et depuis le sous-répertoire Apache bin comme suit :

**httpd.exe -k install**

- Pour supprimer un service Apache, vous devez écrire la commande suivante :

**httpd.exe -k uninstall**

- Pour supprimer un service Apache particulier , vous devez écrire la commande suivante :

**httpd.exe -k uninstall -n « Nom service »**

- Le démarrage, le redémarrage et l'arrêt d'un service Apache s'effectuent via le Moniteur de Service Apache, ou en utilisant des commandes telles que NET START Apache2.4 et NET STOP Apache2.4, ou encore via le gestionnaire de services standard de Windows.

- Pour démarrer un service Apache installé, vous devez écrire la commande suivante :

**httpd.exe -k start -n « Nom-Service »**

- Pour arrêter un service Apache via les options de lignes de commande, vous devez écrire l'une des commandes suivantes :

**httpd.exe -k stop -n « Nom-Service »**

**httpd.exe -k shutdown -n « Nom-Service »**

- Pour redémarrer un service en exécution, vous devez écrire la commande suivante :

**httpd.exe -k restart -n « Nom-Service »**

## 03 - Préparer l'environnement de développement

### Configuration du serveur (ports, services, ...)



#### Environnement global

- **ServerRoot chemin de répertoire** : Racine du répertoire d'installation du serveur, par défaut /usr/local/apache.
- **Listen [adresse IP:]numéro port [protocole]** : Les adresses IP et ports sur lesquels le serveur écoute.
- **SecureListen [adresse-IP:]num-port nom-certificat [MUTUAL]** : Active le chiffrement SSL pour le port spécifié.
- **ServerName [protocole://]nom-de-domaine|adresse-ip[:port]** : Nom d'hôte et port que le serveur utilise pour s'authentifier lui-même.
- **SuexecUserGroup Utilisateur Groupe** : L'utilisateur et le groupe sous lesquels les programmes CGI doivent s'exécuter.
- **DBDKeep nombre** : Nombre maximum de connexions maintenues, par défaut 2.
- **DBDMax nombre** : Nombre maximum de connexions, par défaut 10.

## 03 - Préparer l'environnement de développement

### Configuration du serveur (ports, services, ...)



#### Serveur

- **ServerAdmin adresse électronique|URL** : L'adresse électronique que le serveur inclut dans les messages d'erreur envoyés au client.
- **ServerName [protocole://]nom-de-domaine|adresse-ip[:port]** : Nom d'hôte et port que le serveur utilise pour s'authentifier lui-même. On pourrait aussi y mettre son adresse (par exemple : 192.168.8.17). Le nom du serveur n'est valide que si on a un serveur DNS fonctionnel. Il est souhaitable de mettre l'IP.
- **HostnameLookups On|Off[Double]** : Active la recherche DNS sur les adresses IP des clients : c'est le mode d'enregistrement des clients. Quand il est activé ( On ), le client est enregistré par son nom. En mode Off qui est par défaut, l'enregistrement est fait sur l'adresse IP.
- **DocumentRoot chemin répertoire** : Racine principale de l'arborescence des documents visible depuis Internet, c'est le répertoire qui contient les sites Web hébergés sur le serveur. On peut le changer, où y mettre tous les sites Web que le serveur doit héberger. Par défaut : "/usr/local/apache/ +
- **AddLanguage symbole-langue extension [extension] ...** : Associe l'extension de nom de fichier donnée à la langue spécifiée, pour permettre à l'utilisateur de lire dans son navigateur les pages Web dans la langue mentionnée.



## CHAPITRE 3

### Préparer l'environnement de développement

1. Choix des serveurs web
2. Installation du serveur (Apache)
3. Configuration du serveur (ports, services, ...)
- 4. Installation des extensions nécessaires (VS Code)**

## 03 - Préparer l'environnement de développement

### Installation des extensions nécessaires (VS Code)



#### Pré-requis

- httpd peut être compilé sous Windows en utilisant une chaîne de compilation basée sur cmake, ou à partir de fichiers projet Visual Studio.
- Pour compiler Apache, l'environnement doit satisfaire aux conditions suivantes :
  - Espace disque : minimum de 200Mo d'espace disque disponible.
  - Microsoft Visual C++ 6.0 (Visual Studio 97) ou supérieur : Apache peut être compilé en utilisant l'outil ligne de commande, ou depuis l'espace de travail IDE Visual Studio
  - Le SDK de la plate-forme Windows mis à jour, février 2003 ou plus récent
- L'IDE Visual Studio cherche le chemin de awk.exe que dans la variable PATH, ou dans le chemin des exécutables spécifié par l'option de menu Tools -> Options -> (Projects ->) Directories.
- Pour simplifier la compilation d'Apache depuis l'espace de travail IDE de Developer Studio, l'espace de travail Visual Studio Apache.dsw est fourni.

## 03 - Préparer l'environnement de développement

### Installation des extensions nécessaires (VS Code)



#### VS Code

Visual Studio Code est un excellent éditeur pour le développement PHP. Vous obtenez des fonctionnalités telles que la mise en évidence de la syntaxe et la correspondance des crochets, IntelliSense (achèvement du code) et des extraits de code prêts à l'emploi et vous pouvez ajouter plus de fonctionnalités grâce aux extensions VS Code créées par la communauté.

Téléchargez Visual Studio Code pour Windows à partir du site officiel :

<https://code.visualstudio.com/download>

Une fois téléchargé, exécutez le programme d'installation (VSCodeUserSetup-{version}.exe).

Par défaut, VS Code est installé sous

C:\users\{username}\AppData\Local\Programs\Microsoft VS Code.

.NET Framework 4.5.2 ou supérieur est requis pour VS Code.

Pour Windows 7, assurez-vous d'avoir au moins .NET Framework 4.5.2 installé.



The screenshot shows a code editor window for a file named 'HelloWorldController.php'. The code defines a class 'HelloWorldController' with a method 'getHelloWorld()'. A cursor is positioned at the end of the word 'Response' in the line 'return new Response();'. An Intellisense dropdown menu is open, listing various 'Response' related suggestions. To the right of the editor, there is a sidebar with some code snippets and settings.

```
<?php
namespace App\Controller;
class HelloWorldController
{
    public function getHelloWorld()
    {
        return new Response();
    }
}
use Symfony\Component\HttpFoundation\Response;
```

```
use Symfony\Component\HttpFoundation\Response
<?php
public function __construct(
    string $content = '',
    int $status = 200,
    \array $headers = []
) { }
```

Fig. : Exemple affichage code php écrit dans VsCode

## 03 - Préparer l'environnement de développement

### Installation des extensions nécessaires (VS Code)



#### VS Code

Installer les extensions à partir du site officiel : <https://marketplace.visualstudio.com/>

The screenshot shows the Visual Studio Marketplace interface. The search bar at the top contains the text 'php'. Below it, the results section displays 492 items. The results are organized in a grid of cards, each representing a different extension. The extensions listed include:

- PHP Debug (Ydebug, xdebug.org) - 5.1M, 4.5 stars, FREE
- PHP Intelephense (Ben Mertijn, intelephense.com) - 5.1M, 4.5 stars, FREE
- PHP Extension Pack (Xdebug, xdebug.org) - 2.8M, 4.5 stars, FREE
- PHP DocBlocker (Neil Brayfield) - 725K, 4.5 stars, FREE
- Format HTML in PHP (rif2k) - 947K, 4.5 stars, FREE
- PHP Server (brapira) - 652K, 4.5 stars, FREE
- PHP Formatter (Sophisticode) - 596K, 4 stars, FREE
- PHP Namespace Reso (Mehedi Hassan) - 597K, 4.5 stars, FREE
- phpfmt - PHP formatter (kokorin) - 660K, 4 stars, FREE
- php cs fixer (junitly) - 601K, 4 stars, FREE
- PHP IntelliSense (Damjan Cvetko) - 294K, 4 stars, FREE
- PHP intellisense for codeigniter (small) - 631K, 4 stars, FREE
- PHP Getters & Setters (phproberto) - 269K, 4 stars, FREE
- PHP Tools for VS Code (DEVSENSE.com) - 268K, 4 stars, FREE
- phpcs (Ioannis Kappas) - 1.2M, 4 stars, FREE
- PHP Symbols (lin yang) - 160K, 4 stars, FREE
- PHP Debug (Robert Lu) - 114K, 4 stars, FREE
- PHP Sniffer (wongjin) - 942K, 4 stars, FREE

# PARTIE 1 : Introduire la notion Client/Serveur

## Références et ressources



### Références et ressources

Choix du serveur web :

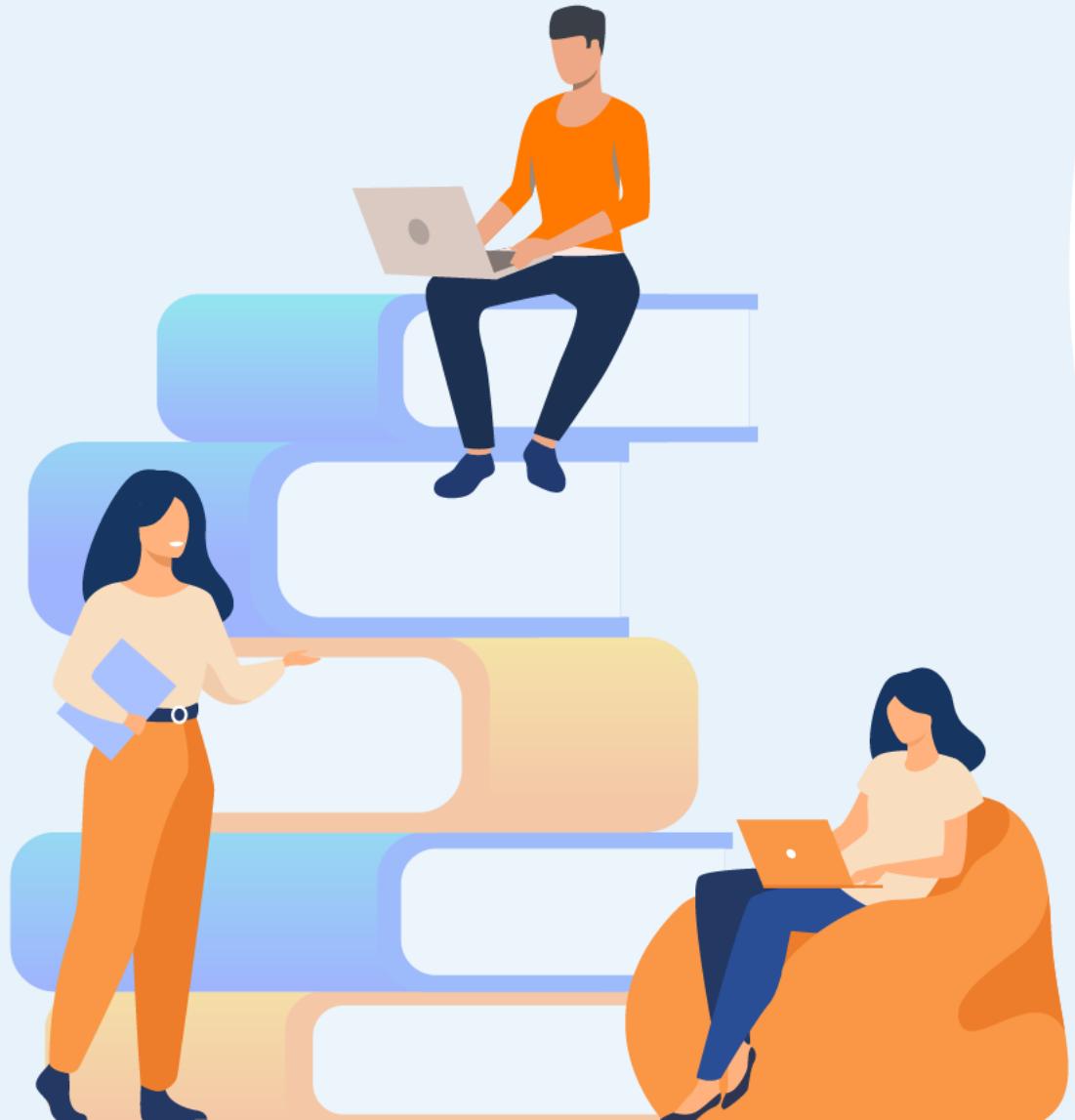
[https://w3techs.com/technologies/cross/web\\_server/ranking](https://w3techs.com/technologies/cross/web_server/ranking)  
[https://w3techs.com/technologies/overview/web\\_server](https://w3techs.com/technologies/overview/web_server)  
<https://www.javatpoint.com/difference-between-apache-and-nginx>

Apache :

<https://httpd.apache.org/docs/current/>  
<https://httpd.apache.org/docs/2.4/platform/windows.html#cust>  
<https://httpd.apache.org/docs/2.4/mod/quickreference.html>

Vs Code :

<https://code.visualstudio.com/docs/languages/php>



## PARTIE 2

### Programmer en PHP

**Dans ce module, vous allez :**

- Maitriser le langage PHP
- Manipuler les tableaux
- Manipuler les fichiers
- Maitriser le concept de l'orienté objet en PHP
- Manipuler les méthodes magiques en PHP



**48 heures**



## CHAPITRE 1

### Maitriser le langage PHP

**Ce que vous allez apprendre dans ce chapitre :**

- Structure générale d'un script PHP
- Manipulation des Variables/constantes/Affectation
- Manipulation des types de données
- Instructions de sortie
- Contrôles de flux et boucles
- Formulaires simples
- Transmission de variables (GET, POST)
- Variables d'environnement
- Redirection entre pages
- Fonctions sur les chaînes de caractères et les dates



**12 heures**



# CHAPITRE 1

## Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

# 01 - Maîtriser le langage PHP

## Structure générale d'un script PHP

### Balises PHP

- Lorsque PHP traite un fichier, il cherche les balises d'ouverture et de fermeture (`<?php` et `?>`) qui délimitent le code qu'il doit interpréter.
- En version 7, PHP accepte deux syntaxes pour les balises :

```
<?php ... ?>
```

```
<? ... ?>
```

- Avant la version 7, PHP acceptait deux syntaxes supplémentaires pour les balises :

```
<script language="php"> ... </script>
```

```
<% ... %>
```

- Tout ce qui est à l'extérieur de la balise PHP est transmis tel quel au navigateur; seul les instructions PHP sont interprétées par le moteur PHP.

```
<?php
//Commentaire sur une seule ligne, style C++

/*
    Ceci est un
    commentaire
    sur
    plusieurs lignes
*/

# Ceci est un commentaire style shell sur une seule ligne
?>
```



### Commentaires

- `//` : permet d'ajouter un commentaire sur une seule ligne
- `#` : permet d'ajouter un commentaire sur une seule ligne
- `/* ... */` : permet d'ajouter un commentaire sur plusieurs lignes

### La fonction echo

echo - Affiche une chaîne de caractères

PHP inclus une balise ouvrante echo courte <?= qui est un raccourci au code plus verbeux <?php echo

### Séparation des instructions

PHP requiert que les instructions soient terminées par un point-virgule à la fin de chaque instruction.

```
<?php  
  
echo 'Ceci est un test';  
echo 'Ceci est un autre test';  
  
?>  
  
<?=br/>  
'Et un test final';  
  
?>
```



## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. **Manipulation des Variables/constantes/Affectation**
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

### Les variables essentielles

En PHP, les variables sont représentées par un signe dollar "\$" suivi du nom de la variable.

Un nom de variable valide doit respecter les règles suivantes (après le symbole \$) :

- Le nom est sensible à la casse : \$a et \$A sont deux variables distinctes.
- Le premier caractère doit être une lettre ou un souligné \_
- À partir du deuxième caractère seul les lettres, chiffres ou soulignés sont acceptés.

Exemples :

```
<?php  
$var; //nom de variable valide  
$Var; //nom de variable valide  
$4vars; //nom de variable invalide: commencer avec un nombre n'est pas autorisé  
$_var; //nom de variable valide  
$état; //nom de variable valide  
$éta+; //nom de variable invalide: les caractères spéciaux comme + ne sont pas autorisés  
&var Vat; //nom de variable invalide: l'espace n'est pas autorisé  
?>
```

### Variables prédéfinies

Les variables "superglobales" sont disponibles quel que soit le contexte du script.

#### **`$_SERVER`**

Variables de serveur et d'exécution

#### **`$_FILES`**

Variable de téléchargement de fichier via HTTP

#### **`$_ENV`**

Variables d'environnement

#### **`$http_response_header`**

En-têtes de réponse HTTP

#### **`$GLOBALS`**

Référence toutes les variables disponibles dans un contexte global

#### **`$_GET`**

Variables HTTP GET

#### **`$_REQUEST`**

Variables de requête HTTP

#### **`$_COOKIE`**

Cookies HTTP

#### **`$argc`**

Le nombre d'arguments passés au script

#### **`$_POST`**

Variables HTTP POST

#### **`$_SESSION`**

Variables de session

#### **`$php_errormsg`**

Le dernier message d'erreur

#### **`$argv`**

Tableau d'arguments passés au script

### Portée des variables

- Une variable globale doit être déclarée à l'intérieur de chaque fonction afin de pouvoir être utilisée dans cette fonction.
  - Il suffit d'utiliser le mot clé **global** avant la variable pour l'utiliser.
- Une variable statique a une portée locale uniquement, mais elle ne perd pas sa valeur lorsque le script appelle la fonction.
  - Il suffit d'utiliser le mot clé **static** avant la variable ou l'attribut ou la méthode pour l'utiliser.
- Les propriétés statiques sont accédées en utilisant l'opérateur de résolution de portée **::**
- Les variables statiques peuvent être assignées des valeurs qui sont issue d'expression constante, mais les expressions dynamiques, tel que les appels de fonctions, résulteront en une erreur d'analyse.

```
<?php
function test() {
    $var = "variable locale";

    echo '$var dans le contexte global : ' . $GLOBALS['var'] . "\n";
    echo '$var dans le contexte courant : ' . $var . "\n";
}

$var = "Exemple de contenu";
test();
?>
```

```
<?php
function test()
{
    //la variable $i est initialisée uniquement lors du premier appel à la fonction
    static $i = 0;
    echo $i;
    // à chaque fois la fonction est appelée, elle affichera une valeur de $i + 1
    $i++;
}
?>
```

# 01 - Maîtriser le langage PHP

## Manipulation des Variables/constantes/Affectation



### Variables dynamiques

- Une variable dynamique prend la valeur d'une variable et l'utilise comme nom d'une autre variable, en utilisant le " \$\$ " précédent la variable.
- Les accolades peuvent aussi être utilisées, pour clairement délimiter le nom de la propriété

```
<?php
$a = 'Bonjour' ;
$$a = 'étudiant' ;
Echo "$a ${$a}" ; // Bonjour étudiant
Echo "$a $Bonjour" ; // Bonjour étudiant
?>
```

The screenshot shows a web-based IDE interface for PHP. The URL in the address bar is `onecompiler.com/php/3xrswjc6y`. The main area displays a file named `HelloWorld.php` with the following code:

```
1 <?php
2 $a = "Bonjour" ;
3 $$a = "étudiant" ;
4 Echo "$a ${$a}" ; // Bonjour étudiant
5 Echo "\n\r";
6 Echo "$a $Bonjour"; // Bonjour étudiant
?>
```

To the right of the code editor, there are tabs for `NEW`, `PHP`, and a `RUN` button. Below the code editor, there is an input field labeled `STDIN` with the placeholder `Input for the program (Optional)`. The output window below shows the results of the execution:

Output:  
Bonjour étudiant  
Bonjour étudiant

- Les superglobales ne peuvent pas être utilisées comme variables dynamiques dans les fonctions ou les méthodes des classes.

### Exemples fonctions de gestion des variables

#### **floatval**

Convertit une chaîne en nombre à virgule flottante

#### **empty**

Détermine si une variable est vide

#### **get\_resource\_id**

Retourne un entier identifiant une ressource

#### **is\_null**

Indique si une variable vaut null

#### **get\_defined\_vars**

Liste toutes les variables définies

#### **get\_resource\_type**

Retourne le type de ressource

#### **is\_object**

Détermine si une variable est de type objet

#### **unset**

Détruit une variable

#### **print\_r**

Affiche des informations lisibles pour une variable

#### **is\_array**

Détermine si une variable est un tableau

#### **isset**

Détermine si une variable est déclarée et est différente de null

#### **var\_dump**

Affiche les informations d'une variable

#### **var\_export**

Retourne le code PHP utilisé pour générer une variable

### Les constantes

- Une constante est un identifiant (un nom) qui représente une valeur simple.
- Les constantes sont sensibles à la casse.
- Par convention, les constantes sont toujours en majuscule.
- Les constantes peuvent être définies en utilisant le mot clé **const** ou en utilisant la fonction **define()**.
- une constante n'est pas prefixée d'un \$
- Pour vérifier qu'une constante est définie, utiliser la fonction **defined()**.
- Si une constante indéfinie est utilisée une Error est renvoyée lancée.
- Contrairement aux constantes définies en utilisant l'instruction define(), les constantes définies en utilisant le mot-clé const doivent être déclarées au plus haut niveau du contexte, car elles seront définies au moment de la compilation.

```
<? Php  
  
const test = 'Bonjour les étudiants !';  
echo test; //Affiche Bonjour les étudiants !  
  
?>
```

### Constantes magiques

- Une constante magique est une constante prédéfinie dans PHP qui se change en fonction du contexte.

Nom	Description
<code>__LINE__</code>	La ligne courante dans le fichier.
<code>__FILE__</code>	Le chemin complet et le nom du fichier courant avec les liens symboliques résolus. Si utilisé pour une inclusion, le nom du fichier inclus est retourné.
<code>__DIR__</code>	Le dossier du fichier. Si utilisé dans une inclusion, le dossier du fichier inclus sera retourné. C'est l'équivalent de dirname(__FILE__). Ce nom de dossier ne contiendra pas de slash final, sauf si c'est le dossier racine.
<code>__FUNCTION__</code>	Le nom de la fonction, ou {closure} pour les fonctions anonymes.
<code>__CLASS__</code>	Le nom de la classe courante. Le nom de la classe contient l'espace de nom dans lequel cette classe a été déclarée. Lorsqu'elle est utilisée dans une méthode de trait, __CLASS__ est le nom de la classe dans laquelle le trait est utilisé.
<code>__TRAIT__</code>	Le nom du trait. Le nom du trait inclut l'espace de nom dans lequel il a été déclaré.
<code>__METHOD__</code>	Le nom de la méthode courante.
<code>__NAMESPACE__</code>	Le nom de l'espace de noms courant.

