

L3 informatique

2022-2023

Programmation Web

Partie 6 : initiation à PHP

Étienne André

Université Sorbonne Paris Nord
Etienne.Andre@univ-paris13.fr



Version: 1^{er} février 2023

Outline

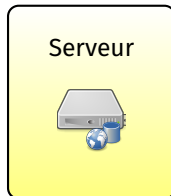
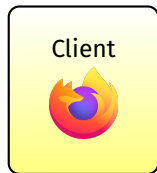
- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Motivation : Web dynamique

- **Web statique** : exclusivement HTML (et CSS)
 - Pages HTML stockées telles quelles sur le serveur
 - Rare! (à part de tous petits sites Web)
- **Problème des sites statiques** :
 - Pas de base de données
 - Pas de personnalisation (requête, profil personnalisé, date...)
- **Site Web dynamique** : génération de pages HTML par un programme (ou script) **côté serveur** (*backend*)
 - Utilisation possible d'une base de données
 - Quelques langages pour le Web dynamique côté serveur : PHP, Ruby, ASP.NET, Node.JS...

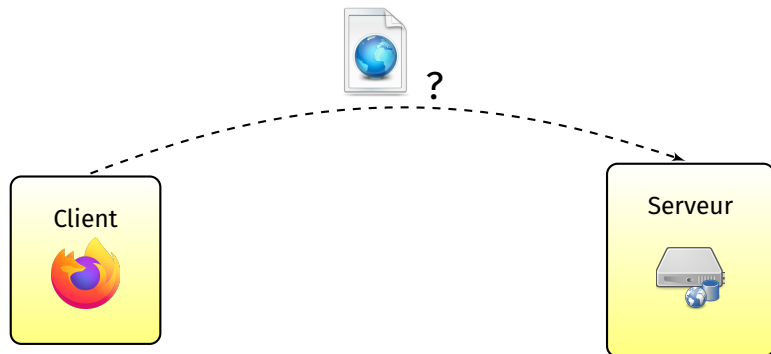
Requête client-serveur (rappel du cours « client-serveur »)

Client :



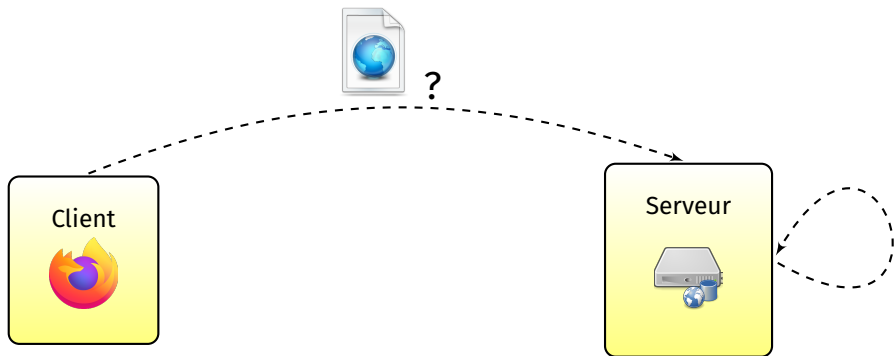
Requête client-serveur (rappel du cours « client-serveur »)

Client : « Bonjour, je voudrais `https://www.univ-paris13.fr` »



Requête client-serveur (rappel du cours « client-serveur »)

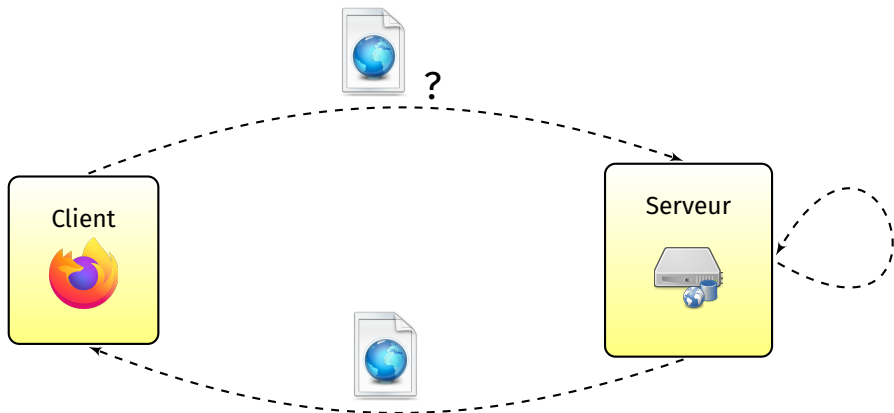
Client : « Bonjour, je voudrais `https://www.univ-paris13.fr` »
(préparation de la page Web à l'aide d'un langage tel que PHP)



Requête client-serveur (rappel du cours « client-serveur »)

Client : « Bonjour, je voudrais `https://www.univ-paris13.fr` »
(préparation de la page Web à l'aide d'un langage tel que PHP)

Serveur : « Bonjour, voici la page Web »



Introduction

- PHP = PHP : Hypertext Preprocessor



- 😊 Acronyme récursif! (ou sigle autoréférentiel ou sigle récursif)
 - Autres exemples : Bing, GNU, Visa...

Historique

- 1994 : Première version par Rasmus Lerdorf (à l'origine : une bibliothèque logicielle en C)
- 1997 : réimplémentation complète par deux étudiants (Andi Gutmans et Zeev Suraski)
- 2015 : sortie de la version 7
- 2020 : sortie de la version 8
- 2022 : utilisé sur 78 % des serveurs Web

Source : <https://w3techs.com/>

Principales caractéristiques (1/2)

- Langage **impératif**
- Langage **orienté objet** (présence de classes, interfaces et objets)
- Langage **faiblement typé**
- Langage sous **licence libre**

Principales caractéristiques (2/2)

- Fichier avec extension `.php`
 - Peut être entièrement écrit en PHP, ou en HTML avec code PHP imbriqué
- Exécution entièrement côté serveur : pas accessible depuis le client (qui ne reçoit que la page HTML résultat)
- Nécessité d'une application côté serveur (exemple : serveur Apache ou nginx)



PHP et le Web

- PHP est surtout utilisé dans un environnement de développement Web
 - Tourne sur un **serveur Web**

- PHP peut être notamment utilisé pour générer de façon dynamique
 - des pages HTML
 - des feuilles de style CSS
 - des scripts JavaScript

Outline

- 1 Introduction
- 2 Syntaxe de base**
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Outline

- 1 Introduction
- 2 Syntaxe de base
 - **Aperçu**
 - Les variables
 - Structures de contrôle
 - Les tableaux
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Bref aperçu

Syntaxe relativement inspirée du langage C

```
1 <?php
2   $films = array('Taxi Driver', 'Spring Breakers',
3     'Parasite', '12 Monkeys');
4   echo "\n" . '<ul>';
5   foreach($films as $film) {
6     echo "\n\t" . '<li>' . $film . '</li>';
7   }
8   echo "\n" . '</ul>';
9 ?>
```

...mais aussi beaucoup plus puissante et haut niveau que le C!

Balises

Code à insérer entre `<?php` et `?>`

```
1 <?php
2     $langage = 'PHP';
3     echo 'Le ' . $langage . ", c'est bien !";
4     // Affiche "Le PHP, c'est bien !"
5 ?>
```


Interactions entre PHP et HTML

☹️ Option 1 (non recommandée et dépréciée) : insertion dans le code HTML

```
1 <p>Nous sommes le <script language="php">echo date('d-m-y  
h:i:s');</script></p>
```

☹️ Option 2 (pas très recommandée) : insertion dans le code HTML

```
1 <p>Nous sommes le <?php echo date('d-m-y h:i:s');?></p>
```

😊 Option 3 (meilleure) : génération de l'intégralité du HTML

```
1 <?php  
2     $aujourd'hui = date('d-m-Y h:i:s');  
3     echo "<p>Nous sommes le" . $aujourd'hui . '</p>';  
4 ?>
```