Tab. : Les constantes magiques de PHP source <https://www.php.net/manual/fr/language.constants.magic.php>

### Les opérateurs d'affectation

- L'opérateur d'affectation le plus simple est le **signe =**
- Il signifie que l'opérande de gauche se voit affecter la valeur de l'expression qui est à droite du signe égal.
- L'affectation par référence se fait grâce au **signe &**

Exemples :

```
<?php
$a = 2; // $a est maintenant égal à 2
$b = 3; // $b est maintenant égal à 3
$c = $a + $b; // $c est maintenant égal à 5
$d = $$c; // $d est maintenant égal à 5

$e = "Bonjour ";
$e .= " les étudiants"; // affecte la valeur "Bonjour les étudiants" à la variable e

$c += $a*2 // $c est maintenant égal à 9 de même que $d est maintenant égal à 9
?>
```



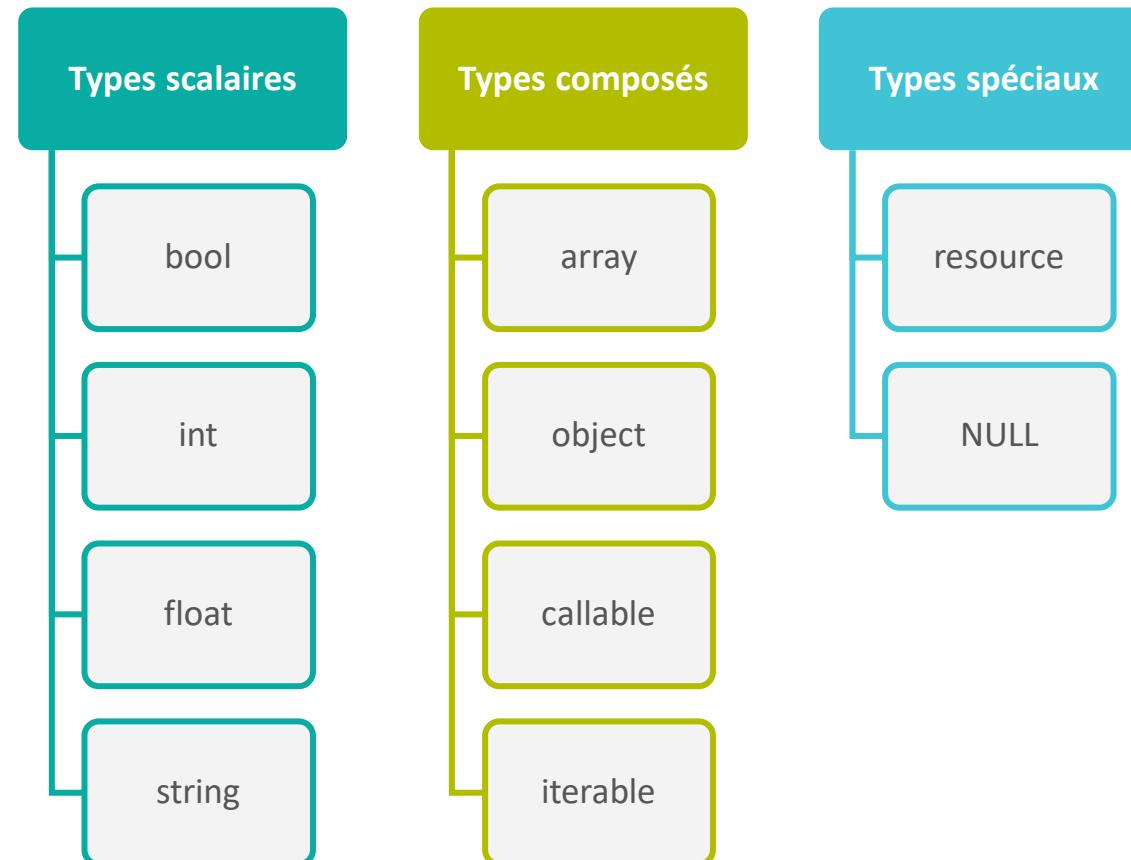
# CHAPITRE 1

## Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
- 3. Manipulation des types de données**
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

### Introduction

- Les variables PHP pourront stocker différents types de valeurs comme des nombres ou des tableaux. Par abus de langage, on parle des "types variables" de PHP.
- PHP supporte 10 types basiques :



# 01 - Maîtriser le langage PHP

## Manipulation des types de données



### Type booléen

- Il peut avoir **true** ou **false**.
- ==** est un opérateur qui teste l'égalité et retourne un booléen.
- Pour convertir explicitement une valeur en booléen, utilisez **(bool)** ou **(boolean)**.
- Lors d'une conversion en booléen, les valeurs suivantes sont considérées comme false :
  - false**
  - l'entier 0, les nombres à virgule flottante 0.0 et -0.0
  - la chaîne vide, et la chaîne "0"
  - un tableau avec aucun élément
  - le type spécial **NULL** (incluant les variables non définies)
  - les objets SimpleXML créés depuis des éléments vide sans attributs.

```
<?php
$bool_val = (bool)true;
echo $bool_val; // Affiche : 1

$bool_val2 = (bool)false;
echo $bool_val2; // N'affiche rien

$bool_exp1 = (bool)false;
echo $bool_exp1 ? 'true' : 'false'; // Affiche : false

$bool_exp2 = (bool)false;
echo json_encode($bool_exp2); // Affiche : false

$bool_exp3 = (bool)false;
echo var_export($bool_exp3); // Affiche : false

$bool_exp4 = (bool)false;
echo (int)$bool_exp4; // Affiche : 0

$bool_exp5 = (bool)false;
var_dump($bool_exp5); // Affiche : bool(false)
?>
```

# 01 - Maîtriser le langage PHP

## Manipulation des types de données

### Type entier

- Un entier est un nombre appartenant à l'ensemble  $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ .
- Les entiers peuvent être spécifiés en notation:
  - Décimale (base 10)
  - Hexadécimale (base 16) : Pour utiliser la notation hexadécimale, précédez le nombre de 0x
  - Octale (base 8) : Pour utiliser cette notation, précédez le nombre d'un 0 (zéro). À partir PHP 8.1.0, la notation octale peut être précédé avec 0o ou 0O
  - Binaire (base 2) : Pour utiliser la notation binaire, précédez le nombre de 0b.
- L'opérateur de négation peut être utilisé pour désigner un entier négatif.
- À partir de PHP 7.4.0, les entiers littéraux peuvent contenir des underscores (\_) entre les chiffres, pour une meilleure lisibilité des littéraux. Ces underscores sont supprimés par le scanner de PHP. Exemple : \$a = 1\_203\_489 // c'est la même chose que \$a = 1203489
- Pour convertir explicitement une valeur en un entier, utiliser soit le mot-clé (int), soit (integer).
  - Depuis un booléen : false correspond à 0, et true correspond à 1.
  - Depuis un nombre décimal : le nombre sera arrondi vers zéro.

```
<?php  
// un nombre décimal  
$a = 1234;  
  
// un nombre octal (équivalent à 83 en décimal)  
$a = 0123;  
  
// un nombre octal (à partir de PHP 8.1.0)  
$a = 0o123;  
  
// un nombre hexadecimale (équivalent à 26 en décimal)  
$a = 0x1A;  
  
// un nombre binaire (équivalent à 255 en decimal)  
$a = 0b11111111;  
  
// un nombre décimal (à partir de PHP 7.4.0)  
$a = 1_234_567;  
?>
```



### Type nombre à virgules

- À partir de PHP7.4.0
  - LNUM [0-9]+(\_[0-9]+)\*
  - DNUM ([0-9]\*(\_[0-9]+)\*[\.]{LNUM}) | ({LNUM}[\.][0-9]\*(\_[0-9]+)\*)
  - EXPONENT\_DNUM ({LNUM} | {DNUM}) [eE][+-]? {LNUM})
- Le type « nombre décimal » ou Float en anglais.
- Pour convertir explicitement une valeur en un float, utiliser le mot-clé (float).
  - Depuis une chaîne de caractères : Si une chaîne est numérique ou numérique de tête alors elle sera transformée en sa valeur flottante correspondante, sinon elle sera convertie à zéro.
  - Depuis un booléen : false correspond à 0.0, et true correspond à 1.0
  - Depuis d'autres types : la conversion est effectuée d'abord en int puis en float

```
<?php
$a = 1.234;
$b = 1.2e3;
$c = 7E-10;
$d = 1_234.567; // à partir de PHP 7.4.0
$epsilon = 0.00001;
$f = 1.23456780;
?>
```

### Type chaîne de caractères

La façon la plus simple de spécifier une chaîne de caractères est de l'entourer de guillemets simples (le caractère ').

- Pour spécifier un guillemet simple littéral, il doit être échappé à l'aide d'un antislash (\ \ )
- Pour spécifier un antislash littéral, doublez-le (\ \ \ )
- Les séquences d'échappement pour les caractères spéciaux ne seront pas interprétées
- Un retour à la ligne dans la chaîne de caractère permet d'avoir une nouvelle ligne.

Si la chaîne de caractères est entourée de guillemets doubles (le caractère ") :

- Les noms de variables seront interprétés.
- \n : Fin de ligne (LF ou 0x0A (10) en ASCII)
- \r : Retour à la ligne (CR ou 0x0D (13) en ASCII)
- \t : Tabulation horizontale (HT or 0x09 (9) en ASCII)
- \v : Tabulation verticale (VT ou 0x0B (11) en ASCII)
- \e : échappement (ESC or 0x1B (27) en ASCII)
- \f : Saut de page (FF ou 0x0C (12) en ASCII)
- \\ : Antislash
- \\$ : Signe dollar
- " : Guillemet double

```
<?php
echo 'ceci est une chaîne simple';
echo "une autre chaîne simple ";
echo 'Vous pouvez également ajouter des nouvelles lignes
dans vos chaînes
de cette façon';
echo « Des nouvelles lignes \ndans vos chaînes »;
// Affiche : Ahmed a dit : "J'ai un autre exemple ici"
echo 'Ahmed a dit : "J\'ai un autre exemple ici"';
// Affiche : Voulez-vous supprimer C:\*.*?
echo 'Voulez-vous supprimer C:\\*.*?';
// Affiche : Voulez-vous supprimer C:\*.*?
echo 'Voulez-vous supprimer C:\*.*?';
// Affiche : Ceci n'affichera pas \n de nouvelle ligne
echo 'Ceci n\'affichera pas \n de nouvelle ligne';
$traitees = "égales à";
$iici = 2;
// Affiche : Les variables ne seront pas $traitees $iici
echo 'Les variables ne seront pas $traitees $iici';
// Affiche : Les variables ne seront pas égales à 2
echo "Les variables ne seront pas $traitees $iici";
```

# 01 - Maîtriser le langage PHP

## Manipulation des types de données



### Type tableau

- Syntaxe d'un tableau est :
  - **clé => valeur**
  - La **clé** peut être soit un int, soit une chaîne de caractères. Les nombres à virgule flottante seront aussi modifiés en entier.
  - La **valeur** peut être de n'importe quel type.
  - Structure de langage **array()**. La syntaxe courte emplace array() par []
  - Le tableau prend un nombre illimité de paramètres, chacun séparé par une virgule

### Exemples:

```
<?php
// signifie la 0 => Salut, 1 => Aloha, 2=>Hello
$tableau1 = array("Salut", "Aloha", "Hello");
Echo $tableau1[0]; // Afficher : Salut
// signifie la 0 => Salut, 1 => Aloha, 2=>Hello
$tableau2 = ["Salut", "Aloha", "Hello"];
Echo $tableau2[0]; // Afficher : Salut

$tableau3 = array(
    "0" => "Salut" ,
    "1" => "Aloha" ,
    "2" => "Hello" ,
);
Echo $tableau3[2]; // Afficher : Hello
$tableau4 = [
    "0" => "Salut" ,
    "1" => "Aloha" ,
    "2" => "Hello" ,
];
Echo $tableau4[2]; // Afficher : Hello
?>
```

### Type objet

Pour créer un nouvel objet, le mot clé **new** est utilisé afin d'instancier une class.

Une **class** définit le comportement d'un objet. La classe ne contient aucune donnée.

Un **objet** est une instance d'une classe qui contient des données.

Un **membre** est une variable qui appartient à un objet.

Une **méthode** est une fonction qui appartient à un objet et a accès à ses membres.

Un **constructeur** est une méthode spécifique qui est exécutée lorsqu'un objet est créé.

### Type callable

- Les fonctions de rappel peuvent être des fonctions simples ou des méthodes objet, y compris des méthodes de classe statique.
- Les fonctions de rappel peuvent être identifiées via le type **callable**.
- Une méthode d'un objet instancié est passée comme un tableau contenant un objet à l'index 0, et le nom de la méthode à l'index 1.
- Les méthodes de classe statiques peuvent également être transmises sans instancier un objet de cette classe en passant le nom de la classe au lieu de l'objet à l'index 0, ou en passant 'NomClass::NomMethode'

### Type itérable

- Un iterable est un pseudo-type introduit en PHP 7.1. Il accepte n'importe quel tableau ou objet implémentant l'interface Traversable. (Interface permettant de détecter si une classe peut être parcourue en utilisant **foreach**).
- Tous les tableaux sont des itérables.
- Un tableau peut être utilisé comme argument d'une fonction qui nécessite un iterable.

Un itérateur doit avoir ces méthodes :

- **current()** : Renvoie l'élément sur lequel le pointeur est entrain de pointer.
- **key()** : Renvoie la clé associée à l'élément courant dans la liste.
- **next()** : Déplace le pointeur vers l'élément suivant dans la liste.
- **rewind()** : Déplace le pointeur sur le premier élément de la liste.
- **valid()** : Il doit retourner false si le pointeur interne ne pointe sur rien (par exemple, si next() est appelé à la fin de la liste). Dans tous les autres cas, il retourne vrai.

### Type ressource

Une ressource est une variable spéciale, contenant une référence vers une ressource externe.

Exemples :

Ressource	Définition
AddressInfo	AddressInfo (sockets extension)
Connexion cubrid persistante	Connexion persistante à une base de données CUBRID
curl_multi	cURL multi handle
Dba	Lien vers une base de données DBA
Fdf	Fichier FDF
interbase query	Requête Interbase
mysql result	Résultat MySQL
Connexion oci8	Connexion sur une base de données Oracle
odbc result	Résultat ODBC
OpenSSL X.509	Clé publique

Tab. : Types des ressources PHP source <https://www.php.net/manual/fr/resource.php>

### Type NULL

Il y a une seule valeur de type null, et c'est la constante insensible à la casse null.

Une variable est considérée comme null si la constante null lui est assignée ou n'a pas encore reçu de valeur.

Transyper une variable vers null est OBSOLÈTE à partir de PHP 7.2.0, et SUPPRIMÉE à partir de PHP 8.0.0

### Modification de types

**(int), (integer) :**  
modification en int

**(bool), (boolean) :**  
modification en bool

**(float), (double), (real) :**  
modification en float

**(string) :**  
modification en string

**(array) :**  
modification en array

**(object) :**  
modification en object



## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
- 4. Instructions de sortie**
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

### Envoi vers le navigateur

- Trois fonctions permettent d'envoyer du texte vers le navigateur :
  - `echo(string ...$expressions) : void` → Affiche une chaîne de caractères.
  - `print(string $expression) : int` → Affiche une chaîne de caractères.
  - `printf(string $format, mixed ...$values) : int` → Affiche une chaîne de caractères formatée.
- **Echo :**
  - N'a pas de valeur de retour.
  - Peut prendre plusieurs arguments (bien que cette utilisation soit rare).
  - Est légèrement plus rapide que print.
- **Print :**
  - Renvoie la valeur 1, il peut donc être utilisé dans des expressions.
  - Peut prendre un seul argument.
- **Le type mixed :**
  - La valeur peut être de n'importe quelle valeur.



## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. **Contrôles de flux et boucles**
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

### Introduction

- L'ordre de flux est l'ordre dans lequel les instructions d'un programme sont exécutées.
- Les instructions qui changent l'ordre d'exécution sont :
  - Les itérations.
  - Les instructions conditionnelles.
  - Les ruptures de séquences.
- Les instructions de contrôle sont utilisées pour contrôler l'exécution d'un programme :
  - Instructions de boucle : elles sont utilisées pour exécuter un bloc de code à x reprises.
  - Instructions de sélection : elles vous permettent d'exécuter un bloc de code spécifique dans plusieurs blocs de code en fonction de critères de sélection.
  - Instructions de saut : Ces instructions sont utilisées pour passer d'un bloc de code à un autre.

### Syntaxe alternative

Pour les fonctions de contrôle if, while, for, foreach et switch. PHP propose une autre manière de rassembler des instructions à l'intérieur d'un bloc.

Le principe est de remplacer l'accolade d'ouverture par deux points (:) et l'accolade de fermeture par, respectivement, endif;, endwhile;, endfor;, endforeach;, ou endswitch;

Vous ne pouvez pas utiliser différentes syntaxes dans le même bloc de contrôle.

### L'instruction if else

- Si l'expression vaut true, PHP exécutera l'instruction et si elle vaut false, l'instruction sera ignorée.
- Les instructions après le else ne sont exécutées que si l'expression du if est false.

```
if (expression)
    commandes
else
    commandes
```

### L'instruction elseif

- L'expression elseif est exécutée seulement si le if précédent et tout autre elseif précédent sont évalués comme false, et que votre elseif est évalué à true.
- Le « elseif » et « else if » sont traités de la même façon seulement quand des accolades sont utilisées. (Exemple : `else if ($a == $b){ echo $a." égal ".$b;}` est une synthèse correcte, tandis que `else if ($a == $b) : echo $a." égal ".$b;` est une synthèse qui génère une erreur PHP. De la même façon, les deux syntaxes `elseif ($a == $b){ echo $a." égal ".$b;}` et `elseif ($a == $b) : echo $a." égal ".$b;` sont toutes les deux correctes.

```
if (expression)
    commandes
elseif (expression)
    commandes
elseif (expression)
    commandes
elseif (expression)
    commandes
else
    commandes
```

### L'instruction switch

- L'instruction switch équivaut à une série d'instructions if.
- Un cas spécial est default. Ce cas est utilisé lorsque tous les autres cas ont échoué.
- PHP continue d'exécuter les instructions jusqu'à la fin du bloc d'instructions du switch, ou bien dès qu'il trouve l'instruction break.
- Dans une commande switch, une condition n'est évaluée qu'une fois, et le résultat est comparé à chaque case.

Exemples :

```
<?php
switch ($i) {
    case 0:
        echo "i égal 0";
        break;
    case 1:
        echo "i égal 1";
        break;
    case 2:
        echo "i égal 2";
        break;
    default:
        echo "i n'est ni égal à 2, ni à 1, ni à 0.";
}
?>
```

### L'instruction match

- De la même manière qu'une instruction switch, l'instruction match a une expression de sujet qui est comparée à plusieurs alternatives.
- Différences entre match et switch :
  - match évaluera une valeur un peu comme les expressions ternaires.
  - la comparaison match est un contrôle d'identité (==) plutôt qu'un contrôle d'égalité faible (==) comme switch.
  - match renvoie une valeur.

```
<?php
$valeur_retour = match (expression) {
    expression_conditionnelle_unique => expression_retour,
    expression1_conditionnelle, expression2_conditionnelle => expression_retour,
};
```

### L'instruction while

- PHP exécute l'instruction tant que l'expression de la boucle while est évaluée comme true. Si l'expression du while est false avant la première itération, l'instruction ne sera jamais exécutée.

```
while (expression) :  
    commandes  
endwhile;
```

### L'instruction do-while

- La principale différence par rapport à la boucle while est que la première itération de la boucle do-while est toujours exécutée.

```
do  
    commandes  
while (expression);
```

### L'instruction for

- **expr1** : est évaluée (exécutée), au début de la boucle.
- **expr2** : est évaluée, au début de chaque itération. Si l'évaluation vaut true, la boucle continue et les commandes sont exécutées. Si l'évaluation vaut false, l'exécution de la boucle s'arrête.
- **expr3** : est évaluée (exécutée), à la fin de chaque itération.
- Les expressions peuvent éventuellement être laissées vides ou peuvent contenir plusieurs expressions séparées par des virgules.

```
for (expr1; expr2; expr3)
    commandes
```

### L'instruction foreach

- foreach ne fonctionne que pour les tableaux et les objets.
- La forme suivante passe en revue le tableau iterable\_expression. À chaque itération, la valeur de l'élément courant est assignée à \$value.

```
foreach (iterable_expression as $value){
    //commandes
}
```

- La forme suivante assignera en plus la clé de l'élément courant à la variable \$key à chaque itération.

```
foreach (iterable_expression as $key =>
    $value){
    //commandes
}
```

### L'instruction break

- L'instruction break permet de sortir d'une structure for, foreach, while, do-while ou switch.
- break accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées doivent être interrompues.

```
<?php
$i = 0;
while (++$i) {
    switch ($i) {
        case 5:
            echo "At 5<br />\n";
            break 1; /* Termine uniquement le switch. */
        case 10:
            echo "At 10; quitting<br />\n";
            break 2; /* Termine le switch et la boucle while. */
        default:
            break;
    }
}
?>
```

### L'instruction continue

- L'instruction continue est utilisée dans une boucle pour contourner l'instruction de l'itération actuelle et poursuivre l'exécution sous la condition évaluée, en commençant l'itération suivante.
- continue accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées doivent être interrompues.
- La structure continue s'applique aux structures switch et se comporte de la même manière que break.

```
<?php
for ($i = 0; $i <= 10; ++$i) {
    if ($i % 2)
        continue;
    print "$i\n"; // Affiche : 0 2 4 6 8 10
}
?>
```



## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
- 6. Formulaires simples**
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

### Formulaires HTML

- Lorsqu'un formulaire est envoyé à un script PHP, toutes les variables du formulaire seront automatiquement disponibles dans le script.
- Exemple d'un formulaire qui demande de remplir le nom et prénom avec un bouton OK. Lorsque un visiteur remplit le formulaire, et clique sur le bouton **OK**, le fichier **action.php** est appelé :

```
<form action="action.php" method="post">
  <p>Votre nom : <input type="text" name="nom" /></p>
  <p>Votre prénom : <input type="text" name="prenom" /></p>
  <p><input type="submit" value="OK"></p>
</form>
```

### Formulaires XForms

- XForms est un dialecte XML permettant de créer des formulaires en ligne conçus pour être utilisés avec HTML, XHTML, WML ou SVG.

#### Exemple :

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://www.w3.org/2002/xforms">
<head>
  <title>Exemple</title>
  <f:model>
    <f:submission action= "action.php" method="post" id="s"/>
  </f:model>
</head>
<body>
  <p>
    <input ref="nom"><label>Votre nom:</label></input>
    <input ref="prenom"><label>Votre prénom:</label></input>
    <f:submit submission="s"><f:label>OK</f:label></f:submit>
  </p>
</body>
</html>
```



# CHAPITRE 1

## Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. **Transmission de variables (GET, POST)**
8. Variables d'environnement
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

# 01 - Maîtriser le langage PHP

## Transmission de variables (GET, POST)

### \$\_GET

- Elle donne les valeurs des informations indiquées dans l'url.
- Les informations après le point d'interrogation ? d'une URL sont en réalité des données que l'on fait transiter d'une page à une autre.

#### Exemple :

- Le site s'appelle : lesite.com
- Ma page PHP de utilisateur : action.php
- Pour accéder à la page de l'utilisateur, l'URL sera :  
`http://www.lesite.com/action.php`
- Pour envoyer les informations de l'utilisateur à la page php nous utiliserons :  
`http://www.lesite.com/action.php?nom=Rasmus&prenom=Lerdorf`
- Les paramètres sont séparés par le symbole &

#### Exemple de formulaire :

```
<form action="action.php" method="get">
<p>Votre nom : <input type="text" name="nom" /></p>
<p>Votre prénom : <input type="text" name="prenom" /></p>
<p><input type="submit" value="OK"></p>
</form>
```

- Quand l'utilisateur clique sur le bouton « OK », l'URL envoyée au serveur sera visible par le visiteur dans la barre d'adresse comme ceci:  
`http://www.lesite.com/action.php?nom=Rasmus&prenom=Lerdorf`
- Le fichier « action.php » peut maintenant employer la variable super globale `$_GET` pour récupérer les données du formulaire

```
<?php
echo "Bonjour $_GET["nom"] $_GET["prenom"] et bienvenue";
?>
```

### \$\_POST et \$\_REQUEST

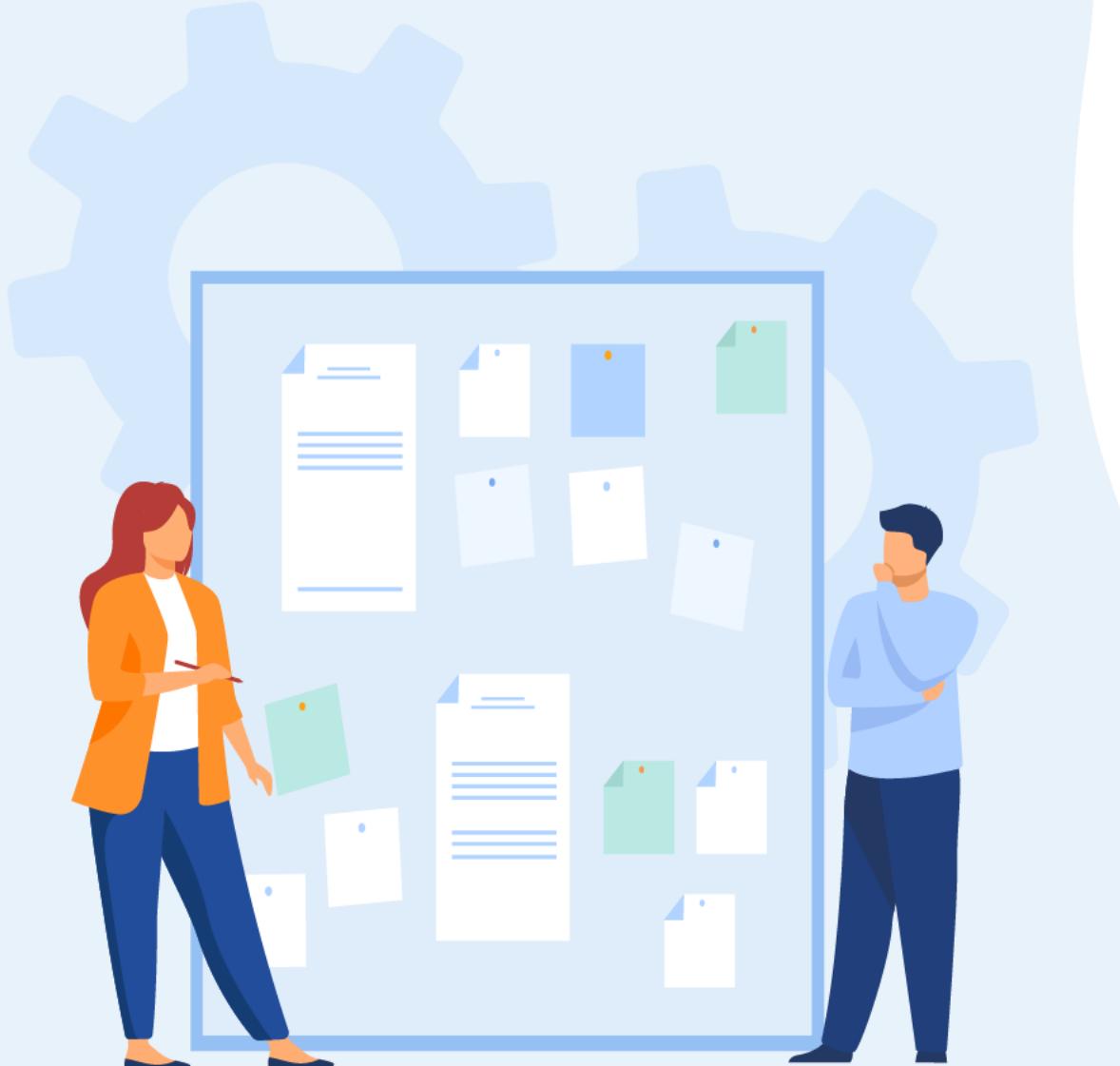
- PHP \$\_POST est une variable super globale PHP qui est utilisée pour collecter des données de formulaire après avoir soumis un formulaire HTML avec method="post".
- \$\_POST est également utilisé pour passer des variables.
- Lorsque nous envoyons des variables via un formulaire, la variable est récupérée de cette façon \$\_POST['Variable'] et non plus \$Variable

Exemple de formulaire envoyé :

```
<form action="action.php" method="post">
<p>Votre nom : <input type="text" name="nom" /></p>
<p>Votre prénom : <input type="text" name="prenom" /></p>
<p><input type="submit" value="OK"></p>
</form>
```

- Pour accéder aux variables du formulaire :

```
<?php
echo $_POST['nom'];
echo $_REQUEST['prenom'];
?>
```



# CHAPITRE 1

## Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
- 8. Variables d'environnement**
9. Redirection entre pages
10. Fonctions sur les chaînes de caractères et les dates

# 01 - Maîtriser le langage PHP

## Variables d'environnement



### **\$\_SERVER**

`$_SERVER` est un tableau contenant des informations comme les en-têtes, dossiers et chemins du script.

Liste dans <https://www.php.net/manual/fr/reserved.variables.server.php>

Exemples :

#### **PHP\_SELF**

Le nom du fichier du script en cours d'exécution, par rapport à la racine web

#### **GATEWAY\_INTERFACE**

Numéro de révision de l'interface CGI du serveur

#### **SERVER\_ADDR**

L'adresse IP du serveur sous lequel le script courant est en train d'être exécuté

#### **SERVER\_NAME**

Le nom du serveur hôte qui exécute le script suivant

#### **SERVER\_SOFTWARE**

Chaîne d'identification du serveur, qui est donnée dans les en-têtes lors de la réponse aux requêtes

#### **REQUEST\_METHOD**

Méthode de requête utilisée pour accéder à la page

#### **HTTP\_ACCEPT\_LANGUAGE**

Contenu de l'en-tête Accept-Language: de la requête courante, si elle existe

#### **HTTP\_CONNECTION**

Contenu de l'en-tête Connection: de la requête courante, si elle existe

#### **HTTP\_HOST**

Contenu de l'en-tête Host: de la requête courante, si elle existe

#### **HTTPS**

Défini à une valeur non-vide si le script a été appelé via le protocole HTTPS

#### **REMOTE\_PORT**

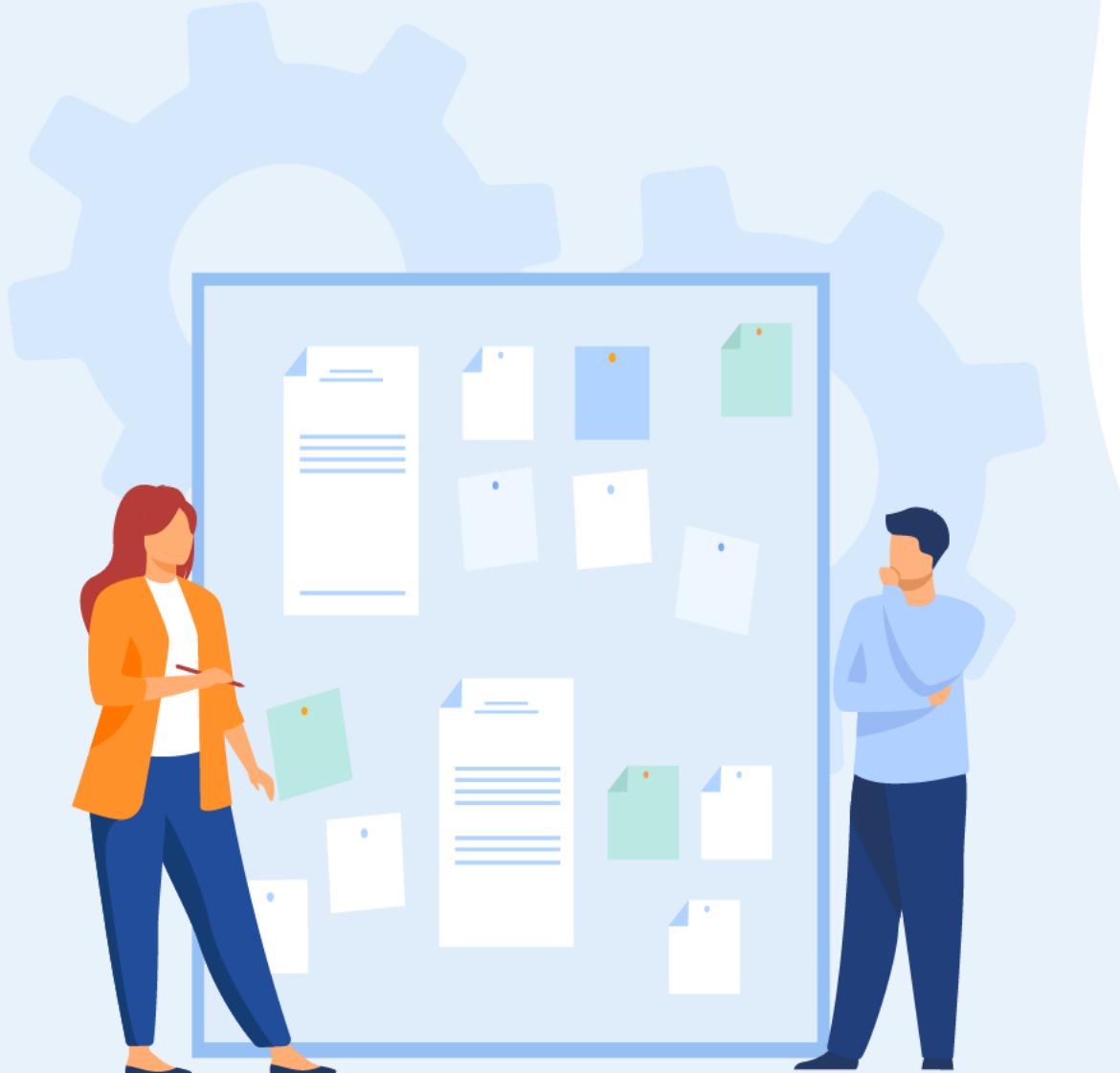
Le port utilisé par la machine cliente pour communiquer avec le serveur web

#### **REMOTE\_USER**

L'utilisateur authentifié

### Variable prédefinies

- **`$_FILES`** - Variable de téléchargement de fichier via HTTP :
  - `$_FILES['nom_de_la_variable']['name']` : Le nom original du fichier qui provient de la machine du client
  - `$_FILES['nom_de_la_variable']['type']` : Le type MIME du fichier
  - `$_FILES['nom_de_la_variable']['size']` : La taille du fichier en bytes (soit 8 bits ou un octet)
  - `$_FILES['nom_de_la_variable']['tmp_name']` : Le nom temporaire du fichier stocké sur le serveur
  - `$_FILES['nom_de_la_variable']['error']` : Le code erreur associé à l'upload
- **`$_SESSION`** - Variables de session. Liste des fonctions dans <https://www.php.net/manual/fr/ref.session.php>
- **`$_ENV`** - Variables d'environnement
- **`$_COOKIE`** - Cookies HTTP



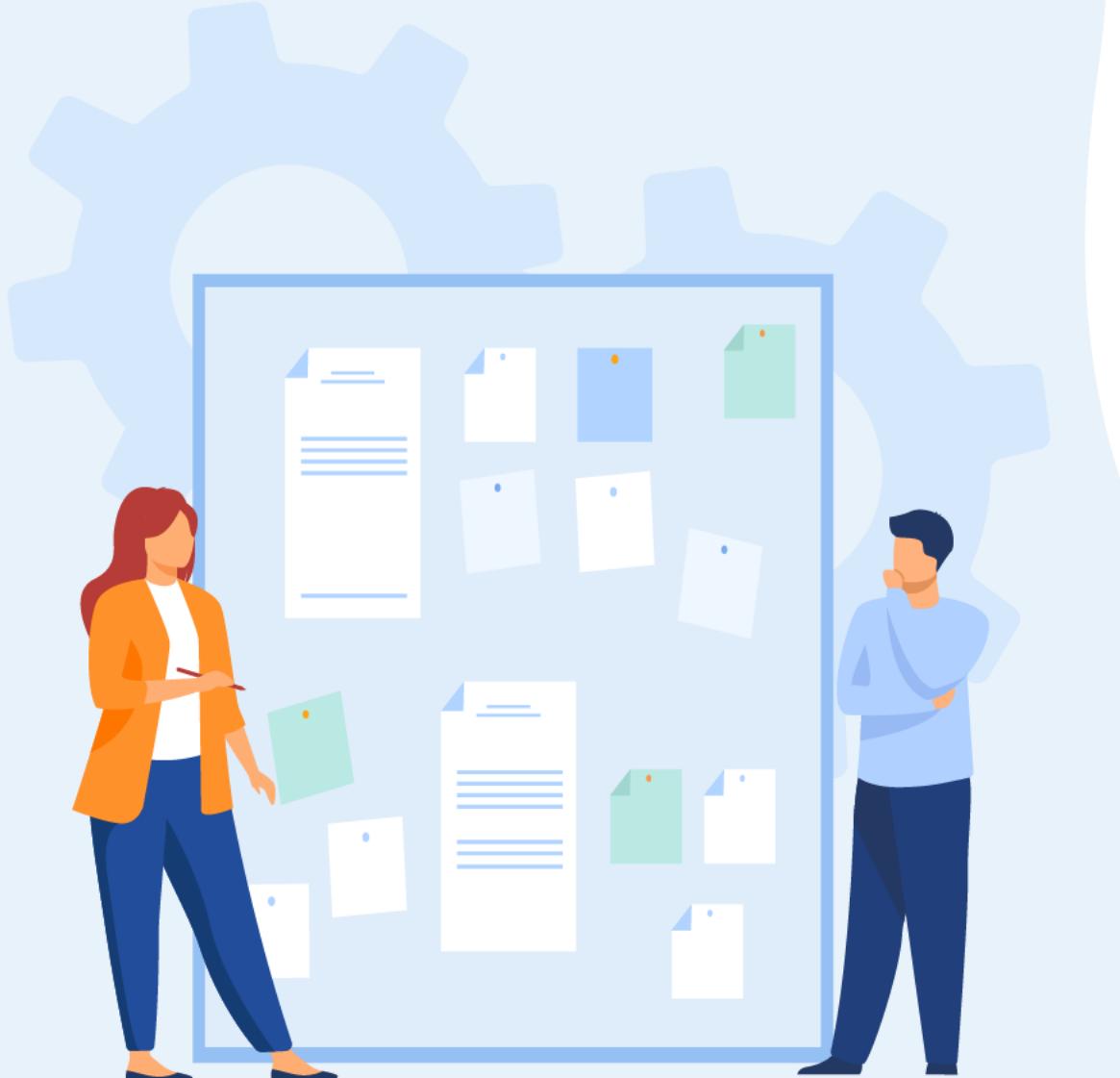
## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
- 9. Redirection entre pages**
10. Fonctions sur les chaînes de caractères et les dates

### header

- **header** = Envoie un en-tête HTTP brut
- La syntaxe est `header(string $header, bool $replace = true, int $response_code = 0): void`
- header : l'en-tête. Il y a deux en-têtes spéciaux :
  - Le premier commence par la chaîne "HTTP/" (insensible à la casse), qui est utilisée pour signifier le statut HTTP à envoyer  
**Exemple :** `header("$_SERVER["SERVER_PROTOCOL"]." 404 Not Found")`
  - Le deuxième type d'appel spécial est "Location:". Il renvoie un en-tête au client et un statut REDIRECT (302) au navigateur
- replace : paramètre optionnel. Indique si la fonction header() doit remplacer un en-tête précédemment publié ou ajouter un autre en-tête du même type. Par défaut, le nouvel en-tête écrasera le précédent, mais vous pouvez forcer plusieurs en-têtes pour le même type d'en-tête si vous passez false dans ce paramètre.
- response\_code : Force le code réponse HTTP à la valeur spécifiée.
- header() doit être appelée avant que le moindre contenu ne soit envoyé, soit par des lignes HTML habituelles dans le fichier, soit par des affichages PHP.



## CHAPITRE 1

### Maitriser le langage PHP

1. Structure générale d'un script PHP
2. Manipulation des Variables/constantes/Affectation
3. Manipulation des types de données
4. Instructions de sortie
5. Contrôles de flux et boucles
6. Formulaires simples
7. Transmission de variables (GET, POST)
8. Variables d'environnement
9. Redirection entre pages
- 10. Fonctions sur les chaînes de caractères et les dates**

# 01 - Maîtriser le langage PHP

## Fonctions sur les chaînes de caractères et les dates



### Fonctions sur les chaînes de caractères

Référence officielle : <https://www.php.net/manual/fr/ref.strings.php>

#### **addcslashes**

Ajoute des slash dans une chaîne, à la mode du langage C

#### **addslashes**

Ajoute des antislashes dans une chaîne

#### **chop - Alias de rtrim**

Supprime les espaces (ou d'autres caractères) de fin de chaîne

#### **ltrim**

Supprime les espaces (ou d'autres caractères) de début de chaîne

#### **count\_chars**

Retourne des statistiques sur les caractères utilisés dans une chaîne

#### **similar\_text**

Calcule la similarité de deux chaînes

#### **strlen**

Calcule la taille d'une chaîne

# 01 - Maîtriser le langage PHP

## Fonctions sur les chaînes de caractères et les dates



### Fonctions sur les chaînes de caractères

#### **strtoupper**

Renvoie une chaîne en majuscules

#### **strr**

Remplace des caractères dans une chaîne

#### **mb\_convert\_case**

Modifie la casse d'une chaîne

#### **mb\_strtolower**

Met tous les caractères en minuscules

#### **mb\_strtoupper**

Met tous les caractères en majuscules

#### **mb\_strwidth**

Retourne la taille d'une chaîne

#### **preg\_grep**

Retourne un tableau avec les résultats de la recherche

#### **preg\_match\_all**

Expression rationnelle globale

### Fonctions sur les Date/Heure

Référence officielle : <https://www.php.net/manual/fr/ref.datetime.php>

#### **checkdate**

Valide une date grégorienne : checkdate(int \$month, int \$day, int \$year): bool

#### **DateTime::add -- date\_add**

Ajoute une durée à un objet DateTime

#### **DateTime::setDate -- date\_date\_set**

Assigne la date

#### **DateTime::diff --**

#### **DateTimeImmutable::diff --**

#### **DateTimeInterface::diff -- date\_diff**

Retourne la différence entre deux objets DateTime

#### **DateTime::sub -- date\_sub**

Soustrait une durée à un objet DateTime

#### **getdate**

Retourne la date/heure

# 01 - Maîtriser le langage PHP

## Fonctions sur les chaînes de caractères et les dates



WEBFORCE  
BE THE CHANGE

### Fonctions sur les Date/Heure

- date = Formate une date/heure locale

Caractères pour le paramètre format	Description	Exemple de valeurs retournées
<i>Jour</i>	---	---
d	Jour du mois, sur deux chiffres (avec un zéro initial)	01 à 31
D	Jour de la semaine, en trois lettres (et en anglais - par défaut : en anglais, ou sinon, dans la langue locale du serveur)	Mon à Sun
j	Jour du mois sans les zéros initiaux	1 à 31
I ('L' minuscule)	Jour de la semaine, textuel, version longue, en anglais	Sunday à Saturday
N	Représentation numérique ISO 8601 du jour de la semaine	1 (pour Lundi) à 7 (pour Dimanche)
S	Suffixe ordinal d'un nombre pour le jour du mois, en anglais, sur deux lettres	st, nd, rd ou th. Fonctionne bien avec j
w	Jour de la semaine au format numérique	0 (pour dimanche) à 6 (pour samedi)
z	Jour de l'année	0 à 365
<i>Semaine</i>	---	---
W	Numéro de semaine dans l'année ISO 8601, les semaines commencent le lundi	Exemple : 42 (la 42ème semaine de l'année)

Tab : Exemple de caractère reconnus dans le paramètre format

## Chapitre 2

### Traiter les données en PHP



**Ce que vous allez apprendre dans ce chapitre :**

- Traitement des tableaux (simple, Associatif)
- Manipulation de fichier (chargement, Suppression, téléchargement)



**08 heures**



## CHAPITRE 2

### Traiter les données en PHP

1. **Traitement des tableaux (simple, Associatif)**
2. Manipulation de fichier (chargement, Suppression, téléchargement)

## 02 - Traiter les données en PHP

### Traitement des tableaux (simple, Associatif)

#### Tableaux indexés ou simples

Un tableau indexé contient des indices numériques qui indexent le contenu souhaité. Ces indexées commencent par défaut de 0 et s'incrémentent de 1 à chaque fois.

##### Exemples :

```
$tab=array("tab1", "tab2", "tab3");  
  
$tab=array();  
  
    $tab[0]= "tab1";  
  
    $tab[1]= "tab2";  
  
    $tab[2]= "tab3";
```

#### Tableaux associatifs

Un tableau associatif contient des clés qui sont des chaînes de caractères.

##### Exemples :

```
$tab=array("cle1" => "tab1", "cle2" => "tab2", "cle3" => "tab3");  
  
$tab=array(  
    "cle1" => "tab1",  
    "cle2" => "tab2",  
    "cle3" => "tab3"  
);  
  
$tab=array();  
  
    $tab["cle1"] = "tab1";  
    $tab["cle2"] = "tab2";  
    $tab["cle3"] = "tab3";
```

## 02 - Traiter les données en PHP

### Traitement des tableaux (simple, Associatif)



#### Constantes pré-définies

Liste : <https://www.php.net/manual/fr/array.constants.php>

##### **CASE\_LOWER (int)**

est utilisée avec `array_change_key_case()` et sert à convertir tous les index d'un tableau en minuscules. C'est aussi le comportement par défaut de `array_change_key_case()`

##### **CASE\_UPPER (int)**

est utilisée avec `array_change_key_case()` et sert à convertir tous les index d'un tableau en majuscules

##### **SORT\_ASC (int)**

est utilisée avec `array_multisort()` pour trier en ordre ascendant

##### **SORT\_DESC (int)**

est utilisée avec `array_multisort()` pour trier en ordre descendant

##### **ARRAY\_FILTER\_USE\_KEY (int)**

est utilisé avec `array_filter()` pour passer chaque clé comme premier argument à la fonction de rappel fournie

##### **ARRAY\_FILTER\_USE\_BOTH (int)**

est utilisé avec `array_filter()` pour passer la valeur et la clé à la fonction de rappel fournie

## 02 - Traiter les données en PHP

### Traitement des tableaux (simple, Associatif)



#### Fonctions sur les tableaux

Liste : <https://www.php.net/manual/fr/ref.array.php>

##### **array\_chunk**

Sépare un tableau en tableaux de taille inférieure

##### **array\_diff\_key**

Calcule la différence de deux tableaux en utilisant les clés pour comparaison

##### **array\_diff**

Calcule la différence entre des tableaux

##### **array\_fill\_keys**

Remplit un tableau avec des valeurs, en spécifiant les clés

##### **array\_key\_first**

Récupère la première clé d'un tableau

##### **array\_merge**

Fusionne plusieurs tableaux en un seul

##### **array\_search**

Recherche dans un tableau la première clé associée à la valeur

#### Opérateurs de tableaux

Exemple	Nom	Résultat
<code>\$a + \$b</code>	Union	Union de <code>\$a</code> et <code>\$b</code> .
<code>\$a == \$b</code>	Égalité	<b>true</b> si <code>\$a</code> et <code>\$b</code> contiennent les mêmes paires clés/valeurs.
<code>\$a === \$b</code>	Identique	<b>true</b> si <code>\$a</code> et <code>\$b</code> contiennent les mêmes paires clés/valeurs dans le même ordre et du même type.
<code>\$a != \$b</code>	Inégalité	<b>true</b> si <code>\$a</code> n'est pas égal à <code>\$b</code> .
<code>\$a &lt;&gt; \$b</code>	Inégalité	<b>true</b> si <code>\$a</code> n'est pas égal à <code>\$b</code> .
<code>\$a !== \$b</code>	Non-identique	<b>true</b> si <code>\$a</code> n'est pas identique à <code>\$b</code> .

Tab : Opérateurs de tableaux

Source : <https://www.php.net/manual/fr/language.operators.array.php>



## CHAPITRE 2

### Traiter les données en PHP

1. Traitement des tableaux (simple, Associatif)
2. Manipulation de fichier (**chargement, Suppression, téléchargement**)

#### Ouvrir un fichier

- fopen = Ouvre un fichier ou une URL

Expression :