Commentaires

Trois syntaxes pour les commentaires :

- Syntaxe dite « à la C » : bloc entre `/*` et `*/`
- Fin d'une ligne après `//`
- Fin d'une ligne après `#` (syntaxe dite à la Bash; moins utilisée?)

```
1 <?php
2     $langage = 'PHP'; // Définition d'une variable
3     /* $langage = 'OCaml';
4     echo 'Je déteste le !' . $langage . '!';
5     */
6     echo 'Le ' . $langage . ", c'est bien !";
7     # Affiche "Le PHP, c'est bien !"
8 ?>
```

😊 PHP est exécuté côté serveur : les commentaires sont invisibles depuis le client

Outline

- 1 Introduction
- 2 **Syntaxe de base**
 - Aperçu
 - **Les variables**
 - Structures de contrôle
 - Les tableaux
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Les variables

- Syntaxe des noms de variables : $\$(lettre | _)(lettre | chiffre | _)*$

```
1 // Exemples de déclarations de 3 variables différentes
2 $film = 2046;
3 $_film = 'The Assassin';
4 $Film = "Taxi Driver";
```

- Sensible à la casse ($\$film \neq \$Film$)

- Typage dynamique

```
1 // Déclarations successives
2 $film = 2046;
3 $film = 'The Assassin';
4 $film = false;
5 $film = array(2046, 'Spring Breakers', "Burning");
```

Les nombres

■ Les entiers

```
1 // Nombres entiers
2 $a = 2046; $b = 300;
3 echo 2046 - 300;
4 echo intval(2046, 300); // affiche 6 (divison entière)
```

■ Les flottants

```
1 // Nombres flottants
2 $a = 2.046; $b = 2.04e6;
3 echo $a+$b; echo $a/$b;
```

■ Nombreuses fonctions autour des nombres

⚠ Comme dans la plupart des langages, les nombres sont sujets au débordement et aux erreurs d'arrondi (`float`)

Les chaînes de caractères

Deux syntaxes principales :

■ Guillemets simples : aucun nom de variable interprété

```
1 echo 'J\'aime "Pour quelques $dollars de plus"\n';  
2 // affiche: J'aime "Pour quelques $dollars de plus"\n
```

■ Guillemets doubles : variables et retours à la ligne interprétés

```
1 $film = 2046;  
2 echo "J'aime le film \"$film\"\n";  
3 // affiche: 'J'aime le film "2046"' suivi d'un retour à  
4 la ligne  
5 echo "\t J'aime le film \"${film}\"";  
6 // affiche: ' J'aime le film "2046"'
```

En cas de doute : guillemets simples

Les variables externes (super-globales)

Super-globale	Description	Exemple
<code>\$_GLOBALS</code>	Toutes les variables globales	<code>\$_GLOBALS['ma_variable']</code>
<code>\$_POST</code>	Les variables envoyées par un formulaire (méthode <code>POST</code>)	<code>\$_POST['mot_de_passe']</code>
<code>\$_GET</code>	Les variables passées en paramètres à l'URL, ou par un formulaire (méthode <code>GET</code>)	<code>\$_GET['page_demandee']</code>
<code>\$_FILES</code>	Tableau associatif des valeurs associées aux documents téléchargés par le script courant	<code>\$_FILES['nom_fichier']['size']</code>
<code>\$_SESSION</code>	Tableau associatif des valeurs associées à la session courante	<code>\$_SESSION['login']</code>
<code>\$_COOKIE</code>	Tableau associatif des valeurs associées à un cookie	<code>\$_COOKIE['internaute']</code>
<code>\$_SERVER</code>	Informations liées au serveur utilisé	<code>\$_SERVER['SERVER_PORT']</code>

Outline

- 1 Introduction
- 2 **Syntaxe de base**
 - Aperçu
 - Les variables
 - **Structures de contrôle**
 - Les tableaux
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Structures de contrôle : if

Conditionnelle :

```
1 $film = 'Spring Breakers'; // Définition d'une variable
2 if($film == 'The Assassin'){
3     echo 'Meilleur film de la décennie 2010s';
4 }else{
5     echo 'Pas le meilleur film';
6 }
7 // Affiche 'Pas le meilleur film'
```

Structures de contrôle : boucle `while`

Boucle :

```
1 $film = 0;  
2 while($film < 2046){  
3     $film ++;  
4 }  
5 echo $film; // affiche 2046
```

Structures de contrôle : boucle `do while`

Boucle :

```
1 $film = 2046;  
2 do{  
3     echo $film;  
4 }while($film <> 2046);  
5 // affiche 2046
```

Note : le corps est toujours exécuté au moins une fois, même si la condition est initialement fausse!

Structures de contrôle : boucle `for`

Itération avec condition :

```
1 $films = array('Taxi Driver', 'Spring Breakers',  
2             'Parasite', '12 Monkeys');  
3 for($i = 0; $i < count($films); $i++) {  
4     echo $films[$i] . "\n";  
}
```

Structures de contrôle : boucle `foreach`


Itération sur un ensemble :

```
1 $films = array('Taxi Driver', 'Spring Breakers',  
2           'Parasite', '12 Monkeys');  
3 foreach($films as $film) {  
4     echo $film . "\n";  
}
```

Structures de contrôle : case

Traitement par cas :

```
1 switch ($film) {  
2     case 'Spring Breakers':  
3     case '12 Monkeys':  
4         echo 'Film américain';  
5         break;  
6     case 'Tigre et Dragon':  
7         echo 'Film taiwanais';  
8         break;  
9     default:  
10        echo 'Film inconnu';  
11 }
```

 Ne pas oublier le `break;` sinon les instructions suivantes sont exécutées

Outline

- 1 Introduction
- 2 **Syntaxe de base**
 - Aperçu
 - Les variables
 - Structures de contrôle
 - **Les tableaux**
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Les tableaux en PHP

- À mi-chemin entre deux concepts de programmation :
 - un **tableau** (valeurs avec indices croissants), et
 - une **table associative** (ensemble de couples (clé, valeur))

- Caractéristiques
 - Pas de taille prédéfinie
 - Pas de type : peuvent contenir (presque) n'importe quoi

- Syntaxe
 - Syntaxe de déclaration : `array()`
 - Association d'une valeur à une clé : `clé => valeur`

Les tableaux en PHP : exemple

```
1 // Ensemble de valeurs
2 $realisatrices = array('Ducournau', 'Despentes', 'Sciamma');
3 echo $realisatrices[1]; // Affiche 'Despentes'
4
5 // Clés => valeurs
6 $films = array(
7     'Petite Maman' => array('real' => 'Sciamma',
8         'sortie' => 2021,
9         'interpretation' => array('Sanz', 'Meurisse')
10 ),
11     'Jackie Brown' => array('real' => 'Tarantino',
12         'sortie' => 1997,
13         'interpretation' => array('Grier', 'Jackson', 'De
14     Niro')
15 ),
16     'Lalaland'      => 'aucune info'
17 );
18 echo $films['Jackie Brown']['interpretation'][2]; //
19     Affiche 'De Niro'
```

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions**
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

Les fonctions

- Fonctions définies par le langage (*built-in*) : plus de 1000!
- Fonctions définies dans le programme : définies par le mot-clé `function`

```
1 function affiche_film($titre, $annee) {  
2     echo "\n<p>$titre a été réalisé en $annee.</p>";  
3 }  
4  
5 affiche_film('Taxi Driver',      1976);  
6 affiche_film('Tigre et Dragon',  2000);  
7 affiche_film('Dunkerque',        2017);  
8 // <p>Taxi Driver a été réalisé en 1976.</p>  
9 // <p>Tigre et Dragon a été réalisé en 2000.</p>  
10 // <p>Dunkerque a été réalisé en 2017.</p>
```

Mode strict

Avec PHP 7, un peu plus de typage a été introduit avec le mode `strict`

■ Déclaration

```
1 <?php
2 // Première instruction du fichier
3 declare(strict_types=1);
4 ?>
```

■ Conséquence : typage optionnel des fonctions (mais doit alors être valide!)