```
fopen(  
    string $filename,  
    string $mode,  
    bool $use_include_path = false,  
    resource $context = ?  
): resource
```

- Filename : est de la forme "protocole://"
- mode : spécifie le type d'accès désiré au flux
- Liste des protocoles et des gestionnaires supportés : <https://www.php.net/manual/fr/wrappers.php>
- use\_include\_path : paramètre optionnel peut être défini à 1 ou à true pour chercher le fichier dans l'include\_path.



Attention

Une configuration peut être nécessaire voir <https://www.php.net/manual/fr/filesystem.configuration.php>

#### Paramètre mode de fopen()

Liste des modes possibles pour la fonction fopen() en utilisant le paramètre mode

Mode	Description
'r'	Ouvre en lecture seule, et place le pointeur de fichier au début du fichier.
'r+'	Ouvre en lecture et écriture, et place le pointeur de fichier au début du fichier.
'w'	Ouvre en écriture seule ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. <b>Si le fichier n'existe pas, on tente de le créer.</b>
'w+'	Ouvre en lecture et écriture ; le comportement est le même que pour 'w'.
'a'	Ouvre en écriture seule ; place le pointeur de fichier à la fin du fichier. <b>Si le fichier n'existe pas, on tente de le créer.</b>
'a+'	Ouvre en lecture et écriture ; place le pointeur de fichier à la fin du fichier. <b>Si le fichier n'existe pas, on tente de le créer.</b>
'x'	Crée et ouvre le fichier en écriture seulement ; place le pointeur de fichier au début du fichier. Si le fichier existe déjà, fopen() va échouer, en retournant false et en générant une erreur de niveau E_WARNING. <b>Si le fichier n'existe pas, fopen() tente de le créer.</b>
'x+'	Crée et ouvre le fichier pour lecture et écriture; le comportement est le même que pour 'x'.
'c'	Ouvre le fichier pour écriture seulement. <b>Si le fichier n'existe pas, il sera créé et s'il existe, il n'est pas tronqué et l'appel à la fonction n'échoue pas.</b> Le pointeur du fichier est positionné au début.
'c+'	Ouvre le fichier pour lecture et écriture, le comportement est le même que pour le mode 'c'.
'e'	Défini l'indicateur close-on-exec sur le descripteur de fichier ouvert. Disponible uniquement en PHP compilé sur les systèmes conforme POSIX.1-2008.

Tab : Liste des modes pour la fonction fopen() en utilisant le paramètre mode. Source : <https://www.php.net/manual/fr/function.fopen.php>

## 02 - Traiter les données en PHP

### Manipulation de fichier



#### Lire un fichier

##### **fread()**

Lecture du fichier en mode binaire

##### **file()**

Lit le fichier et renvoie le résultat dans un tableau

##### **file\_get\_contents**

Lit tout un fichier dans une chaîne

##### **fstat**

Lit les informations sur un fichier à partir d'un pointeur de fichier

##### **fgets()**

Récupère la ligne courante à partir de l'emplacement du pointeur sur fichier

##### **fgetc**

Lit un caractère dans un fichier

##### **readfile**

Affiche un fichier

#### Autre opérations sur un fichier

Liste : <https://www.php.net/manual/fr/ref.filesystem.php>

##### **fclose**

Ferme un fichier

##### **fwrite**

Écrit un fichier en mode binaire

##### **fputs**

Alias de fwrite

##### **file\_put\_contents**

Écrit des données dans un fichier

## CHAPITRE 3

### Utiliser l'orientée objet en PHP



**Ce que vous allez apprendre dans ce chapitre :**

- Intérêt de programmer en Orienté Objet en PHP
- Application des concepts de base de la programmation orientée objet en PHP
- Utilisation des méthodes magiques en PHP



08 heures



## CHAPITRE 3

### Utiliser l'orientée objet en PHP

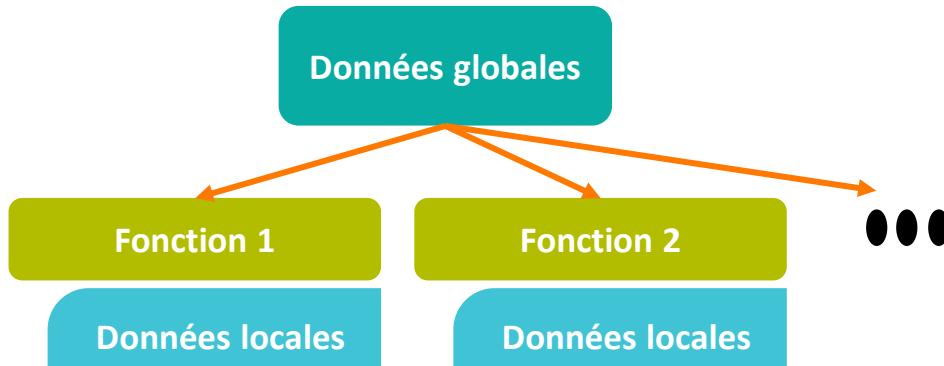
1. **Intérêt de programmer en Orienté Objet en PHP**
2. Application des concepts de base de la programmation orientée objet en PHP
3. Utilisation des méthodes magiques en PHP

## 03 - Utiliser l'orientée objet en PHP

### Intérêt de programmer en Orienté Objet en PHP

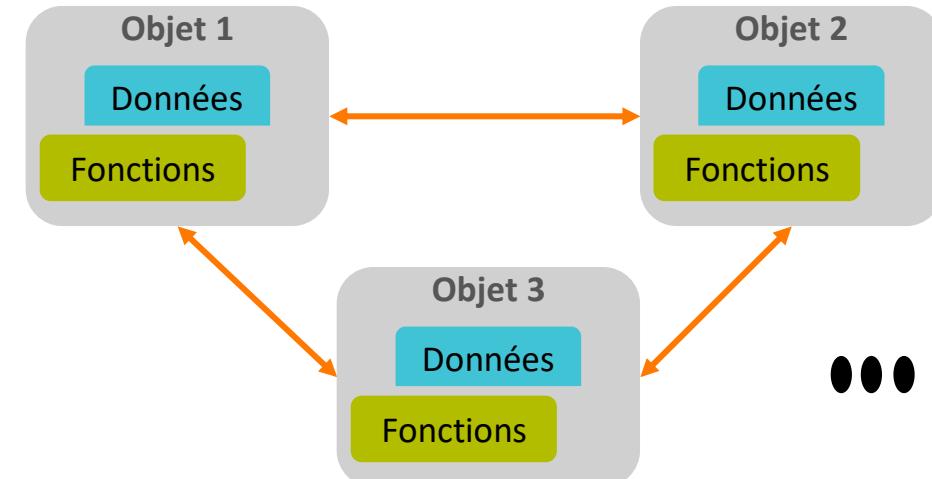
#### Programmation procédurale

- Le programme est divisé en procédures ou fonctions.
- Chaque fonction contient des données différentes.
- Une procédure permet d'effectuer des opérations sur les données généralement contenues dans des variables.
- Les opérations sont exécutées selon leur ordre d'écriture dans le script.



#### Programmation orientée objet

- Le programme est divisé en objets.
- Les objets agissent comme une seule unité.
- Un objet est une entité qui va pouvoir contenir un ensemble de fonctions et de variables.
- Regrouper des tâches précises au sein d'objets pour obtenir une nouvelle organisation du code



## 03 - Utiliser l'orientée objet en PHP

### Intérêt de programmer en Orienté Objet en PHP



#### Avantages d'une programmation orientée objet (POO)

Meilleur organisation du projet.

Facilite la maintenance du code.

Souplesse à évoluer le logiciel.

Factorisation des comportements, puisque les fonctions deviennent interdépendantes.

Masquage des données ou l'encapsulation consiste à masquer les détails d'implémentation d'un objet, en définissant une interface.

Spécificateur d'accès en utilisant un assesseur qui est une méthode d'accès pour connaître ou modifier la valeur d'un attribut d'un objet.

Transmission des propriétés grâce à l'héritage (ainsi que le polymorphisme), ainsi éviter la duplication de code.

l'agrégation qui permet de définir des objets composés d'autres objets.

## 03 - Utiliser l'orientée objet en PHP

### Intérêt de programmer en Orienté Objet en PHP



#### Les objets et les classes

Une classe :

- Est un ensemble de variables et de fonctions (attributs et méthodes).
- Contient le nom des propriétés qu'on pourra manipuler
- Les propriétés, ce sont des variables internes où on stocke des valeurs.
- Les méthodes sont des fonctions internes à la classe.
- Un objet est une instance de la classe pour pouvoir l'utiliser.

Exemple :

- **Classe** : Voiture
- **Propriétés** : Type, Energie, Puissance réelle, Boite de vitesse, Portes/Places , Vitesse maximale, Consommation moyenne, Emission de CO2, Couleur ...
- **Méthodes** : Accélérer, Freiner, Klaxonner, Feux, Tourner à droite, Reculer ...
- **Objets** : voiture rouge type hybride diesel 4 portes ... , voiture bleu essence 8cv type berline ..., ...



## CHAPITRE 3

### Utiliser l'orientée objet en PHP

1. Intérêt de programmer en Orienté Objet en PHP
2. **Application des concepts de base de la programmation orientée objet en PHP**
3. Utilisation des méthodes magiques en PHP

## 03 - Utiliser l'orientée objet en PHP

### Application des concepts de base de la programmation orientée objet en PHP

#### Syntaxe de base

- Une définition de classe basique commence par le mot-clé class, suivi du nom de la classe.

##### Le nom de la classe :

- Peut être ^[a-zA-Z\_\x80-\xff][a-zA-Z0-9\_\x80-\xff]\*\$
  - Ne doit pas être un mot réservé en PHP
- Liste des mots réservé en PHP : <https://www.php.net/manual/fr/reserved.php>

##### La pseudo-variable \$this est :

- Disponible lorsqu'une méthode est appelée depuis un contexte objet.
  - La valeur de l'objet appelant.
- Le mot-clé new est utilisé pour créer une instance d'une classe.
- Le mot-clé extends permet d'étendre les constantes, méthodes et les propriétés à une autre class.

```
<?php  
class NomClass  
{  
    // Déclaration d'une propriété  
  
    // Déclaration des méthodes  
}  
?>
```

```
<?php  
class Voiture  
{  
    // Déclaration d'une propriété  
  
    // Déclaration des méthodes  
}  
?>
```

## 03 - Utiliser l'orientée objet en PHP

### Application des concepts de base de la programmation orientée objet en PHP



#### Déclaration des attributs

Types pour déclarer un attribut :

- **public** : l'attribut sera accessible de l'intérieur de la classe dont il est membre comme de l'extérieur.
  - **private** : dans ce cas, l'attribut est accessible seulement de l'intérieur de la classe dont il est membre.
  - **protected** : dans ce cas, l'attribut est accessible seulement de l'intérieur de la classe dont il est membre ainsi que des classes qui héritent de cette classe.
- L'encapsulation est un principe fondamental de la POO. Il vise à masquer les attributs aux utilisateurs du code. C'est la visibilité private ou protected qui est recommandée.

```
class Voiture
{
    private $type = "Berline";
    private $Vitesse = 0;
    // Déclaration des méthodes
}
```

#### Déclaration des méthodes

Types pour déclarer une méthode :

- **public** : la méthode sera accessible de l'intérieur de la classe dont il est membre comme de l'extérieur.
- **private** : dans ce cas, la méthode est accessible seulement de l'intérieur de la classe. Généralement sollicitée par une autre méthode public.

```
class Voiture
{
    // Déclaration des méthodes

    public function accelerer(){
        // Corps de la méthode accelerer()
    }

    public function freiner(){
        // Corps de la méthode freiner()
    }
}
```

## 03 - Utiliser l'orientée objet en PHP

### Application des concepts de base de la programmation orientée objet en PHP



#### Les fonctionnalités d'un objet

- Le mot-clé **extends** permet de définir une classe dérivée.
- Le mot-clé **function** permet de déclarer une méthode.
- Le mot-clé **parent** permet d'accéder au parent de l'objet courant.
- **\$this->** est obligatoire pour accéder aux membres de l'objet courant.
- Mot-clé **final** empêche les classes enfants de redéfinir une méthode ou constante en préfixant la définition avec final. Si la classe elle-même est définie comme finale, elle ne pourra pas être étendue.
- Une propriété peut être déclarée avec le modificateur **readonly**, qui empêche la modification de la propriété après l'initialisation.

# 03 - Utiliser l'orientée objet en PHP

## Application des concepts de base de la programmation orientée objet en PHP



### Fonctions Classes / Objets

#### `__autoload`

Tente de charger une classe indéfinie

#### `get_called_class`

Le nom de la classe en "Late Static Binding"

#### `get_declared_classes`

Liste toutes les classes définies dans PHP

#### `get_object_vars`

Retourne les propriétés d'un objet

#### `is_subclass_of`

Détermine si un objet est une sous-classe d'une classe donnée ou l'implémente

#### `class_alias`

Crée un alias de classe

#### `get_class_methods`

Retourne les noms des méthodes d'une classe

#### `get_declared_interfaces`

Retourne un tableau avec toutes les interfaces déclarées

#### `get_parent_class`

Retourne le nom de la classe parente d'un objet

#### `method_exists`

Vérifie si la méthode existe pour une classe

#### `class_exists`

Vérifie si une classe a été définie

#### `get_class_vars`

Retourne les valeurs par défaut des propriétés d'une classe

#### `get_declared_traits`

Retourne un tableau contenant tous les traits déclarés

#### `interface_exists`

Vérifie si une interface a été définie

#### `property_exists`

Vérifie si un objet ou une classe possède une propriété

#### `enum_exists`

Checks if the enum has been defined

#### `get_class`

Retourne le nom de la classe d'un objet

#### `get_mangled_object_vars`

Returns an array of mangled object properties

#### `is_a`

Vérifie si l'objet est une instance d'une classe donnée ou a cette classe parmi ses parents

#### `trait_exists`

Vérifie si un trait existe



## CHAPITRE 3

### Utiliser l'orientée objet en PHP

1. Intérêt de programmer en Orienté Objet en PHP
2. Application des concepts de base de la programmation orientée objet en PHP
3. **Utilisation des méthodes magiques en PHP**

#### Méthodes magiques

Les méthodes spéciales peuvent écraser l'action par défaut de PHP.

Les méthodes commençant par double sous-tirets \_\_ sont réservées par PHP. (Cette notation est réservée aux méthodes magiques)

17 méthodes sont considérées magiques

Toutes les méthodes magiques, à l'exception de \_\_construct(), \_\_destruct(), et \_\_clone(), doivent être déclarées en tant que public, sinon une E\_WARNING est émise.

#### \_\_construct()

Syntaxe : \_\_construct(mixed ...\$values = ""): void

Les classes qui possèdent une méthode constructeur appellent cette méthode à chaque création d'une nouvelle instance de l'objet.

Si vous voulez utiliser un constructeur parent, il sera nécessaire de faire appel à parent::\_\_construct() depuis le constructeur enfant.

Un constructeur peut définir un nombre d'arguments, qui peuvent être requis, avoir un type, et peuvent avoir une valeur par défaut.