■ Sinon : exception (Fatal error: Uncaught TypeError)

```
1 function somme(int $a, int $b) : int{
2     return $a + $b;
3 }
4 function affiche_film(string $titre, int $annee) : void{
5     echo "\n<p>$titre a été réalisé en $annee.</p>";
6 }
```

Valeurs des arguments par défaut

Si un argument n'est pas spécifié, la **valeur par défaut** est utilisée

Exemple :

```
1 function affiche_film(string $titre, int $annee = 2023) {
2     echo "\n<p>$titre a été réalisé en $annee.</p>";
3 }
4 affiche_film('Taxi Driver', 1976);
5 affiche_film('Tigre et Dragon', 2000);
6 affiche_film('Dunkerque');
7 // <p>Taxi Driver a été réalisé en 1976.</p>
8 // <p>Tigre et Dragon a été réalisé en 1976.</p>
9 // <p>Dunkerque a été réalisé en 2023.</p>
```

Passage par référence

Il est possible de passer un argument **par référence** et non par valeur : sa valeur est donc modifiée, y compris à l'extérieur de la fonction

```
1 // Passage par valeur
2 function increment(int $n){
3     $n++;
4 }
5 // Passage par référence (syntaxe &)
6 function increment2(int &$n){
7     $n++;
8 }
9 $a = 12; $b = 2045;
10 increment($a);
11 echo $a; // affiche 12
12
13 increment2($b);
14 echo $b; // affiche 2046
```

Portée des variables dans les fonctions

- Variable **locale** : variable définie dans une fonction
- Si une variable locale possède le même nom qu'une variable globale, la variable locale est prioritaire
 - La variable globale peut être accédée par le tableau de variables super-globales prédéfini `$GLOBAL['nom_variable']` ; ou en déclarant la variable avec le mot-clé `global`

```
1 $a = 300;  
2 $b = 2010;  
3 $c = 2045;  
4 function test_var_globales(){  
5     global $b;  
6     echo $a; // n'affiche rien !  
7     $b += 2;  
8     $GLOBALS['c'] ++;  
9 }  
10 test_var_globales();  
11 echo $b; // affiche 2012  
12 echo $c; // affiche 2046
```

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet**
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données

PHP et la programmation orientée objet

Quelques fonctionnalités orientées objet en PHP :

- Classes
- Interfaces
- Classes anonymes
- Objets
- Héritage (simple)
- Méthodes statiques
- Visibilité différenciée (`public`, `protected`, `private`)

Mini-initiation à l'objet en PHP via un exemple

```
1 class Film{
2     protected $titre = ''; // accessible dans les sous-classes
3     function __construct($titre) {
4         $this->titre = $titre;
5     }
6     public function affiche(){
7         echo 'Le film ' . $this->titre . ' est bien';
8     }
9 }
10
11 class Chefdoeuvre extends Film{ // heritage
12     public function affiche(){ // surcharge
13         echo 'Le film ' . $this->titre . ' est génial';
14     }
15 }
16 $film1 = new Film('Kill Bill');
17 $film2 = new Chefdoeuvre('Jackie Brown');
18 $film1->affiche(); // 'Le film Kill Bill est bien'
19 $film2->affiche(); // 'Le film Jackie Brown est génial'
```

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web**
- 6 Interactions avec les bases de données

Génération de code HTML

PHP est généralement utilisé pour générer tout ou partie du code HTML

- et/ou CSS, et/ou JavaScript

Exemple de code PHP :

```
1 $films = array('Taxi Driver', 'Spring Breakers',  
2 'Parasite', '12 Monkeys');  
3 echo "\n" . '<ul>';  
4 foreach($films as $film) {  
5     echo "\n\t" . '<li>' . $film . '</li>';  
6 }  
7 echo "\n" . '</ul>';
```