```
class Voiture{
    public function __construct(){
        $this->couleur="Blanche";
        $this->vitesse=0;
    }
}
$maVoiture = new Voiture(); // le constructeur est appelé et les attributs initialisés
```

#### \_\_destruct()

Syntaxe : \_\_destruct(): void

La méthode destructeur est appelée dès qu'il n'y a plus de référence sur un objet donné.

Si vous voulez utiliser un destructeur parent, il sera nécessaire de faire appel à parent::\_\_destruct() depuis le destructeur enfant.

Le destructeur sera appelé même si l'exécution du script est stoppée en utilisant la fonction exit().

#### Surcharge de propriétés

- **\_\_set()** :
  - Est sollicitée lors de l'écriture de données vers des propriétés inaccessibles (protégées ou privées) ou non existante.
  - Syntaxe : public \_\_set(string \$name, mixed \$value): void
- **\_\_get()** :
  - Est appelée pour lire des données depuis des propriétés inaccessibles (protégées ou privées) ou non existante.
  - Syntaxe : public \_\_get(string \$name): mixed
- **\_\_isset()** :
  - Est sollicitée lorsque isset() ou empty() sont appelées sur des propriétés inaccessibles (protégées ou privées) ou non existante.
  - Syntaxe : public \_\_isset(string \$name): bool
- **\_\_unset()** :
  - Est invoquée lorsque unset() est appelée sur des propriétés inaccessibles (protégées ou privées) ou non existante.
  - Syntaxe : public \_\_unset(string \$name): void

## 03 - Utiliser l'orientée objet en PHP

### Utilisation des méthodes magiques en PHP

#### \_\_sleep() et \_\_serialize()

- \_\_sleep() :
  - Est de valider des données en attente ou d'effectuer des opérations de nettoyage.
  - Syntaxe : public \_\_sleep(): array
- \_\_serialize():
  - Vérifie si la classe a une méthode avec le nom magique \_\_sleep().
  - Retourne les noms de propriétés privées des classes parentes.  
Construit et retourne un tableau associatif de paire clé/valeur qui représente la forme linéarisée de l'objet
  - Syntaxe : public \_\_serialize(): array

**Si \_\_serialize() et \_\_sleep() sont tout les deux définie dans le même objet, alors seulement \_\_serialize() sera appelé.**

#### \_\_wakeup() et \_\_unserialize():

- \_\_wakeup() :
  - Est de rétablir toute connexion de base de données qui aurait été perdue durant la linéarisation et d'effectuer des tâches de réinitialisation.
  - Syntaxe : public \_\_wakeup(): void
- \_\_unserialize():
  - Vérifie si la classe a une méthode avec le nom magique \_\_wakeup().
  - Peut reconstruire toute ressource que l'objet pourrait posséder.
  - Syntaxe : public \_\_unserialize(array \$data): void

**Si \_\_unserialize() et \_\_wakeup() sont tout les deux définie dans le même objet, alors seulement \_\_unserialize() sera appelée.**

## 03 - Utiliser l'orientée objet en PHP

### Utilisation des méthodes magiques en PHP



#### Surcharge de méthodes

- **`__call()`** :
  - Est appelée lorsque l'on invoque des méthodes inaccessibles dans un contexte objet.
  - Syntaxe: `public __call(string $name, array $arguments): mixed`
- **`__callStatic()`** :
  - Est lancée lorsque l'on invoque des méthodes inaccessibles dans un contexte statique.
  - Syntaxe : `public static __callStatic(string $name, array $arguments): mixed`

## 03 - Utiliser l'orientée objet en PHP

### Utilisation des méthodes magiques en PHP



#### Autres méthodes magiques

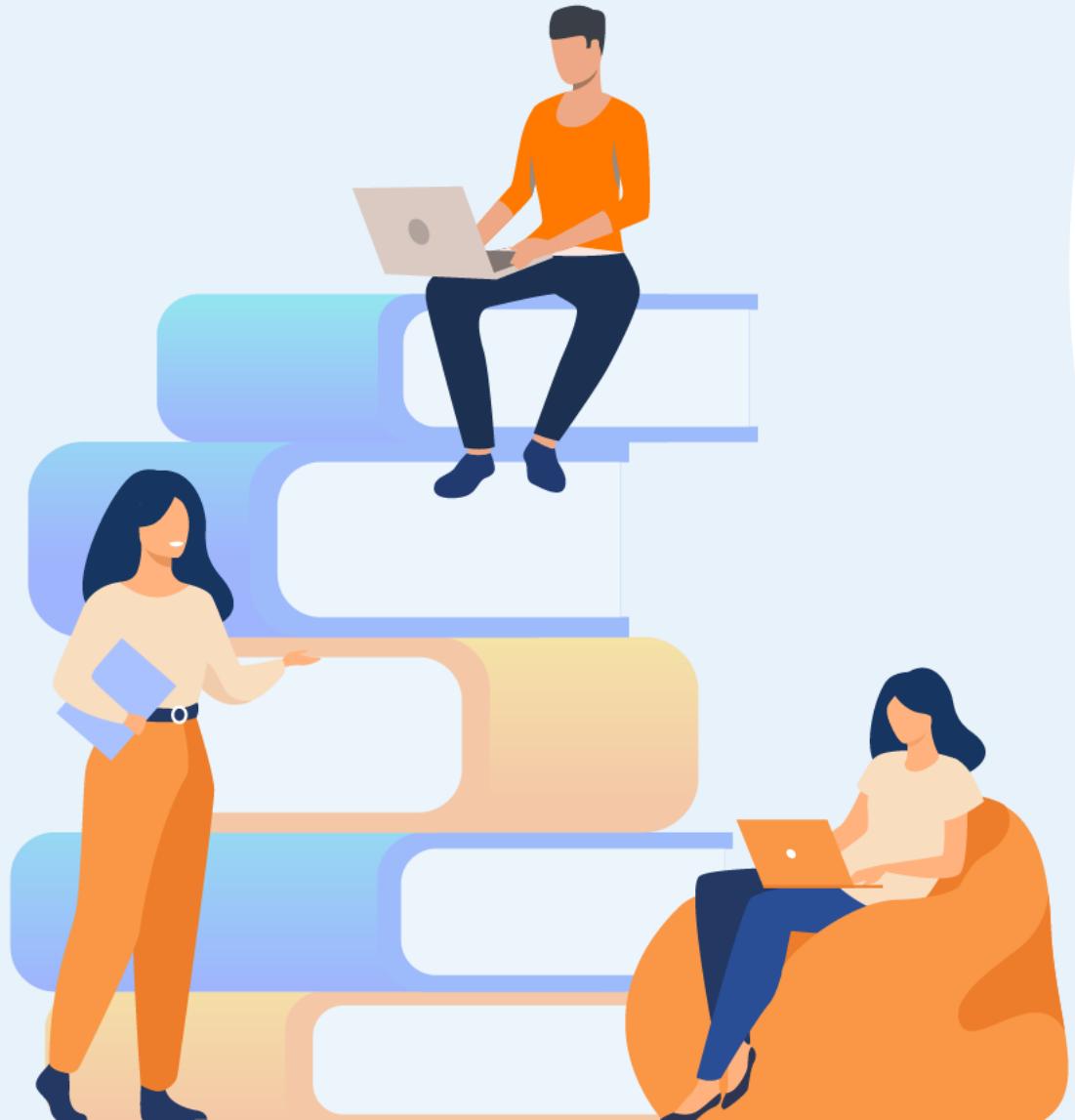
- **`__toString()`:**
  - Détermine comment l'objet doit réagir lorsqu'il est traité comme une chaîne de caractères.
  - Syntaxe : `public __toString(): string`
- **`__invoke()` :**
  - Est appelée lorsqu'un script tente d'appeler un objet comme une fonction.
  - Syntaxe : `__invoke( ...$values): mixed`
- **`__set_state()` :**
  - Est appelée pour les classes exportées par la fonction `var_export()`.
  - Syntaxe : `static __set_state(array $properties): object`
- **`__clone()` :**
  - Un nouvel objet sera appelée, pour permettre à chaque propriété qui doit l'être d'être modifiée.
  - Syntaxe : `__clone(): void`

## PARTIE 2 : Programmer en PHP

### Références et ressources

- Le site officiel : <https://www.php.net/manual/fr/>
- <https://openclassrooms.com/fr/courses>
- <https://www.phpfacile.com>
- <https://php.developpez.com/>





## PARTIE 3

### Manipuler les données

**Dans ce module, vous allez :**

- Interroger une base de données à travers un formulaire
- Manipuler des données CRUD
- Utiliser les sessions et les cookies
- Acquérir les notions de sécurisation des données



 **36 heures**



## Chapitre 1

### Écrire des scripts d'accès aux données

**Ce que vous allez apprendre dans ce chapitre :**

- Connexion à une base de données MySQL avec PDO
- Interrogation d'une base de données à travers un formulaire
- Récupération des résultats
- Manipulation des données CRUD



12 heures



## CHAPITRE 1

### Écrire des scripts d'accès aux données

1. Connexion à une base de données MySQL avec PDO
2. Interrogation d'une base de données à travers un formulaire
3. Récupération des résultats
4. Manipulation des données CRUD

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Introduction

- Pour enregistrer les données en PHP vous pouvez utiliser :
  - Les variables
  - Les fichiers
  - Une base de données
- Une base de données (BDD) permet de stocker, organiser et analyser les données.
- Un système de gestion de base de données (SGBD) permet d'accéder aux bases de données.
- Une couche d'abstraction de base de données est une interface de programmation d'applications qui unifie la communication entre une application informatique et des bases de données.
- **L'API** : est une solution informatique qui permet à des applications de communiquer avec d'autres applications et de s'échanger mutuellement des services ou des données sans connaître les détails de leur mise en œuvre.
- **Une DLL** : Une Dynamic Link Library est une bibliothèque logicielle dont les fonctions sont chargées en mémoire par un programme, au besoin, lors de son exécution, par opposition aux bibliothèques logicielles statiques ou partagées dont les fonctions sont chargées en mémoire avant le début de l'exécution du programme. (source Wikipédia)

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Les bases de données

**CUBRID** : Est une solution SGBDR (Système de Gestion de Base de Données Relationnelles) gratuite et open source.

**dBase** : Cette extension n'est pas recommandée par PHP et n'est plus intégrée depuis PHP 5.3.0

**Firebird/InterBase** : Cette extension n'est pas recommandée par PHP et n'est plus intégrée depuis PHP 7.4.0

**IBM DB2 - Fonctions IBM DB2, Cloudscape et Apache Derby**

**MongoDB - MongoDB driver** : Ces fonctions vous permettent un accès aux IBM DB2 Universal Database, IBM Cloudscape et Apache Derby qui utilisent DB2 Call Level Interface (CLI).

**MySQL - Plugins et drivers MySQL** : recommandé par PHP qui offre plusieurs drivers et plugins pour accéder et gérer MySQL.

**OCI8 - Oracle OCI8** : Ces fonctions vous permettent d'accéder aux bases de données Oracle. Elles supportent les commandes SQL et PL/SQL.

**PostgreSQL** : Recommandé par PHP. La base de données PostgreSQL est un produit Open Source, disponible sans frais.

**SQLite3** : Support des bases de données SQLite version 3.

**SQLSRV - Driver Microsoft SQL Server pour PHP** : Vous permet d'accéder à un serveur de base de données Microsoft SQL et SQL Azure.

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### MySQL

Avant de créer une connexion à un serveur de base de données MySQL, vous devez avoir :

- Un serveur de base de données MySQL installé sur votre système local ou sur un serveur distant.
- Une base de données sur le serveur MySQL.
- Un compte MySQL avec un nom d'utilisateur et un mot de passe permettant d'accéder à la base de données.

• Les plateformes de développement Web PHP comme EasyPHP, LAMP, WAMP, MAMP, LEMP et XAMPP fournissent **MySQL**.

Fonctions MySQL :

- `mysql_connect` — Ouvre une connexion à un serveur MySQL
- `mysql_create_db` — Crée une base de données MySQL
- `mysql_ping` — Vérifie la connexion au serveur MySQL, et s'y reconnecte au besoin
- `mysql_query` — Envoie une requête à un serveur MySQL
- ...

• Liste complète des fonctions MySQL en PHP : <https://www.php.net/manual/fr/ref.mysql.php>

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



WEBFORCE  
BE THE CHANGE

### Se connecter à MySQL en PHP

Pour pouvoir manipuler nos bases de données MySQL en PHP (sans passer par phpMyAdmin par exemple). PHP met à notre disposition deux API (Application Programming Interface) :

- L'extension MySQLi
  - L'extension PDO (PHP Data Objects)
- La performance globale des deux extensions MySQLi et PDO peut être considérée comme identique. Nous allons choisir dans ce qui suit du cours l'interface PDO.

Interface d'abstraction pour le langage PHP :

- DBA : Couche d'abstraction de base de données (style dbm). Ces fonctions permettent l'accès aux bases de données du style Berkeley DB.
- ODBC - ODBC (Unifié) : vous donne accès à diverses bases de données qui ont emprunté la sémantique des API ODBC pour implémenter leur propres API. Les bases de données suivantes sont supportées par l'ODBC unifié : Adabas D, IBM DB2, iODBC, Solid, et Sybase SQL Anywhere.
- PDO : PHP Data Objects définit une excellente interface pour accéder à une base de données depuis PHP.

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Installation PDO

- Choisissez les autres fichiers DLL spécifiques à votre base de données et utilisez soit la fonction `dl()` pour les charger au moment de l'exécution ou activez-les dans le fichier `php.ini` en-dessous de la ligne `php_pdo.dll`.

#### Fonction `dl` :

- Définition** : Charge une extension PHP à la volée
  - Syntaxe** : `dl(string $extension_filename) : bool`
- 
- La librairie DLL doit exister dans le dossier système `extension_dir`.
  - Directive `extension_dir` : Spécifie le répertoire dans lequel PHP doit chercher des extensions externes à charger. Il est recommandé de spécifier un chemin absolu.
  - Il faut redémarrer PHP après chaque modification de votre `php.ini`, pour que les nouvelles directives de configuration prennent effet.
  - Lors de la compilation utilisez `--with-pdo-mysql[=DIR]` pour installer l'extension PDO MySQL, où [=DIR] est le chemin de la bibliothèque de base de MySQL

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Fonctions PDO

- PDO::beginTransaction → Démarre une transaction
- PDO::commit → Valide une transaction
- PDO::\_\_construct → Crée une instance PDO qui représente une connexion à la base
- PDO::errorCode → Retourne le SQLSTATE associé avec la dernière opération sur la base de données
- PDO::errorInfo → Retourne les informations associées à l'erreur lors de la dernière opération sur la base de données
- PDO::exec → Exécute une requête SQL et retourne le nombre de lignes affectées
- PDO::getAttribute → Récupère un attribut d'une connexion à une base de données
- PDO::getAvailableDrivers → Retourne la liste des pilotes PDO disponibles
- PDO::inTransaction → Vérifie si nous sommes dans une transaction
- PDO::lastInsertId → Retourne l'identifiant de la dernière ligne insérée ou la valeur d'une séquence
- PDO::prepare → Prépare une requête à l'exécution et retourne un objet
- PDO::query → Prépare et Exécute une requête SQL sans marque substitutive
- PDO::quote → Protège une chaîne pour l'utiliser dans une requête SQL PDO
- PDO::rollBack → Annule une transaction
- PDO::setAttribute → Configure un attribut PDO

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO

### La classe PDOStatement

- **PDOStatement** : Représente une requête préparée et, une fois exécutée, le jeu de résultats associé.

Méthodes :

- **PDOStatement::bindParam** : Lie une colonne à une variable PHP
- **PDOStatement::bindValue** : Lie un paramètre à un nom de variable spécifique
- **PDOStatement::columnCount** : Retourne le nombre de colonnes dans le jeu de résultats
- **PDOStatement::debugDumpParams** : Détaille une commande préparée SQL
- **PDOStatement::errorCode** : Récupère les informations sur l'erreur associée lors de la dernière opération sur la requête
- **PDOStatement::errorInfo** : Récupère les informations sur l'erreur associée lors de la dernière opération sur la requête
- **PDOStatement::execute** : Exécute une requête préparée
- **PDOStatement::fetchAll** : Retourne un tableau contenant toutes les lignes du jeu d'enregistrements
- **PDOStatement::getAttribute** : Récupère un attribut de requête
- ...
- Liste complète : <https://www.php.net/manual/fr/class.pdostatement.php>

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### La classe PDOException

- **PDOException** : Représente une erreur émise par PDO.

Propriétés :

- **errorInfo** : Retourne les informations associées à l'erreur lors de la dernière opération sur la base de données. Correspond à PDO::errorInfo() ou PDOStatement::errorInfo().
- **errorcode** : Retourne le SQLSTATE associé avec la dernière opération sur la base de données. (SQLSTATE : un identifiant alphanumérique de cinq caractères défini dans le standard ANSI SQL.)
- **PDO::errorInfo**
  - **Syntaxe** : public PDO::errorInfo(): array
  - **Retourne un tableau contenant** :
    - [0] : Code d'erreur SQLSTATE
    - [1] : Code d'erreur spécifique au driver.
    - [2] : Message d'erreur spécifique au driver.
- **PDO::errorCode**
  - **Syntaxe** : public PDO::errorCode(): ?string
  - Retourne null si aucune opération n'a été exécutée sur la base de données.

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Configuration

- Le pilote PDO\_MYSQL implémente l'interface de PDO pour autoriser l'accès de PHP aux bases de données MySQL. Pour déclarer le pilote du SGBD MySQL, il faut ajouter la ligne suivante dans le fichier php.ini : extension=php\_pdo\_mysql.dll
- L'extension PDO peut utiliser soit la bibliothèque mysqlnd, soit la bibliothèque libmysqlclient.
- La bibliothèque interne mysqlnd est la bibliothèque par défaut et qui est recommandé par le PHP.

Nom	Défaut	Modifiable	Historique
<a href="#"><u>pdo.dsn.*</u></a>		<i>php.ini</i> seulement	

Tab.1 : Options de configuration PDO. Source : <https://www.php.net/manual/fr/pdo.configuration.php>

Nom	Défaut	Modifiable
<a href="#"><u>pdo_mysql.default_socket</u></a>	"/tmp/mysql.sock"	PHP_INI_SYSTEM
<a href="#"><u>pdo_mysql.debug</u></a>	NULL	PHP_INI_SYSTEM

Tab.2 : Options de configuration du driver PDO\_MYSQL. Source : <https://www.php.net/manual/fr/ref pdo-mysql.php>

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### SQL

- SQL (Structured Query Language) est un langage de programmation standardisé utilisé pour gérer les bases de données au sein des SGBD et plus particulièrement les SGBD relationnelles (ex: MySQL), permettant d'effectuer des opérations sur les données qu'elles contiennent.

**Les instructions SQL couvrent 4 domaines :**

- Langage de définition de données (LDD) : permet de créer ou supprimer des objets dans la base de données.
  - Langage de manipulation de données (LMD) : permet de manipuler les données contenues dans les tables.
  - Langage de contrôle de données (LCD) : permet de gérer les utilisateurs d'une base de données ainsi que leurs droits sur les objets.
  - Langage de contrôle des transactions (LCT) : permet de valider ou annuler des modifications de données dans la base de données.
- 
- Liste des commandes SQL : <https://sql.sh/>
  - Une erreur à la ligne de la fonction `fetchAll()` est affichée lorsqu'une erreur de requête SQL est détectée dans un code PHP.

**Exemple Erreur `fetchAll()` :**

Fatal error: Call to a member function `fetchAll()` on a non-object in `index.php` on line 13

# 01 - Écrire des scripts d'accès aux données

## Connexion à une base de données MySQL avec PDO



### Exemple: Script de connexion à MySQL avec PDO

- Le serveur de base de données MySQL local.
- Une base de données nommée « test ».
- Le compte root avec un mot de passe vide.

```
<?php

$host = 'localhost';
$dbname = 'test';
$username = 'root';
$password = "";

try {

    $connexion = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    echo "Vous êtes maintenant connecté à $dbname sur $host.";

} catch (PDOException $e) {
    die("Impossible de se connecter à la base de donnée $dbname :" . $e->getMessage());
}

?>
```



## CHAPITRE 1

### Écrire des scripts d'accès aux données

1. Connexion à une base de données MySQL avec PDO
2. **Interrogation d'une base de données à travers un formulaire**
3. Récupération des résultats
4. Manipulation des données CRUD

# 01 - Écrire des scripts d'accès aux données

## Interrogation d'une base de données à travers un formulaire

### Paramétrer la connexion

- Pour établir la connexion, il faut créer un DSN (Data Source Name) en renseignant des informations comme par exemple :
  - Le type de base de données qui est MySQL.
  - Le nom de la base de données.
  - Le port de connexion.
  - L'encodage.
  - ...

```
$dsn =  
'mysql:host=ADRESSE_DU_SERVEUR;dbname=  
e=VOTRE_BASE_DE_DONNEES;port=VOTRE_  
PORT;charset=VOTRE_ENCODAGE';
```

### Connexion au serveur MySQL

- Pour connecter PHP à MySQL avec PDO, il est obligatoire de fournir les quatre renseignements suivants :
  - Le nom d'hôte
  - La base de données
  - L'identifiant (login)
  - Le mot de passe

Syntaxe :

```
$connexion_pdo = new PDO($dsn,  
'VOTRE_NOM_UTILISATEUR_SERVEUR',  
'VOTRE_MOT_DE_PASSE');
```

Exemple :

```
$<?php  
$mysqlConnection = new PDO  
'mysql:host=localhost;dbname=testDB;charset=utf8', 'root', 'root'  
);  
?>
```

# 01 - Écrire des scripts d'accès aux données

## Interrogation d'une base de données à travers un formulaire



### Connexion avec gestion des erreurs

- Pour afficher des détails sur l'erreur, il faut activer les erreurs lors de la connexion à la base de données via PDO.
- Activer les erreurs lors de la connexion à la base de données permettra d'avoir un message d'erreur détaillé.

Exemple :

```
<?php
$db = new PDO(
    'mysql:host=localhost;dbname=test;charset=utf8',
    'root',
    '',
    [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION],
);
?>
```

# 01 - Écrire des scripts d'accès aux données

## Interrogation d'une base de données à travers un formulaire



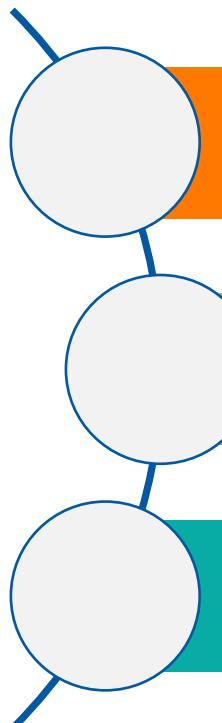
### Utiliser MySQL

- Création d'une base de données : **CREATE DATABASE test**
- Création d'un login et mot de passe : **CREATE USER 'user1'@'localhost' IDENTIFIED BY ".monPass1!"**  
(On crée un utilisateur « user1 » avec le mot de passe « .monPass1! »)
- Création d'une table :  
**CREATE TABLE personne (**  
id int(11) unsigned NOT NULL auto\_increment PRIMARY KEY,  
nom varchar(30) NOT NULL,  
email varchar(20) NOT NULL,  
timestamp int(11) unsigned NOT NULL  
**);**
- Enregistrer les données en PHP : **\$table = "personne";**

# 01 - Écrire des scripts d'accès aux données

## Interrogation d'une base de données à travers un formulaire

### Requêtes SQL



Ajouter une colonne : `ALTER TABLE [tablename] ADD COLUMN [colonne3] VARCHAR(100);`

Ajouter une colonne avec un ID unique par incrémentation automatique : `ALTER TABLE [tablename] ADD COLUMN [colonne4] int NOT NULL AUTO_INCREMENT PRIMARY KEY;`

Insérer un enregistrement dans une table SQL : `INSERT INTO [tablename] ([colonne1], [colonne2]) VALUES ('[valeur1]', '[valeur2]');`

# 01 - Écrire des scripts d'accès aux données

## Interrogation d'une base de données à travers un formulaire

### Formulaire HTML

- En HTML, il existe principalement deux méthodes pour interagir avec un utilisateur :
  - Les liens (balise `<a>`).
  - Les formulaires (balise `<form>`).

Exemple :

```
<html>
  <head>
    <title>Exemple formulaire en PHP/MySQL</title>
  </head>
  <body>
    <form method="post" action="controller.php">
      Name : <input type="text" name="nom"
placeholder="Entrez votre nom" /><br />
      Email : <input type="email" name="email"
placeholder="Entrer votre Email" /><br />
      <input type="submit" value="Soumettre" />
    </form>
  </body>
</html>);
```

### Page traitement PHP

- La page PHP permet de collecter des données à partir du formulaire.
- La valeur de l'attribut « `action` » dans la balise de formulaire HTML signifie que toutes les valeurs des champs d'entrée seront envoyées au fichier `.php` présent dans la valeur de l'attribut « `action` ».

Exemple (`controller.php`) :

```
<?php
// Vérifie qu'il provient d'un formulaire
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (!isset($name)){
    die("S'il vous plaît entrez votre nom");
  }
  if (!isset($email)){
    die("S'il vous plaît entrez votre adresse e-mail");
  }
  print "Salut " . $name . "!, votre adresse e-mail est " . $email;
}
?>
```

### Stockage des données dans MySql

Prérequis : une table « utilisateurs » créée dans la base de donnée MySQL.

```
<?php
    //identifiants mysql
    $host = "localhost";
    $username = "root";
    $password = "";
    $database = "utilisateurs";

    $nom = $_POST["nom"];
    $email = $_POST["email"];

    //Ouvrir une nouvelle connexion au serveur MySQL
    $connexionPDO = new PDO("mysql:host=$host;dbname=$database", $username, $password);

    //préparer la requête d'insertion SQL
    $statement = $connexionPDO->prepare("INSERT INTO users (nom, email) VALUES(?, ?)");
    //Associer les valeurs et exécuter la requête d'insertion
    $statement->bindParam('ss', $nom, $email);
?>
```



## CHAPITRE 1

### Écrire des scripts d'accès aux données

1. Connexion à une base de données MySQL avec PDO
2. Interrogation d'une base de données à travers un formulaire
- 3. Récupération des résultats**
4. Manipulation des données CRUD

# 01 - Écrire des scripts d'accès aux données

## Récupération des résultats



### Requêtes SQL

Envoyez une requête SQL depuis votre script PHP en utilisant : PDO::query()

Afficher toutes les données d'une table : SELECT \* FROM [tablename];

Sélectionner des enregistrements spécifiques : SELECT \* FROM [table] WHERE [colonne] = [valeur];

Sélectionnez les enregistrements qui contiennent la valeur [val] : SELECT \* FROM [table] WHERE [colonne] LIKE '%[val]%' ;

Sélectionnez les enregistrements commençant par 'val1' et se terminant par 'val2' : SELECT \* FROM [table] WHERE [colonne] LIKE '[val1\_val2]' ;

Sélectionnez des enregistrements sans doublons : SELECT **distinct** nom, email FROM personne WHERE nom = "Mohammed";

Faire un SELECT sur plusieurs tables : SELECT [table1].[colonne1], [table2].[colonne2] FROM [table1], [table2];

# 01 - Écrire des scripts d'accès aux données

## Récupération des résultats



WEBFORCE  
BE THE CHANGE

### Constantes PDO

#### PDO::FETCH\_ASSOC (int)

- Spécifie que la méthode de récupération doit retourner chaque ligne dans un tableau indexé par les noms des colonnes comme elles sont retournées dans le jeu de résultats correspondant. Si le jeu de résultats contient de multiples colonnes avec le même nom, PDO::FETCH\_ASSOC retourne une seule valeur par nom de colonne.

#### PDO::FETCH\_NAMED (int)

- Spécifie que la méthode de récupération doit retourner chaque ligne dans un tableau indexé par les noms des colonnes comme elles sont retournées dans le jeu de résultats correspondant. Si le jeu de résultats contient de multiples colonnes avec le même nom, PDO::FETCH\_NAMED retourne un tableau de valeurs par nom de colonne.

#### PDO::FETCH\_COLUMN (int)

- Spécifie que la méthode de récupération doit retourner uniquement une seule colonne demandée depuis la prochaine ligne du jeu de résultats.
- Liste des constantes prédéfinies : <https://www.php.net/manual/fr/pdo.constants.php>



## CHAPITRE 1

### Écrire des scripts d'accès aux données

1. Connexion à une base de données MYSQL avec PDO
2. Interrogation d'une base de données à travers un formulaire
3. Récupération des résultats
4. **Manipulation des données CRUD**

# 01 - Écrire des scripts d'accès aux données

## Manipulation des données CRUD



### CRUD

- CRUD est un acronyme des noms des quatre opérations de base de la gestion de la persistance des données et applications :
  - Create (créer)
  - Read ou Retrieve (lire)
  - Update (mettre à jour)
  - Delete ou Destroy (supprimer)
- Les opérations CRUD sont exécutées différemment en fonction de l'environnement, comme indiqué ci-dessous dans le tableau :

CRUD-Operation	SQL	RESTful HTTP	XQuery
Create	INSERT	POST, PUT	insert
Read	SELECT	GET, HEAD	copy/modify/return
Update	UPDATE	PUT, PATCH	replace, rename
Delete	DELETE	DELETE	delete

Tab.3 : Opérations CRUD dans SQL, RESTful HTTP et XQuery

# 01 - Écrire des scripts d'accès aux données

## Manipulation des données CRUD



### Requêtes SQL

Mettre à jour des enregistrements : UPDATE [table] SET [colonne] = '[new\_val]' WHERE [colonne] = '[old\_val]';

Supprimer des enregistrements : DELETE FROM [table] WHERE [colonne] = [valeur];

Supprimer tous les enregistrements d'une table : DELETE FROM [table];

Supprimer les colonnes d'une table : ALTER TABLE [table] DROP COLUMN [colonne];

Supprimer une table : DROP TABLE [table];

Supprimer une base de donnée : DROP DATABASE [database];

Créer un nouvel utilisateur : CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';

Accorder un accès complet à l'utilisateur sur toutes les tables : GRANT ALL PRIVILEGES ON database.\* TO 'user'@'localhost';

# 01 - Écrire des scripts d'accès aux données

## Manipulation des données CRUD



### Insérer plusieurs entrées dans une table

Exemple :

```
$dbc->beginTransaction();

$sql1 = "INSERT INTO Clients(Nom,Prenom,Pays,Mail)
          VALUES('Ahmed','Ben','Maroc','test@gmail.com')";
$dbc->exec($sql1);

$sql2 = "INSERT INTO Clients(Nom,Prenom,Pays,Mail)
          VALUES('Julia','Joly','France','test@yahoo.com')";
$dbc->exec($sql2);

$dbc->commit();
echo 'Entrées ajoutées dans la table';
```

# 01 - Écrire des scripts d'accès aux données

## Manipulation des données CRUD



### Exemple organisation des fichiers PHP

#### **index.php :**

Page d'accueil de notre application  
CRUD.

#### **create.php :**

Créez de nouveaux enregistrements  
avec un formulaire HTML et envoyez  
des données au serveur avec une  
requête POST.

#### **read.php :**

Affichez les enregistrements de notre  
table de base de données et naviguez  
avec la pagination.

#### **update.php :**

Met à jour les enregistrements  
existants avec un formulaire HTML et  
envoie les données au serveur avec  
une requête POST.

#### **delete.php :**

Confirme et supprime les  
enregistrements par ID (requête GET  
pour obtenir l'ID).

#### **functions.php :**

Fonctions de base de modèles et  
fonction de connexion MySQL (nous  
n'avons donc pas à répéter le code  
dans chaque fichier).

#### **style.css :**

La feuille de style de notre application,  
cela changera l'apparence de notre  
application.



## Chapitre 2

### Sécuriser les données

**Ce que vous allez apprendre dans ce chapitre :**

- Utilisation des sessions et des cookies
- Sécurisation des données



**12 heures**



## CHAPITRE 2

### Sécuriser les données

1. Utilisation des sessions et des cookies
2. Sécurisation des données

#### Introduction

- Pour transmettre des variables de page en page, on peut utiliser :
  - Les divers champs des formulaires.
  - Passer les variables directement à travers les liens.
  - Utiliser les cookies.
  - Utiliser les sessions.
- Quand une session est créée sur le serveur, ce dernier envoie son identifiant (unique) au client sous forme d'un cookie.

#### Cookies

- Les cookies sont un mécanisme d'enregistrement d'informations sur le client, et de lecture de ces informations.
- Les cookies font partie des en-têtes HTTP, ce qui impose que les fonctions doivent être appelées avant tout affichage de texte.
- variables\_order string : Définit l'ordre d'analyse des variables EGPCS (Environment, Get, Post, Cookie, et Server). Par exemple, si variables\_order est défini à "SP" alors PHP créera superglobals \$\_SERVER et \$\_POST, mais ne créera pas \$\_ENV, \$\_GET, et \$\_COOKIE. Le fait de définir cet ordre à "" signifie qu'aucune superglobale ne sera définie.
- Tous les cookies envoyés au serveur par le client seront automatiquement inclus dans un tableau auto-global \$\_COOKIE si variables\_order contient "C".
- Si vous souhaitez affecter plusieurs valeurs à un seul cookie, ajoutez [] au nom du cookie.

#### La fonction setcookie()

Il faut appeler cette fonction avant toute balise <html> ou <head> et aussi des caractères d'espacement blanc.

Une fois que les cookies ont été placés, ils seront accessibles lors du prochain chargement de page dans le tableau \$\_COOKIE. Les valeurs des cookies peuvent aussi exister dans la variable \$\_REQUEST.

**Syntaxe :** setcookie( string \$name, string \$value = "", int \$expires\_or\_options = 0, string \$path = "", string \$domain = "", bool \$secure = false, bool \$httponly = false, array \$options = [] ) : bool

**name :** le nom du cookie

**value :** Cette valeur est stockée sur l'ordinateur du client.

**expires\_or\_options :** Le temps après lequel le cookie expire, c'est un timestamp Unix. Exemple : time() + 60 \* 60 fera expirer le cookie dans 1 heure. Si vous ne spécifiez pas ce paramètre ou s'il vaut 0, le cookie expirera à la fin de la session (lorsque le navigateur sera fermé).

#### La fonction setcookie()

**Path** : Le chemin sur le serveur sur lequel le cookie sera disponible.

**Domain** : Le (sous-)domaine pour lequel le cookie est disponible.

**Secure** : Indique si le cookie doit uniquement être transmis à travers une connexion sécurisée HTTPS depuis le client. Lorsque ce paramètre vaut true, le cookie ne sera envoyé que si la connexion est sécurisée.

**Httponly** : Lorsque ce paramètre vaut true, le cookie ne sera accessible que par le protocole HTTP. Cela signifie que le cookie ne sera pas accessible via des langages de scripts, comme Javascript.

**Options** : Un tableau associatif qui peut avoir comme clés expires, path, domain, secure, httponly et samesite.

#### La fonction setrawcookie()

Envoie un cookie sans encoder sa valeur en URL.

**Syntaxe :** setrawcookie ( string \$name, string \$value = ?, int \$expires\_or\_options = 0, string \$path = ?, string \$domain = ?, bool \$secure = false, bool \$httponly = false, array \$options = [] ) : bool

setrawcookie() est identique à setcookie() excepté que la valeur du cookie ne sera pas automatiquement encodée URL lors de l'envoi au navigateur.

#### Les sessions : Introduction

- Le support des sessions de PHP est un moyen de préserver des données entre plusieurs accès.
- Le support des sessions vous permet de stocker des données entre les requêtes dans le tableau super-globale `$_SESSION`.
- Pour utiliser les sessions, il faut que `session.auto_start` soit activé dans `php.ini` ou la fonction `session_start()` soit appelée.
- Les variables de type resource ne peuvent être stockées dans une session.
- Si vous débutez une session en appelant la fonction `session_start()`, un enregistrement dans la session ne sera pas créé tant que la variable ne sera pas enregistrée en utilisant la fonction `session_register()` ou en ajoutant une clé à la variable super-globale `$_SESSION`.
- Si vous activez `session.auto_start` alors le seul moyen de mettre des objets en session est de charger les définitions de classe avec `auto-prepend_file` dans lequel vous chargez les définitions dont vous avez besoin, ou bien vous devrez utiliser `serialize()` sur votre objet, et `unserialize()` pour le récupérer.

#### Les sessions : Configuration à l'exécution

- Le comportement de ces fonctions est affecté par la configuration dans le fichier php.ini.
- **Options de configuration pour les sessions :**
  - session.save\_path : valeur par défaut est = ""
  - session.gc\_maxlifetime : valeur par défaut est = "1440"
  - session.serialize\_handler : valeur par défaut est = "php"
  - session.cookie\_path : valeur par défaut est = "/"
  - session.cookie\_httponly : valeur par défaut est = "0"
  - session.cache\_limiter : valeur par défaut est = "nocache"
  - session.cache\_expire : valeur par défaut est = "180"
  - session.use\_trans\_sid : valeur par défaut est = "0"
  - session.trans\_sid\_tags : valeur par défaut est = "a=href,area=href,frame=src,form=
  - session.upload\_progress.cleanup : valeur par défaut est = "1"
  - ...
- Liste des options de configuration : <https://www.php.net/manual/fr/session.configuration.php>

## 02 - Sécuriser les données

### Utilisation des sessions et des cookies



#### Les sessions : Fonctions

##### **session\_abort :**

Abandonne les changements sur le tableau de session et termine la session

##### **session\_cache\_expire :**

Récupère et/ou définit le délai d'expiration du cache

##### **session\_cache\_limiter :**

Lit et/ou modifie le limiteur de cache de session

##### **session\_commit :**

Alias de session\_write\_close

##### **session\_create\_id :**

Crée un nouvel ID de session

##### **session\_decode :**

Décode les données encodées de session

##### **session\_destroy :**

Détruit une session

##### **session\_encode :**

Encode les données de session

##### **session\_gc :**

Exécute le ramasse miette des données de session

##### **session\_get\_cookie\_params :**

Lit la configuration du cookie de session

##### **session\_id :**

Lit et/ou modifie l'identifiant courant de session

##### **session\_module\_name :**

Lit et/ou modifie le module de session courant

##### **session\_name :**

Lit et/ou modifie le nom de la session

##### **session\_regenerate\_id :**

Remplace l'identifiant de session courant par un nouveau

##### **session\_register\_shutdown :**

Fonction de fermeture de session

##### **session\_reset :**

Réinitialise le tableau de session avec les valeurs originales

##### **session\_save\_path :**

Lit et/ou modifie le chemin de sauvegarde des sessions

##### **session\_set\_cookie\_params :**

Modifie les paramètres du cookie de session

##### **session\_set\_save\_handler :**

Configure les fonctions de stockage de sessions

##### **session\_start :**

Démarre une nouvelle session ou reprend une session existante

##### **session\_status :**

Détermine le statut de la session courante

##### **session\_unset :**

Détruit toutes les variables d'une session

##### **session\_write\_close :**

Écrit les données de session et ferme la session

#### Les sessions : Constantes prédefinies

- Ces constantes sont définies par cette extension, et ne sont disponibles que si cette extension a été compilée avec PHP, ou bien chargée au moment de l'exécution.
- **SID (string)** : Constante contenant le nom de la session et l'identifiant en cours, sous la forme "name=ID" ou une chaîne vide si l'identifiant de session a été défini dans un cookie de session. C'est la même valeur que celle renvoyée par la fonction session\_id().
- **PHP\_SESSION\_DISABLED (int)** : Valeur renvoyée par session\_status() si la session est désactivée.
- **PHP\_SESSION\_NONE (int)** : Valeur renvoyée par session\_status() si la session est activée, mais que la session n'existe pas.
- **PHP\_SESSION\_ACTIVE (int)** : Valeur renvoyée par session\_status() si la session est activée, et que la session existe.

#### Les sessions : Classes et Interfaces

- SessionHandler :
  - Définition : c'est une classe spéciale qui peut être utilisée pour exposer le gestionnaire de sauvegarde de session courant en interne de PHP, par héritage.
  - Fonctions : open, close, read, write, destroy, gc et create\_sid
- SessionHandlerInterface :
  - Définition : c'est une interface qui définit un prototype minimal pour la création d'un gestionnaire de session personnalisé.
  - Méthodes : open, close, read, write, destroy et gc.
- SessionIdInterface :
  - c'est une interface qui définit des méthodes optionnelles pour la création d'un gestionnaire de session personnalisé.
  - Méthode : create\_sid.
- SessionUpdateTimestampHandlerInterface :
  - Définition : c'est une interface qui définit des méthodes optionnelles pour la création d'un gestionnaire de session personnalisé.
  - Méthodes : updateTimestamp et validateId.



## CHAPITRE 2

### Sécuriser les données

1. Utilisation des sessions et des cookies
2. **Sécurisation des données**

#### Introduction

- La sécurité d'un serveur web PHP est fragile puisqu'il permet d'accéder aux fichiers, d'exécuter des commandes, et d'ouvrir des connexions réseaux.
- La common gateway interface (CGI) est une interface de serveurs Web qui permet un échange de données normalisé entre des applications et des serveurs externes. Ainsi, les pages HTML ne sont pas entièrement disponible sur le serveur HTTP qui exécute un autre programme, puis retourne le contenu généré.
- Utiliser PHP comme un exécutable CGI est une possibilité pour les cas où l'on ne veut pas l'utiliser comme un module du serveur web (comme Apache), ou bien lorsque l'on souhaite l'utiliser en combinaison avec un gestionnaire CGI complémentaire, afin de créer un environnement de script sécurisé.
- Lorsque PHP est utilisé en tant que module Apache, celui-ci hérite des permissions accordées à l'utilisateur faisant tourner Apache ce qui peut impacter la sécurité et les autorisations.
- Une erreur de sécurité fréquente est de donner à l'utilisateur Apache les droits de superadministrateur ("root"), ou d'accroître les possibilités d'Apache d'une quelconque autre façon.
- PHP, est constamment testé et amélioré. Chaque nouvelle version rassemble des modifications majeures ou mineures, aussi bien pour renforcer la sécurité, que pour réparer des problèmes de conception et de configuration, ainsi que d'autres points qui peuvent affecter la sécurité et la stabilité globale de votre système.

## 02 - Sécuriser les données

### Sécurisation des données



#### Sécurité des sessions

- Pour protéger les utilisateurs d'une tactique simple, la directive session.use\_only\_cookies doit être activée. Dans ce cas, les cookies doivent être activés obligatoirement côté client sinon les sessions ne fonctionneront pas.
- Implémenter SSL/TLS sur le serveur et le rendre obligatoire pour les utilisateurs. HSTS devrait être utilisé pour améliorer également la sécurité.
- Activer session.use\_strict\_mode quand le gestionnaire de sauvegarde des sessions le supporte, ce qui permet de rejeter un identifiant de session non initialisé, et d'en créer un nouveau.
- Utiliser les timestamps basés sur le gestionnaire de session, et regénérer les identifiants de session en utilisant la fonction session\_regenerate\_id() avec les procédures recommandées.
- session\_regenerate\_id() n'effacera pas les sessions anciennes par défaut. Si un utilisateur accède à une session obsolète (session ayant expirée), l'accès à cette session doit être refusé en utilisant des timestamps. Il est également recommandé de supprimer le statut d'authentification de toutes les sessions utilisateurs sinon cela peut représenter un axe d'attaque.
- Utiliser la fonction session\_gc() pour supprimer les données obsolètes depuis le stockage des données de sessions.
- Utiliser l'option 'read\_and\_close' avec la fonction session\_start(). session\_start(['read\_and\_close'=>1]); va clôturer la session aussi vite que possible après la mise à jour de la variable globale \$\_SESSION en utilisant la fonction session\_commit().
- N'utilisez jamais de données confidentielles comme préfixe. Si l'identifiant utilisateur est confidentiel, vous devriez utiliser la fonction hash\_hmac().
- Utiliser une clé de hachage sécurisé à usage unique comme clé d'auto-identification en utilisant la fonction setcookie(). Utiliser un hachage sécurisé plus fort que SHA-2. i.e. SHA-256 ou supérieur avec des données aléatoires depuis la fonction random\_bytes() ou via /dev/urandom.
- Utiliser la fonction output\_add\_rewrite\_var() pour la protection des fraudes de requêtes intersites.

#### Sécurisation des configurations INI de session

- **session.cookie\_lifetime=0** : Elle informe les navigateurs de ne pas stocker le cookie dans un espace de stockage permanent. Aussi, lorsque le navigateur se ferme, le cookie d'identification de session est supprimé immédiatement.
- **session.use\_cookies=On et session.use\_only\_cookies=On** : Si session.use\_only\_cookies=Off, le module de session utilisera les valeurs de l'identifiant de sessions définies par les variables GET/POST/URL fournies, et le cookie de l'identifiant de session ne sera pas initialisé.
- **session.use\_strict\_mode=On** : Ce mode évite que le module de session utilise un identifiant de session non initialisé
- **session.cookie\_httponly=On** : Permet de refuser l'accès à un cookie de session depuis javascript. Cette configuration évite qu'un cookie ne soit corrompu par une injection Javascript.
- **session.cookie\_secure=On** : Permet d'accéder au cookie d'identifiant de session uniquement lorsque le protocole est HTTPS.
- **session.cookie\_samesite="Lax" ou session.cookie\_samesite="Strict"** : Les cookies utilisant Lax seront accessibles via une requête GET originaire d'un autre domaine, alors que les cookies utilisant Strict ne le seront pas. Cet attribut est une façon de mitiger les attaques de fraudes de requêtes intersites.
- **session.gc\_maxlifetime=[choisissez le plus petit possible]** : est une configuration pour supprimer l'identifiant de session obsolète.
- **session.use\_trans\_sid=Off** : La désactivation de la gestion des identifiants de session de façon transparente permet de sécuriser un peu plus les identifiants de session en éliminant la possibilité d'une injection d'identifiant de session ou de fuite de cet identifiant.
- **session.trans\_sid\_tags=[drapeaux limités]** : (PHP 7.1.0 >=) Les développeurs ne doivent pas réécrire de drapeaux HTML non nécessaires. La valeur par défaut doit être suffisante pour la plupart des utilisations.

### Sécurisation des configurations INI de session (suite)

- **session.trans\_sid\_hosts=[hôtes limités]** : (PHP 7.1.0 >=) Ce paramètre définit une liste blanche des hôtes qui sont autorisés à réécrire les identifiants de session transparents.
- **session.referer\_check=[URL d'origine]** : Lorsque le paramètre session.use\_trans\_sid est actif. Ce paramètre réduit les risques d'injection d'identifiant de session.
- **session.cache\_limiter=nocache** : S'assure que le contenu HTTP n'est pas mis en cache pour les sessions authentifiées. Permet la mise en cache que pour les contenus qui ne sont pas privés.
- **session.sid\_length="48"** : (PHP 7.1.0 >=) La longueur des identifiants de session permet d'avoir des identifiants de session plus forts. Les développeurs doivent utiliser des identifiants de session d'au moins 32 caractères. Au moins 26 caractères sont requis lorsque session.sid\_bits\_per\_character="5".
- **session.sid\_bits\_per\_character="6"** : (PHP 7.1.0 >=) Le nombre d'octets présents dans un caractère d'identifiant de session.
- **session.hash\_function="sha256"** : (PHP 7.1.0 <) Une fonction de hachage forte va générer un identifiant de session fort.
- **session.save\_path=[dossier non lisible par tout le monde]** : Si ce paramètre est défini à un dossier accessible en lecture par tout le monde, comme /tmp (par défaut), les autres utilisateurs du serveur seront capables de récupérer les sessions en listant les fichiers présents dans ce répertoire.

### Sécurité des fichiers

- Puisque PHP a été fait pour permettre aux utilisateurs d'accéder aux fichiers, il est possible de créer un script PHP qui vous permet de lire des fichiers tels que /etc/password, de modifier les connexions ethernet, lancer des impressions de documents, etc.
- Il y a deux mesures primordiales à prendre pour éviter ces manœuvres :
  - Limiter les permissions de l'utilisateur web PHP.
  - Vérifier toutes les variables liées aux chemins et aux fichiers qui sont fournis.
- Si votre système d'identification permet aux utilisateurs de créer leur propre login, et qu'un utilisateur choisi le login ..//etc/, le système est de nouveau exposé. Suivant votre système d'exploitation, vous devrez protéger un grand nombre de fichiers, notamment les entrées de périphériques, (/dev/ ou COM1), les fichiers de configuration (fichiers /etc/ et .ini), les lieux de stockage d'informations (/home/, My Documents), etc. Pour cette raison, il est généralement plus sûr d'établir une politique qui interdit TOUT sauf ce que vous autorisez.

### Sécurité des bases de données

- Pour lire ou stocker des informations, vous devez vous connecter au serveur de bases de données, envoyer une requête valide, lire le résultat et refermer la connexion. De nos jours, le langage le plus courant pour ce type de communication est le langage SQL. Cela signifie qu'une requête SQL est capable de contourner les contrôles et vérifications, comme les identifications, et parfois, les requêtes SQL ont accès aux commandes d'administration.
- L'injection SQL directe est une technique où un pirate modifie une requête SQL existante pour afficher des données cachées, ou pour écraser des valeurs importantes, ou encore exécuter des commandes dangereuses pour la base. Ces attaques sont généralement basées sur l'exploitation de code qui n'est pas écrit de manière sécuritaire.
- Ne vous connectez jamais sur une base de données en tant que super utilisateur ou propriétaire de la base. Utilisez toujours un utilisateur adapté, avec des droits très limités. N'affichez jamais d'informations spécifiques à la base, et notamment des informations concernant le schéma.
- Utilisez des requêtes préparées avec des variables liées. Elles sont disponibles avec PDO, MySQLi ainsi que d'autres bibliothèques.
- Vérifiez que les données ont bien le type attendu. Si l'application attend une entrée numérique, vérifiez vos données avec la fonction ctype\_digit(), ou bien modifiez automatiquement le type avec la fonction settype(), ou encore avec sprintf().
- Si la couche de base de données ne suppose pas les variables liées, alors, mettez entre guillemets toutes les valeurs non numériques qui sont passées à la base de données avec la fonction spécifique à la base de données d'échappement de caractères (e.g. mysql\_real\_escape\_string(), sqlite\_escape\_string(), etc.).
- Il est recommandé d'établir des connexions au serveur avec le protocole SSL, pour chiffrer les échanges clients/serveur, afin d'améliorer la sécurité. Vous pouvez aussi utiliser un client SSH pour chiffrer la connexion entre les clients et le serveur de bases de données.
- Dans le cas de données vraiment sensibles, si la représentation originale n'est pas nécessaire (pour affichage, ou comparaison), utiliser un hash est une bonne solution.

#### Sécurité des données transmises par les internautes

- Il est vivement recommandé d'examiner minutieusement votre code pour vous assurer qu'il n'y a pas de variable envoyée par le client web qui ne soit pas suffisamment vérifiée avant utilisation.
- Il est conseillé de répondre aux questions suivantes :
  - Est-ce que ce script n'affectera que les fichiers prévus ?
  - Est-il possible que des valeurs incohérentes ou inattendues soient exploitées ici ?
  - Est-ce que ce script peut être utilisé dans un but différent de celui attendu ?
  - Est-ce que ce script peut être utilisé malicieusement, en conjonction avec d'autres ?
  - Est-ce que toutes les actions seront correctement historisées ?
- Envisager de supprimer l'acquisition automatique des variables (`register_globals`), ou encore toute option qui pourrait conduire à vous tromper sur la validité, la source, ou la valeur d'une variable.
- Travailler avec PHP en mode `error_reporting(E_ALL)`, peut aussi vous aider à détecter que certaines variables sont utilisées avant d'être exploitées ou initialisées.

#### Sécurité de rapport d'erreurs

- Une tactique d'attaque standard consiste à faire faire des erreurs au système, et à analyser les types des erreurs qui sont retournées, ainsi que leur contexte. Cela permet à l'attaquant d'obtenir des informations à propos du serveur, en vue de détecter de possibles faiblesses.
- Il est particulièrement dangereux d'exécuter du code de sources connues avec des gestionnaires de débogage inclus, ou de travailler avec des techniques de débogage répandues.
- Indépendamment de la méthode de gestion des erreurs, la possibilité de tester un système pour identifier des erreurs revient à fournir à un attaquant plus d'informations sur votre système.
- Une erreur de fonction peut indiquer si un système supporte une base de données spécifique, ou bien donner des indices quant à la façon dont une page a été conçue ou développée. Cela peut orienter l'intrus vers les ports de cette base de données ou bien vers une attaque liée à cette application.
- Une erreur de fichier, ou une erreur générale de PHP, peut indiquer quelles sont les permissions du serveur web, ainsi que la structure et l'organisation des fichiers. Les gestionnaires d'erreurs utilisateurs peuvent aussi agraver ce problème, en permettant l'exploitation facile d'informations préalablement "cachées".
- Il est recommandé de scruter toutes les fonctions, et d'essayer de traiter toutes les erreurs. Désactiver le rapport d'erreur, dès que le script est en production. Enfin, utiliser les fonctions de gestion des erreurs de PHP pour créer vos propres gestionnaires d'erreurs.
- Utiliser la fonction `error_reporting()`, pour vous aider à sécuriser votre code, et détecter certaines utilisations dangereuses de variables.
- Tester votre code, avant le déploiement, avec `E_ALL`, vous pouvez rapidement repérer les variables qui ne sont pas protégées. Une fois que le code est prêt à être déployé, vous devriez soit désactiver complètement le rapport d'erreur en passant 0 à la fonction `error_reporting()`, soit en désactivant l'affichage des erreurs en utilisant l'option de configuration `display_errors` de `php.ini`. Si vous choisissez la seconde solution, vous devriez également définir le chemin vers votre fichier de log en utilisant la directive de configuration `error_log`, et en activant la directive `log_errors`.

#### Exemples d'attaques

**Injection SQL** : La solution à cela est d'utiliser des requêtes SQL paramétrées et des objets de données PHP (PDO) pour différencier les données des parties de requêtes.

**Traversée du répertoire** : Il faut appliquer les autorisations appropriées en fonction du statut de l'utilisateur.

**Scripts intersites (attaques XSS)** : Parmi les solutions filtrer toutes les données externes ainsi que d'utiliser les fonctions existantes PHP.

**Falsification de demandes intersites** : Il est recommandé d'utiliser HTTPS pour crypter la connexion. L'utilisation de POST au lieu de GET ainsi qu'un mécanisme d'authentification dans les formulaires.

**Stockage des mots de passe** : Il existe deux façons importantes de stocker des mots de passe en toute sécurité, en utilisant un algorithme de hachage et du sel.

**Détournement de session** : Vérifier les détails de l'emplacement et les informations du navigateur et les faire correspondre avec les données historiques peut fournir des informations sur la session en cours.

**Fixation de session** : L'utilisation des cookies et la régénération d'identifiant de session servent à limiter ces attaques.

**Données de session exposées** : Éviter d'utiliser un magasin de sessions partagé.

**Attaques XML** : De préférence utiliser un ensemble de caractères sur liste blanche pour vous assurer que les caractères indésirables ou spéciaux ne sont pas acceptés.

## 02 - Sécuriser les données

### Sécurisation des données

#### Journal des modification PHP version 8.1.2

- **Cœur :**
  - Correction du bug #81216 (l'opérateur Nullsafe perd le nom de la propriété dynamique).
  - Correction du bug #81585 (cached\_chunks ne compte pas dans real\_size à l'arrêt).
  - Correction du bug #7757 (la constante finale multi-héritée provoque une erreur fatale).
- **Hacher :**
  - Correction du bug #7759 (types de retour incorrects pour hash() et hash\_hmac()).
  - Correction du bug #7826 (Nom d'argument incohérent dans hash\_hmac\_file et hash\_file).
- **MySQL et :**
  - Correction d'un bug où les grands bigints pouvaient être tronqués.
- **Ligne de lecture :**
  - Correction du bug #81598 (Impossible de saisir des caractères Unicode dans le shell interactif PHP 8).
- **Standard :**
  - Correction du bug #7748 (gethostbyaddr génère une chaîne binaire).
  - Correction du bug #7815 (php\_uname ne reconnaît pas les dernières versions de Windows).
- **Liste complète publiée le : 20 janv. 2022 source : <https://www.php.net/ChangeLog-8.php#8.1.2>**

## PARTIE 3 : Manipuler les données

### Références et ressources



#### Références et ressources

<https://www.lebigdata.fr/base-de-donnees>

<https://sql.sh/>

[https://www.w3schools.com/sql/sql\\_intro.asp](https://www.w3schools.com/sql/sql_intro.asp)

<https://dev.mysql.com/doc/connectors/en/apis-php-pdo-mysql.html>

<https://www.php.net/manual/fr/features.cookies.php>

<https://www.php.net/manual/fr/book.session.php>

<https://www.getastral.com/blog/php-security/php-security-guide/>

<https://www.ionos.fr/digitalguide/sites-internet/developpement-web/quest-ce-que-la-cgi/>



## PARTIE 4

### Réaliser un site web avec l'architecture MVC

Dans ce module, vous allez :

- Maitriser le modèle MVC
- Créer différentes couches du modèle MVC
- Acquérir une bonne connaissance sur les Web services
- Manipuler SOAP
- Maitriser REST API



24 heures

# CHAPITRE 1

## Développer des sites dynamiques en MVC



**Ce que vous allez apprendre dans ce chapitre :**

- Présentation du MVC (Modèle - Vue - Contrôleur)
- Crédit des composants Contrôleur
- Crédit des Vues
- Crédit des composants Modèle



**08 heures**

# CHAPITRE 1

## Développer des sites dynamiques en MVC



1. **Présentation du MVC (Modèle - Vue - Contrôleur)**
2. Création des composants Contrôleur
3. Création des Vues
4. Création des composants Modèle

# Développer des sites dynamiques en MVC

## 01 - Présentation du MVC (Modèle - Vue - Contrôleur)



### Introduction

- Dans l'architecture 3-tiers, si une vue modifie les données, toutes les vues concernées par la modification doivent être mises à jour, d'où l'utilité de l'utilisation du MVC au niveau de la couche de présentation. La couche de présentation permet donc d'établir des règles du type « mettre à jour les vues concernant X si Y ou Z sont modifiés ». (Wikipédia)
- MVC (Modèle-Vue-Contrôleur) est un modèle de conception, il est donc indépendant du langage de programmation. Il met l'accent sur la séparation entre la logique métier et l'affichage du logiciel.

MVC est très utilisé dans les applications web.  
Il est décomposé en 3 parties :

Le contrôleur : qui gère la navigation.

Le modèle : qui gère la structuration et l'accès aux données.

La vue : qui gère l'interface utilisateur.

- Les modèles et les contrôleurs sont généralement des classes.
- Les vues sont généralement des Templates HTML ou PDF.

### Avantages & Inconvénients de l'architecture MVC

#### Quelques Avantages

- Meilleure organisation du code.
- Possibilité de réutiliser le code.
- Meilleure gestion de temps pour la maintenance et l'évolution du site.
- Plus de facilité pour les tests unitaires.
- Conception claire et efficace grâce

#### Quelques Inconvénients

- Complexité pour de petites applications. Le temps accordé à l'architecture peut ne pas être rentable pour le projet.
- Augmentation de la difficulté d'implémentation.
- Le nombre important de fichiers représente une charge non négligeable dans un projet.

# Développer des sites dynamiques en MVC

## 01 - Présentation du MVC (Modèle - Vue - Contrôleur)



### Frameworks MVC pour PHP

L'utilisation de Framework peut s'avérer d'un grand avantage au développeur. Il permet de faciliter le travail et réduire le temps de réalisation tout en profitant des avantages qu'offre l'architecture.

#### Exemples de frameworks utilisant l'architecture MVC

- Lavarel : <https://laravel.com/>
- Lumen : <https://lumen.laravel.com/>
- Symfony : <https://symfony.com/>
- Cake PHP : <https://cakephp.org/>
- Zend Framework : <https://framework.zend.com/>
- Yii Framework : <https://www.yiiframework.com/>
- Phalcon : <https://phalcon.io/en-us>
- CodeIgniter : <https://codeigniter.com/>
- Slim : <https://www.slimframework.com/>
- FuelPHP : <https://fuelphp.com/>
- Aura : <https://auraphp.com/>

# Développer des sites dynamiques en MVC

## 01 - Présentation du MVC (Modèle - Vue - Contrôleur)

### Architecture MVC

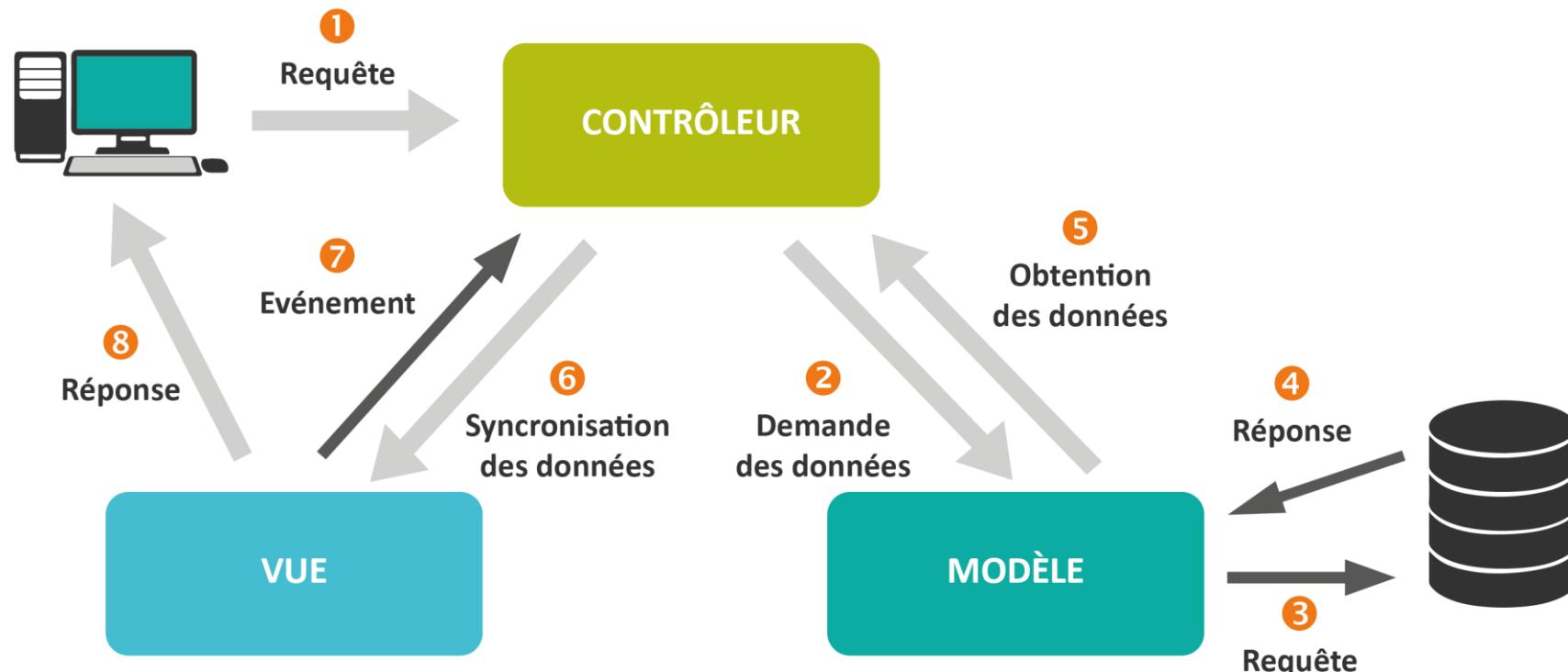


Fig. : Architecture MVC

# Développer des sites dynamiques en MVC

## 01 - Présentation du MVC (Modèle - Vue - Contrôleur)



### Contrôleur

#### Le contrôleur est chargé de :

- Reconnaître l'action (événement) à réaliser pour l'exécuter.
- Demander la construction du modèle (données à renvoyer ou afficher en sortie).
- Tester les erreurs et récupérer les éventuelles exceptions pour éviter les erreurs.
- Appeler la vue (ou la vue d'erreur) pour afficher le résultat de l'action.

Le contrôleur récupère les données du formulaire pour les traiter.

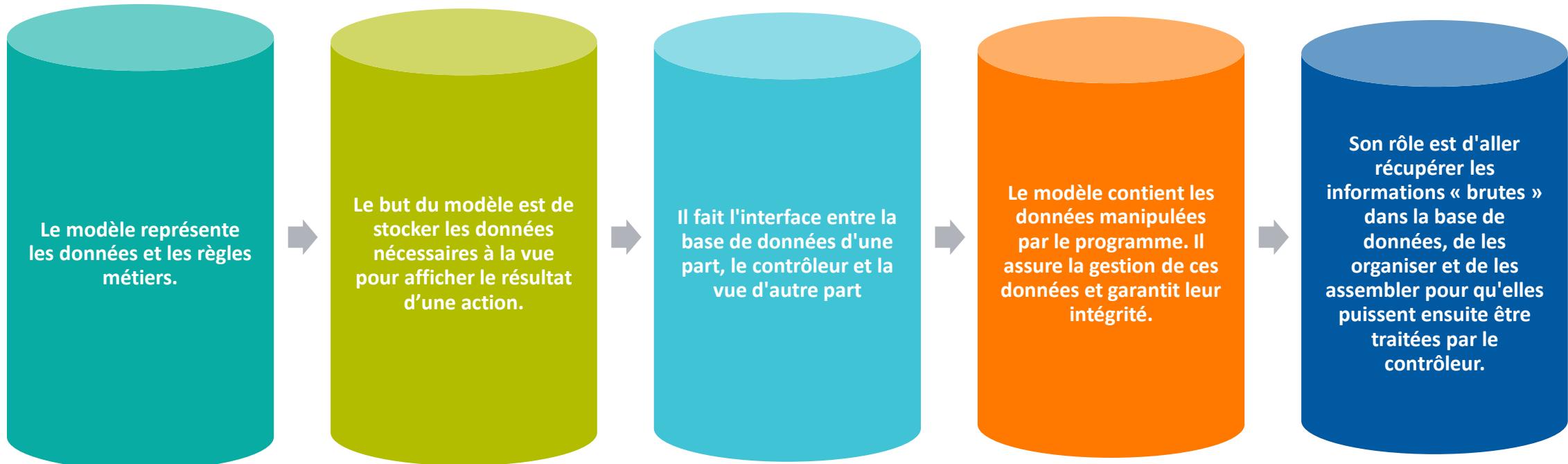
Le contrôleur est chargé de la synchronisation du modèle et de la vue.

Le contrôleur est souvent réparti en plusieurs objets.

# Développer des sites dynamiques en MVC

## 01 - Présentation du MVC (Modèle - Vue - Contrôleur)

### Les Modèles



### Les Vues





## CHAPITRE 1

# Développer des sites dynamiques avec MVC en mode natif

1. Présentation du MVC (Modèle - Vue - Contrôleur)
- 2. Crédit des composants Contrôleur**
3. Crédit des Vues
4. Crédit des composants Modèle

### Création d'un fichier contrôleur

- index.php peut être considéré comme contrôleur. Il établit le lien avec le modèle et la vue qui seront détaillés dans les prochaines sections.
- Nous pourrons créer un nouveau contrôleur si besoin.

#### Exemple :

- Prérequis : Table de commentaires « comments » dans la Base de données « test » ayant les champs suivants : id, auteur, commentaire et date\_commentaire
- Le contrôleur index.php permet de gérer la liste des derniers commentaires.