Génère le code HTML suivant :

```
1 <ul >  
2     <li>Taxi Driver</li>  
3     <li>Spring Breakers</li>  
4     <li>Parasite</li>  
5     <li>12 Monkeys</li>  
6 </ul >
```

Génération de code HTML : attention aux espaces

Ne pas oublier qu'il y a trois niveaux :

- le code PHP
- le code HTML généré par PHP
- et le rendu de la page Web spécifié par le code HTML

Code PHP :

```
1 echo "\n\t" . '<p>Un';  
2 echo ' exemple' . "\n" . ' d\'espacement</p><p>bizarre' .  
   "\n" . '</p>';
```

Code HTML généré

```
1 <p>Un exemple  
2 d'espacement</p><p>bizarre  
3 </p>
```

Rendu sur le navigateur :

```
Un exemple  
bizarre
```

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web**
 - **Requêtes GET**
 - Requêtes POST
 - Cookies
 - Sessions
- 6 Interactions avec les bases de données

La méthode de requête HTTP « GET »

Permet notamment de :

- 1 Récupérer les valeurs passées en paramètre à l'URL
 - Exemple : `https://www.startpage.com/sp/search?query=get+php&cat=web&pl=opensearch&language=français`
 - Massivement utilisé aujourd'hui : regardez les URL des sites que vous visitez!
- 2 Récupérer les informations transmises par un formulaire via la **méthode** `get`
 - `method="get"`

Get et les paramètres de l'URL

- Paramètres de l'URL : ensemble de couples (clé, valeur)

- **Syntaxe** : `https://www.monsite.fr?cle1=valeur1&cle2=valeur2&cle3=valeur3&...`

- Exemple :

`https://www.startpage.com/sp/search?query=get+php&cat=web&pl=opensearch&language=français`

- (query, get+php)
- (cat, web)
- (pl, opensearch)
- (language, français)

- Attention aux caractères spéciaux

- Certains caractères (« / », « & », espaces...) sont proscrits dans les clés et les valeurs
- Utiliser des fonctions de **conversion automatique** (`urlencode`)

Get et PHP

Récupération des arguments : variable PHP prédéfinie `$_GET`

Exemple :

- URL : `https://www.monsite.fr/index.php?film=Titane&annee=2021`
- Code PHP et affichage :

```
1 echo '<p>Le film ' . $_GET['film'] . ' a été réalisé en ' .  
   $_GET['annee'] . '&nbsp;!</p>';  
2 // Si l'on a cliqué sur l'URL ci-dessus, va afficher 'Le  
   film Titane a été réalisé en 2021 !'
```

Get et PHP : caractères spéciaux

- Pour **encoder** une URL (en échappant les espaces, notamment), utiliser `urlencode`
- Les valeurs récupérées dans `$_GET` sont **déjà décodées** (pas besoin de convertir les caractères spéciaux!)

Get et PHP : caractères spéciaux

- Pour **encoder** une URL (en échappant les espaces, notamment), utiliser `urlencode`
- Les valeurs récupérées dans `$_GET` sont **déjà décodées** (pas besoin de convertir les caractères spéciaux!)

Exemple d'encodage d'URL :

```
1 echo '<a href="' . $_SERVER['PHP_SELF'] . '?cours=' .  
    urlencode('Prog Web') . '&prof=' . urlencode('Étienne  
    André') . '">Cliquer ici</a>';
```

Va afficher un lien vers

`https://www.monsite.fr/index.php?cours=Prog+Web&prof=%C3%89tienne+Andr%C3%A9`

Exemple de récupération des paramètres de l'URL :

```
1 echo '<p>Bonjour ' . $_GET['prof'] . ', prof de ' .  
    $_GET['cours'] . '&nbsp;!'</p>';  
2 // Si l'on a cliqué sur l'URL ci-dessus, va afficher  
    'Bonjour Étienne André, prof de Prog Web !'
```

Formulaires (méthode get) et PHP : petit exemple (1/2)