```
<?php
    require('modele.php');
    $comments = getComments(); //fonction présente dans modele.php
    require('view.php');
?>
```



## CHAPITRE 1

### Développer des sites dynamiques en MVC

1. Présentation du MVC (Modèle - Vue - Contrôleur)
2. Création des composants Contrôleur
- 3. Création des Vues**
4. Création des composants Modèle

### Création d'un fichier vue

- L'affichage est important pour un site Web. Il faut bien soigner la vue en utilisant les notions HTML et XHTML, ...

Exemple :

- Prérequis : Table de commentaires « comments » dans la Base de données « test » ayant les champs suivants : id, auteur, commentaire et date\_commentaire.
- Le fichier view.php permet dans cet exemple d'afficher les commentaires.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Mon Site Web Dynamique</title>
    <link href="style.css" rel="stylesheet" />
</head>
<body>
    <h1>Bonjour </h1>
    <p>Derniers commentaires du site :</p>
    <?php
        while ($data = $comments->fetch()){
    ?>
    <div class="news">
        <p>
            <?= nl2br(htmlspecialchars($data['commentaire'])) ?>
            <br />
            <em><a href="#">Commentaires</a></em>
        </p>
    </div>
    <?php } ?>
</body>
</html>
```

# CHAPITRE 1

## Développer des sites dynamiques en MVC



1. Présentation du MVC (Modèle - Vue - Contrôleur)
2. Création des composants Contrôleur
3. Création des Vues
- 4. Création des composants Modèle**

### Création d'un fichier modèle

- Dans une architecture MVC, on commence en général par écrire le modèle.
- Le code fourni par le modèle permet de récupérer les données grâce notamment à des requêtes SQL.

Exemple :

- Prérequis : Table de commentaires « comments » dans la Base de données « test » ayant les champs suivants : id, auteur, commentaire et date\_commentaire.
- Le fichier model.php permet dans cet exemple d'établir la connexion avec la base de donnée uniquement.

```
<?php
    function bdConnect()
    {
        try
        {
            $bd = new PDO('mysql:host=localhost;dbname=test;charset=utf8', 'root', 'root');
            return $bd;
        }
        catch(Exception $e)
        {
            die('Erreur : '.$e->getMessage());
        }
    }
?>
```

### Exemple modèle

- Suite à l'exemple précédent, nous pourrons ajouter d'autres fonctions qui récupèrent les commentaires d'un ID ou les 5 derniers commentaires par exemple :

```
function getCommentsById($Id)
{
    $bd = bdConnect();
    $comments_id = $bd->prepare('
        SELECT id, auteur, commentaire, DATE_FORMAT(date_commentaire, \'%d/%m/%Y à %Hh%imin%ss\') AS comment_date_fr
        FROM comments
        WHERE id = ? ORDER BY date_commentaire DESC');
    $comments_id->execute(array($Id));
    return $comments_id;
}

function getLastFiveComments()
{
    $bd = bdConnect();
    $req = $bd->query('
        SELECT id, auteur, commentaire, DATE_FORMAT(date_commentaire, \'%d/%m/%Y à %Hh%imin%ss\') AS comment_date_fr
        FROM comments
        ORDER BY date_commentaire DESC
        LIMIT 0, 5');
    return $req;
}
```

### Questions

#### Question n°1 :

Dans MVC, quelle partie s'occupe d'afficher la page en HTML ?

1. Modèle
2. Vue
3. Contrôleur

#### Question n°2 :

Laquelle de ces phrases décrit le mieux les différentes phases d'exécution d'un code en MVC ?

1. Le routeur appelle le bon contrôleur, qui récupère des informations depuis le modèle qu'il passe ensuite à la vue
2. Le contrôleur appelle le modèle, qui passe le tout au routeur pour que celui-ci puisse afficher la vue
3. Le routeur appelle le modèle, qui transmet les données au contrôleur qui passe à la vue

#### Question n°3 :

Pourquoi le routeur se trouve souvent dans un fichier nommé index.php à la racine ?

1. Parce que cela permet d'accélérer les traitements
2. Parce que c'est obligatoire pour faire du routing
3. Parce que c'est une convention
4. Parce que le routeur est le premier élément qu'on appelle

#### Question 4 :

MVC est un modèle de conception utilisé pour découpler l'interface utilisateur, les données et \_\_\_\_\_.

1. Logique d'application
2. Application
3. Logique
4. Rien

### Questions

#### Question 5 :

Le \_\_\_\_\_ aide à transmettre les données du contrôleur à afficher et à afficher au contrôleur.

1. Vue
2. Modèle
3. Routeur

#### Question 6 :

Il est utilisé pour afficher l'ensemble des données à l'aide du modèle :

1. Contrôleur
2. Vue
3. Routeur

#### Question 7 :

Pourquoi est-il conseillé de ne pas mélanger PHP, HTML et SQL dans un même fichier ?

1. Cela permet de travailler à plusieurs sur plusieurs fichiers en parallèle
2. Cela permet aux pages web de s'afficher plus rapidement
3. Cela facilite la maintenance du code

#### Question 8 :

En programmation, il est préférable d'écrire en...

1. Français
2. Anglais
3. Arabe
4. N'importe quelle langue

#### Question 9 :

Quand une page ne contient que du code PHP, on ouvre la balise <?php et...

1. On ne la ferme pas à la fin
2. On pense bien à ajouter un ?> à la fin

# Développer des sites dynamiques en MVC

## Quizz (Correction)



### Réponses

#### Question n°1 :

Dans MVC, quelle partie s'occupe d'afficher la page en HTML ?

1. Modèle
2. Vue
3. Contrôleur

#### Question n°2 :

Laquelle de ces phrases décrit le mieux les différentes phases d'exécution d'un code en MVC ?

1. Le routeur appelle le bon contrôleur, qui récupère des informations depuis le modèle qu'il passe ensuite à la vue
2. Le contrôleur appelle le modèle, qui passe le tout au routeur pour que celui-ci puisse afficher la vue
3. Le routeur appelle le modèle, qui transmet les données au contrôleur qui passe à la vue

#### Question n°3 :

Pourquoi le routeur se trouve souvent dans un fichier nommé index.php à la racine ?

1. Parce que cela permet d'accélérer les traitements
2. Parce que c'est obligatoire pour faire du routing
3. Parce que c'est une convention
4. Parce que le routeur est le premier élément qu'on appelle

#### Question 4 :

MVC est un modèle de conception utilisé pour découpler l'interface utilisateur, les données et \_\_\_\_\_.

1. Logique d'application
2. Application
3. Logique
4. Rien

# Développer des sites dynamiques en MVC

## Quizz (Correction)



### Réponses

#### Question 5 :

Le \_\_\_\_\_ aide à transmettre les données du contrôleur à afficher et à afficher au contrôleur.

1. Vue
2. Modèle
3. Routeur

#### Question 6 :

Il est utilisé pour afficher l'ensemble des données à l'aide du modèle :

1. Contrôleur
2. Vue
3. Routeur

#### Question 7 :

Pourquoi est-il conseillé de ne pas mélanger PHP, HTML et SQL dans un même fichier ?

1. Cela permet de travailler à plusieurs sur plusieurs fichiers en parallèle
2. Cela permet aux pages web de s'afficher plus rapidement
3. Cela facilite la maintenance du code

#### Question 8 :

En programmation, il est préférable d'écrire en...

1. Français
2. Anglais
3. Arabe
4. N'importe quelle langue

#### Question 9 :

Quand une page ne contient que du code PHP, on ouvre la balise <?php et...

1. On ne la ferme pas à la fin
2. On pense bien à ajouter un ?> à la fin

# Développer des sites dynamiques en MVC

## Références et ressources



### Références et ressources

<https://bpesquet.developpez.com/tutoriels/php/evoluer-architecture-mvc/>

<https://openclassrooms.com/fr/courses/4670706-adoptez-une-architecture-mvc-en-php/4678736-comment-fonctionne-une-architecture-mvc>

<https://www.univ-orleans.fr/iut-orleans/informatique/intra/tuto/php/php-mvc.html>

## CHAPITRE 2

### Découvrir les Web services et les API REST



Ce que vous allez apprendre dans ce chapitre :

- Introduction aux Web services
- Manipulation de SOAP
- Manipulation de REST API



08 heures



## CHAPITRE 2

### Découvrir les Web services et les API REST

1. **Introduction aux Web services**
2. Manipulation de SOAP
3. Manipulation de REST API

### Introduction

Un Web Service est une application qui permet d'échanger des données avec d'autres applications web.

**Un Web Service fonctionne de la manière suivante :**

- Le client effectue une requête HTTP/SOAP pouvant contenir un corps formatté en XML ou JSON ;
- La requête est transmise à un serveur distant (via les protocoles SOAP ou HTTP) ;
- Une réponse est ensuite délivrée dans un format tel que XML, JSON ou HTML.

**Un Web Service utilise un modèle en couche :**

- L'invocation : elle décrit la structure des messages échangés par le client et le serveur ;
- La découverte : est la phase de recherche et de localisation des données demandées par le client, le plus souvent via le protocole UDDI (Universal Description Discovery and Integration). UDDI est un annuaire (une liste, un répertoire) de services fondé sur XML et plus particulièrement destiné aux services Web ;
- La description qui stipule les paramètres des fonctions et les types de données des services web utilisés. Son protocole standard est le WSDL (Web Services Description Language) qui repose sur la notation XML.

**WSDL est une grammaire XML permettant de décrire un service web. Une description WSDL est un document XML qui commence par la balise <definitions> et qui contient les balises suivantes :**

- **<binding>** : définit le protocole à utiliser pour invoquer le service web ;
- **<port>** : spécifie l'emplacement effectif du service ;
- **<service>** : décrit un ensemble de points finaux du réseau.

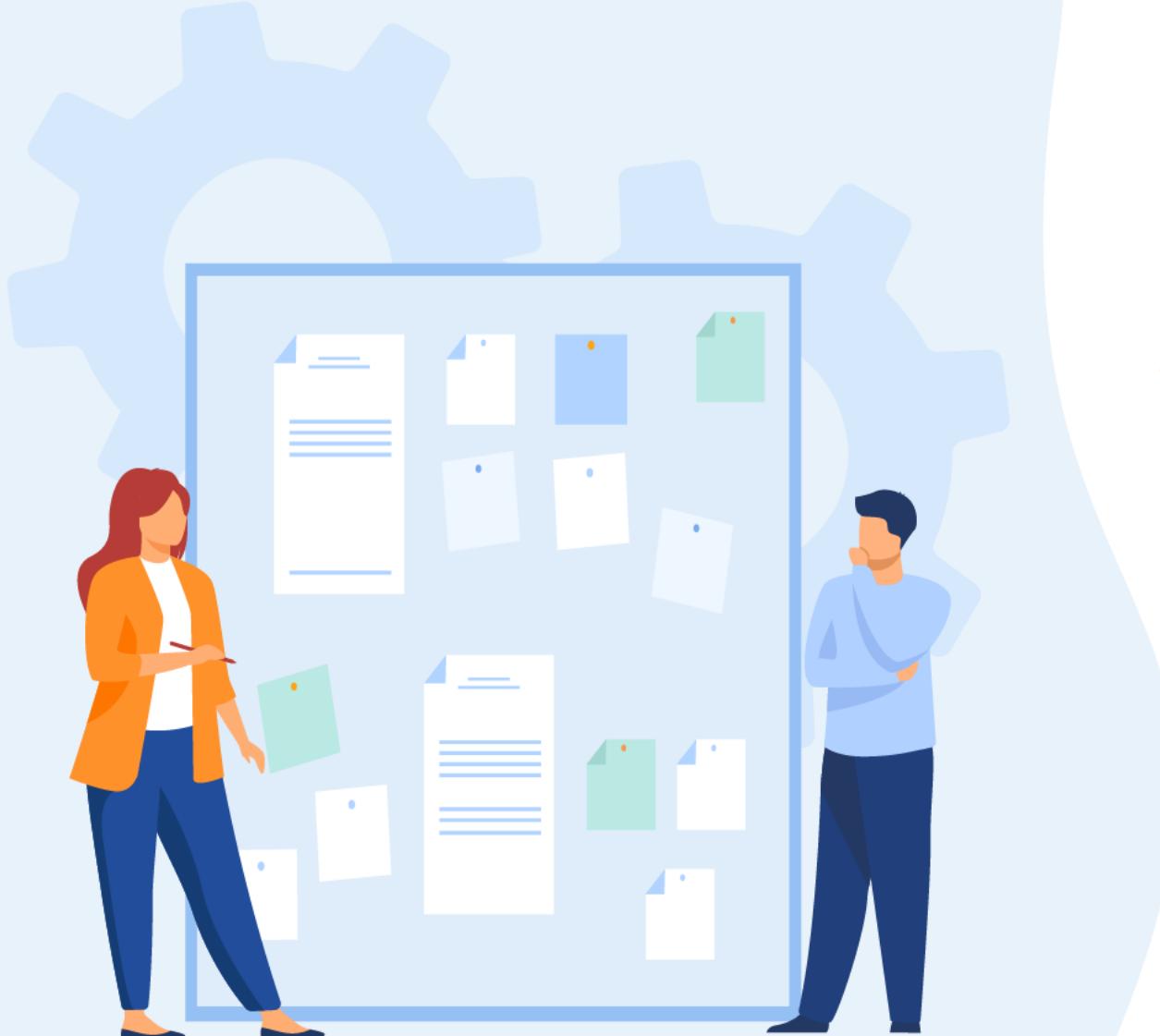
### Avantages & Inconvénients

#### Un Web Service a plusieurs avantages :

- Les services web permettent d'accéder à des fonctionnalités via internet
- Le client peut recevoir des informations d'un serveur distant sans être obligé de stocker toutes les données sur sa machine.
- Le serveur distant peut être interrogé par un grand nombre de client de manière simultanée.
- Les services Web utilisent des standards et protocoles ouverts, ce qui permet une grande compatibilité entre les services.
- Les informations échangées peuvent être chiffrées. C'est le cas en utilisant la technologie SSL présente dans le protocole HTTPS qui est devenue la norme pour les sites web depuis quelques années.

#### Inconvénients

- Un service web a toujours besoin d'un réseau (internet, intranet, ...) pour fonctionner.



## CHAPITRE 2

### Découvrir les Web services et les API REST

1. Introduction aux Web services
2. **Manipulation de SOAP**
3. Manipulation de REST API

### Définition API Web

- Une API (Application Programming Interface) Web permet d'utiliser les ressources, données ou fonctionnalités, d'une application web depuis une autre application.
- Une API est une interface de programmation, c'est-à-dire un ensemble de classes, de fonctions et de méthodes qui servent de façade à un logiciel. D'autres logiciels pourront donc accéder aux services de ce logiciel grâce à cette interface.

Comme toute application, les types d'APIs sont :

- L'API ouverte (ou Open API) disponible publiquement sans restriction ;
- L'API partenaire accessible via une licence ou des droits d'accès spécifiques ;
- L'API interne ou privée (Private API) conçue par une entreprise pour ses besoins.

#### Quelques avantages :

- Industrialiser l'échanges d'informations : une API permet de pouvoir interagir avec un système complexe sans connaître le détail du programme.
- Sécuriser et contrôler les échanges de données : Une API est souvent spécialisée dans un fonctionnement particulier ce qui simplifie son utilisation, sa compréhension et sa sécurisation.
- Optimiser et développer les usages : Une API permet également de facilement mettre à disposition des tiers, des fonctionnalités sans ouvrir l'accès aux systèmes sources.

Quelques exemples : Google Map, Twitter, Instagram, Gmail, ...

### Introduction protocole SOAP

SOAP (Simple Object Access Protocol) est un protocole de communication basé sur XML pour permettre aux applications de s'échanger des informations via HTTP.

SOAP est un protocole officiel géré par le W3C (World Wide Web Consortium).

SOAP permet l'accès aux services web et l'interopérabilité des applications à travers le web. SOAP est portable et donc indépendant de tous système d'exploitation et du type d'ordinateur.

SOAP est un protocole, il impose des règles intégrées. Ces standards assurent la conformité et sont ainsi privilégiés pour certains scénarios d'entreprise. Les standards de conformité intégrés incluent la sécurité, l'atomicité, la cohérence, l'isolement et la durabilité.

### Prérequis et Installation

- **Prérequis :** l'extension PHP libxml. Ceci signifie passer l'option de configuration --with-libxml
- L'extension libxml est activée par défaut, et elle peut être désactivée avec l'option --disable-libxml
- **Fonctions libxml :**
  - **libxml\_clear\_errors** : Vide le buffer d'erreur libxml
  - **libxml\_disable\_entity\_loader** : Désactive le chargement des entités externes
  - **libxml\_get\_errors** : Lit le tableau d'erreurs
  - **libxml\_get\_last\_error** : Lit la dernière erreur libxml
  - **libxml\_set\_external\_entity\_loader** : Change le chargeur d'entités externes par défaut
  - **libxml\_set\_streams\_context** : Configure le contexte de flux pour la prochaine opération libxml
  - **libxml\_use\_internal\_errors** : Désactive le rapport d'erreur libxml et les stocke pour lecture ultérieure
- Pour activer le support de SOAP, configurez PHP avec --enable-soap.
- Il existe 5 fonctions de configuration SOAP dans le fichier php.ini.

### Configuration

1. **soap.wsdl\_cache\_enabled int**
  - **Définition :** Active ou pas le cache WSDL..
  - **Valeur par défaut :** 1
2. **soap.wsdl\_cache\_dir string**
  - **Définition :** Configure le nom du dossier où l'extension SOAP peut mettre les fichiers de cache.
  - **Valeur par défaut :** /tmp
3. **soap.wsdl\_cache\_ttl int**
  - **Définition :** Configure la durée de vie des fichiers dans le cache, exprimé en secondes.
  - **Valeur par défaut :** 86400
4. **soap.wsdl\_cache int**
  - **Définition :** Si soap.wsdl\_cache\_enabled est actif, cette directive détermine le type de cache. Il peut prendre l'une des valeurs suivantes : WSDL\_CACHE\_NONE (0), WSDL\_CACHE\_DISK (1), WSDL\_CACHE\_MEMORY (2) ou WSDL\_CACHE\_BOTH (3). Cela peut aussi être configuré via le paramètre options dans les constructeurs de SoapClient ou SoapServer.
  - **Valeur par défaut :** 1
5. **soap.wsdl\_cache\_limit int**
  - **Définition :** Le nombre maximal de fichier WSDL en cache. La réception de nouveaux fichiers dans le cache, s'il est plein, causera l'effacement des plus anciens.
  - **Valeur par défaut :** 5

# Découvrir les Web services et les API REST

## 02 - Manipulation de SOAP

### La classe SoapClient

La classe SoapClient fournit un client pour les serveurs SOAP 1.1 et SOAP 1.2.

Les méthodes sont :

- **SoapClient::\_\_call** : Appelle une fonction SOAP (obsolète)
- **SoapClient::\_\_construct** : Constructeur SoapClient
- **SoapClient::\_\_doRequest** : Exécute une requête SOAP
- **SoapClient::\_\_getCookies** : Get list of cookies
- **SoapClient::\_\_getFunctions** : Retourne une liste de fonctions SOAP publiées
- **SoapClient::\_\_getLastRequest** : Retourne la dernière requête SOAP
- **SoapClient::\_\_getLastRequestHeaders** : Retourne les en-têtes de la dernière requête SOAP
- **SoapClient::\_\_getLastResponse** : Retourne la dernière réponse SOAP
- **SoapClient::\_\_getLastResponseHeaders** : Retourne les en-têtes de la dernière réponse SOAP
- **SoapClient::\_\_getTypes** : Retourne une liste des types SOAP
- **SoapClient::\_\_setCookie** : Définit un cookie pour les requêtes SOAP
- **SoapClient::\_\_ setLocation** : Configure l'URL du service Web à utiliser
- **SoapClient::\_\_setSoapHeaders** : Ajoute un en-tête SOAP pour les requêtes suivantes
- **SoapClient::\_\_soapCall** : Appelle une fonction SOAP

Propriétés :

- liste dans <https://www.php.net/manual/fr/class.soapclient.php#soapclient.props.-ssl-method>

### La classe SoapServer

La classe SoapServer fournit un serveur pour les protocoles SOAP 1.1 et SOAP 1.2.

#### Méthodes :

- **SoapServer::addFunction** : Ajoute une ou plusieurs fonctions qui vont gérer les requêtes SOAP
- **SoapServer::addSoapHeader** : Ajoute un en-tête SOAP à une réponse
- **SoapServer::\_\_construct** : Constructeur SoapServer
- **SoapServer::fault** : Emet une erreur SoapServer
- **SoapServer::getFunctions** : Retourne la liste des fonctions définies
- **SoapServer::handle** : Traite une requête SOAP
- **SoapServer::setClass** : Configure la classe qui sera utilisée pour gérer les requêtes SOAP
- **SoapServer::setObject** : Configure l'objet qui sera utilisé pour gérer les requêtes SOAP
- **SoapServer::setPersistence** : Active le mode persistant de SoapServer

#### Propriétés :

- private resource \$service;
- private ?SoapFault \$\_\_soap\_fault = null;

### SoapFault

Représente une erreur SOAP.

Méthodes :

- **SoapFault::\_\_construct** : Constructeur SoapFault.
- **SoapFault::\_\_toString** : Produit une représentation chaîne d'un objet SoapFault.

Méthodes héritées:

- **Exception::getMessage()** : Récupère le message de l'exception.
- **Exception::getPrevious()** : Retourne le Throwable précédent. (Throwable : est l'interface de base pour tout objet qui peut être jeté grâce à la déclaration throw, ceci inclus Error et Exception)
- **Exception::getCode()** : Récupère le code de l'exception.
- **Exception::getFile()** : Récupère le fichier dans lequel l'exception a été créée.
- **Exception::getLine()** : Récupère la ligne dans laquelle l'exception a été créée.
- **Exception::getTrace()** : Récupère la trace de la pile.
- **Exception::getTraceAsString()** : Récupère la trace de la pile en tant que chaîne.
- **Exception::\_\_toString()** : Représente l'exception sous la forme d'une chaîne.
- **Exception::\_\_clone()** : Clone l'exception.

Propriétés et Propriétés héritées :

- Liste dans <https://www.php.net/manual/fr/class.soapfault.php>

### SoapHeader

Représente un en-tête SOAP.

Méthodes :

- `SoapHeader::__construct` : Constructeur SoapHeader.

Propriétés :

- `public string $namespace;`
- `public string $name;`
- `public mixed $data = null;`
- `public bool $mustUnderstand;`
- `public string|int|null $actor;`

### SoapParam

Représente un paramètre dans un appel SOAP.

Méthodes :

- `SoapParam::__construct`: Constructeur SoapParam

Propriétés :

- `public string $param_name;`
- `public mixed $param_data;`

### SoapVar

Une classe représentant une variable ou un objet qui utilise des services SOAP.

#### Méthodes :

- SoapVar::\_\_construct : Constructeur SoapVar.

#### Propriétés :

- public int \$enc\_type;
- public mixed \$enc\_value = null;
- public ?string \$enc\_stype = null;
- public ?string \$enc\_ns = null;
- public ?string \$enc\_name = null;
- public ?string \$enc\_namens = null;

### Fonctions SOAP

- **is\_soap\_fault**
  - **Syntaxe :** is\_soap\_fault(mixed \$object): bool
  - **Définition :** Vérifie si SOAP retourne une erreur, sans utiliser les exceptions.
  - **Utilisation :** Pour l'utiliser, créez un objet SoapClient avec l'option exceptions mise à zéro ou à false. Dans ce cas, la méthode SOAP va retourner un objet spécial SoapFault, qui encapsule les détails de l'erreur (code d'erreur, message, acteur et détails). Si exceptions n'est pas configurée, SOAP va émettre une exception. is\_soap\_fault() vérifie si le paramètre fourni est un objet SoapFault.
- **use\_soap\_error\_handler**
  - **Syntaxe :** use\_soap\_error\_handler(bool \$enable = true): bool
  - **Définition :** Active le gestionnaire d'erreurs SOAP natif
  - **Utilisation :** Retourne la valeur précédemment utilisée. Si le paramètre handler est mis à true, le détail des erreurs du serveur SoapServer seront envoyées au client, comme message d'erreur SOAP. Si vaut false le gestionnaire d'erreur standard PHP sera utilisé. Le comportement par défaut est d'envoyer l'erreur au client sous forme de message SOAP.

```
<?php
$client = new SoapClient("some.wsdl", array('exceptions' => 0));
$result = $client->SomeFunction();
if (is_soap_fault($result)) {
    trigger_error("SOAP Fault: (faultcode: {$result->faultcode}, faultstring: {$result->faultstring})", E_USER_ERROR);
}
?>
```



## CHAPITRE 2

### Découvrir les Web services et les API REST

1. Introduction aux Web services
2. Manipulation de SOAP
- 3. Manipulation de REST API**

### Introduction

REST est un type d'architecture d'API qui signifie Representational State Transfer. Il a été inventé par l'Américain Roy Fielding dans les années 2000, afin de mettre en place des méthodes simples pour accéder à des services web.

Une API REST (également appelée API RESTful) est une interface de programmation d'application (API ou API web) qui respecte les contraintes du style d'architecture REST et permet d'interagir avec les services web RESTful.

Dans un service web REST, les requêtes effectuées sur l'URI d'une ressource produisent une réponse dont le corps est formaté dans un format tel que HTML, XML, JSON...

Les données REST sont représentées dans ce qu'on appelle des ressources. Une ressource peut être tout type d'objet nominal (on lui attribue un nom).

Les ressources sont regroupées dans un groupe que l'on appelle une collection.

Le World Wide Web est une application conçue selon l'architecture REST.

# Découvrir les Web services et les API REST

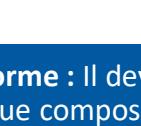
## 03 - Manipulation de REST API



### Exemples de frameworks PHP pour développer une API REST

- **Laravel :**
  - Introduction : est un framework API open-source qui permet aux programmeurs de construire leur API Rest en utilisant l'architecture MVC .
  - Lien : <https://laravel.com/>
- **Lumen :**
  - Introduction : est un framework micro-API réalisé à partir de Laravel. Ce framework a une fonction qui est en corrélation avec chacune des méthodes CRUD de base du protocole HTTP.
  - Lien : <https://www.lumen.com/en-us/home.html>
- **Guzzle :**
  - Introduction : est un framework qui fournit des méthodes simples pour envoyer des requêtes HTTP et créer des clients REST. Il possède également une interface simple, ce qui permet aux développeurs de naviguer facilement dans le framework.
  - Lien : <https://docs.guzzlephp.org/en/stable/>
- **Épiphanie :**
  - Introduction : est un framework micro-API qui convient parfaitement aux développeurs qui souhaitent un processus de création d'API simple et clair
  - Lien : <https://github.com/jmathai/epiphany>

### Consignes architecture REST

-  **Distinction Client-serveur** : Les responsabilités doivent être clairement distinguées entre le client et le serveur.
-  **Sans état** : La communication client – serveur s'effectue sans enregistrer l'état de la session de communication sur le serveur entre deux requêtes successives. Le client ne doit pas être conçu pour se souvenir des requêtes précédentes.
-  **Avec mise en cache** : Une API REST doit pouvoir mettre en cache les interactions client-serveur.
-  **Système en couches** : Un client ne fera pas la différence entre une connexion directe serveur final ou à un serveur intermédiaire. Tous les serveurs impliqués dans les demandes client peuvent être regroupé dans une hiérarchie.
-  **Interface uniforme** : Il devrait y avoir un format standard par lequel le client reçoit les réponses du serveur. Elle simplifie et découpe l'architecture, ce qui permet à chaque composant d'évoluer indépendamment.
-  **Code à la demande** : Les serveurs peuvent temporairement étendre ou modifier les fonctionnalités d'un client en lui transférant du code exécutable.

### API REST basées sur HTTP

L'architecture REST utilise les spécifications originelles du protocole HTTP, plutôt que de réinventer une surcouche (comme le font SOAP ou XML-RPC par exemple).

- **Règle n°1 :** l'URI (Uniform Resource Identifier) comme identifiant des ressources

Quelques exemples de construction d'URL avec RESTful :

- Liste des commentaires : <http://mywebsite.com/comments>
- Filtre et tri sur les commentaires : <http://mywebsite.com/comments?filter=rate2&tri=asc>
- Affichage d'un commentaire : <http://mywebsite.com/comments/203>

- **Règle n°2 :** les méthodes HTTP comme identifiant des opérations

Généralement il y a 4 opérations possibles qui correspondent au CRUD.

Exemple d'URL pour une ressource donnée :

- Créer un commentaire : POST <http://mywebsite.com/comments>
- Affichage d'un commentaire : GET <http://mywebsite.com/comments/203>
- Mettre à jour d'un commentaire : PUT <http://mywebsite.com/comments/203>
- Supprimer un commentaire : DELETE <http://mywebsite.com/comments/203>

### API REST basées sur HTTP

- **Règle n°3 :** les réponses HTTP comme représentation des ressources
  - La réponse envoyée est la représentation d'une ressource.
  - Une ressource peut avoir plusieurs représentations dans des formats divers : HTML, XML, CSV, JSON,....
  - C'est au client de définir quel format de réponse il souhaite recevoir via l'en-tête Accept.

#### Exemples:

- Réponse en HTML : **Accept: text/html**
- Réponse en XML : **Accept: application/xml**

- **Règle n°4 :** les liens comme relation entre ressources
  - Les liens d'une ressource vers une autre indiquent la présence d'une relation.
  - l'attribut **rel** doit être spécifier sur tous les liens pour expliciter cette description et indiquer la nature de la relation.
  - Types de relations de lien : <https://www.iana.org/assignments/link-relations/link-relations.xml#link-relations-1>

**Exemple :** <link rel="self" title="self" href="http://mywebsite.com/comments?q=rate2&page=1&c=5" />

### Composants REST

Composant	Exemples Web modernes
Serveur d'origine	Apache httpd Microsoft IIS
Passerelle	Squid CGI Reverse Proxy
Proxy	CERN Proxy Netscape Proxy, Gauntlet
user agent	Netscape Navigator Lynx MOMspider

Tab. : Composants REST

### Créer une API RESTful

- **Étape 1 :** Créer une table de base de données MySQL. Ensuite insérer des données.
- **Étape 2 :** Créer une connexion à la base de données.
- **Étape 3 :** Implémenter la création des items.

Exemple :

- Nous allons créer un fichier `create.php` et implémenter la fonctionnalité de création d'éléments pour insérer un nouvel enregistrement dans la base de données.
- L'API acceptera les valeurs HTTP POST pour créer un enregistrement. Nous allons créer un objet de la classe `Items.php` et appeler la méthode `create()` pour enregistrer l'enregistrement.
- Dans la classe `Items.php`, nous allons implémenter la méthode `create()` pour insérer les valeurs POST dans la table de la base de données.
- Nous allons tester l'exemple de création d'URL à partir de notre API REST en utilisant l'URL suivante : <https://www.monsite.com/demo/rest-api/items/create>

```
rest-api
config
    Database.php
Class
    Items.php
items
    create.php
    read.php
    update.php
    delete.php
    .htaccess
```

### Créer une API RESTful

- **Étape 4 :** Implémenter les éléments lus

Exemple :

- Nous allons créer un fichier read.php et implémenter la fonctionnalité de lecture des items.
- L'API acceptera les valeurs HTTP GET pour obtenir le/les élément(s) de la requête. Nous allons créer un objet de la classe Items.php et appeler la méthode read() pour renvoyer les données de réponse.
- Dans la classe Items.php, nous allons implémenter la méthode read() pour lire les valeurs GET dans la table de la base de données.
- Nous allons tester l'exemple de lecture d'URL à partir de notre API REST en utilisant l'URL suivante : <https://www.monsite.com/demo/rest-api/items/read>

- **Étape 5 :** Implémenter les éléments de mise à jour

Exemple :

- Nous allons créer un fichier update.php et implémenter la fonctionnalité de mise à jour des items.
- L'API acceptera les valeurs HTTP PUT pour modifier l'enregistrement. Nous allons créer un objet de la classe Items.php et appeler la méthode update() pour mettre à jour les détails des items.
- Dans la classe Items.php, nous allons implémenter la méthode update() pour mettre à jour les valeurs PUT dans la table de la base de données.
- Nous allons tester l'exemple de mise à jour d'URL à partir de notre API REST en utilisant l'URL suivante : <https://www.monsite.com/demo/rest-api/items/update>

### Créer une API RESTful

- **Étape 6 :** Implémenter la suppression d'éléments

Exemple :

- Nous allons créer un fichier delete.php et implémenter la fonctionnalité de suppression des items.
- L'API acceptera les valeurs HTTP DELETE pour supprimer l'enregistrement. Nous allons créer un objet de la classe Items.php et appeler la méthode delete() pour supprimer un enregistrement.
- Dans la classe Items.php, nous allons implémenter la méthode delete() pour supprimer les valeurs DELETE dans la table de la base de données.
- Nous allons tester l'exemple de suppression d'URL à partir de notre API REST en utilisant l'URL suivante : <https://www.monsite.com/demo/rest-api/items/delete>

- **Étape 7 :** Créez des URL de demande SEO conviviales pour l'API REST

Par exemple, nous allons créer un fichier HTACCESS .htaccess pour rendre les URL conviviales pour le référencement.

```
RewriteEngine On      //Règle de réécriture pour
Apache
RewriteRule ^read$ read.php [NC,L]
RewriteRule ^read/([0-9_-]*)$ read.php?id=$1
[NC,L]
RewriteRule ^create$ create.php [NC,L]
RewriteRule ^update$ update.php [NC,L]
RewriteRule ^delete$ delete.php [NC,L]
```

### Sitographie

<https://www.oracle.com/fr/cloud/definition-web-service/>

<https://www.php.net/manual/fr/book.soap.php>

[https://fr.wikipedia.org/wiki/Representational\\_state\\_transfer#Contraintes\\_architecturales](https://fr.wikipedia.org/wiki/Representational_state_transfer#Contraintes_architecturales)

<https://waytolearnx.com/2019/07/creer-et-utiliser-une-api-rest-en-php.html>

<https://blog.nicolashachet.com/developpement-php/larchitecture-rest-expliquee-en-5-regles/>

<https://www.phpzag.com/how-to-create-simple-rest-api-in-php/>