Formulaire HTML

```
1 <form action="<?php echo $_SERVER['PHP_SELF'];?>"
  method="get">
2   <p><label for="idfilm">Film</label>
3     <input type="text" name="film" id="idfilm"
  placeholder="Votre meilleur film"></p>
4 </form>
```

Récupération et traitement des valeurs (PHP)

```
1 if ($_SERVER["REQUEST_METHOD"] == "GET") {
2   // Récupération des données
3   $film = $_GET['film']; // 'film' = 'name' du form
4   if (empty($film)) {
5     echo "<strong>Erreur</strong> : pas de film";
6   } else {
7     echo $film;
8   }
9 }
```

...ces deux codes peuvent être sur la même page (même URL)!

Formulaires (méthode get) et PHP : petit exemple (2/2)

La conversion est automatique à l'encodage et au décodage

Exemple :

- Si l'on entre dans le champ `nom` :

« Le /département/ R&T ?! »

- Alors l'URL générée est :

```
https://www.monsite.fr/index.php?nom=Le+%2Fd%C3%A9partement%2F+R%26T+%3F%21
```

- Et, après récupération via PHP, le texte affiché en HTML est :

« Le /département/ R&T ?! »

Formulaires et PHP : discussion

La **même page** peut être utilisée pour afficher le formulaire et le traitement

- Possibilité de ne pas réafficher le formulaire une fois que les données ont été envoyées et traitées
- En cas d'erreur et de rechargement de la page avec le même formulaire, les informations entrées sont par défaut perdues
 - possibilité de réafficher les informations envoyées (pour pré-remplir le formulaire) grâce aux **sessions** (voir plus loin)

Complémentaire à **JavaScript**

- JavaScript peut effectuer des vérifications avant l'envoi du formulaire
- Attention néanmoins : JavaScript étant exécuté côté client, un client néfaste peut décider de désactiver voire modifier les scripts JavaScript ⇒ **ne pas baser toutes les vérifications sur JavaScript!**

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web**
 - Requêtes GET
 - Requêtes POST**
 - Cookies
 - Sessions
- 6 Interactions avec les bases de données

La méthode de requête HTTP « POST »

- PHP peut récupérer et traiter (notamment) les valeurs associées à un formulaire ayant utilisé la méthode `post`
 - Via la variable PHP prédéfinie `$_POST`

Formulaires (méthode post) et PHP : petit exemple

Exemple (HTML)

```
1 <form action="<?php echo $_SERVER['PHP_SELF'];?>"
  method="post">
2   <p><label for="idfilm">Film</label>
3     <input type="text" name="film" id="idfilm"
      placeholder="Votre film préféré"></p>
4 </form>
```

Récupération et traitement des valeurs (PHP)

```
1 if ($_SERVER["REQUEST_METHOD"] == "POST") {
2   // Récupération des données
3   $film = $_POST['film']; // 'film' = 'name' du form
4   if (empty($film)) {
5     echo "<strong>Erreur</strong> : pas de film";
6   } else {
7     echo $film;
8   }
9 }
```

...ces deux codes peuvent être sur la même page (même URL)!

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web**
 - Requêtes GET
 - Requêtes POST
 - Cookies**
 - Sessions
- 6 Interactions avec les bases de données

Cookies

Cookie : format texte constitué de paires (clé, valeur) échangé avec le serveur, et stocké côté client

- peut stocker des informations telles que des préférences de l'internaute, ou son identifiant (permettant le suivi par le site Web)
- peut avoir une durée de vie de **plusieurs années!**
- peut être supprimé aisément (via le navigateur)
- utilisé (surtout) pour pister les internautes sur un même site, voire sur plusieurs sites différents



Des bons cookies (pas comme les autres)

Les cookies en PHP

Création d'un cookie :

```
1 <?php
2     if(!isset($_COOKIE['id_internaute'])) {
3         setcookie('id_internaute', '12345');
4         // 12345 est généralement un identifiant unique
5     }
6 ?>
```

Accès à un cookie :

```
1 <?php
2     if(isset($_COOKIE['id_internaute'])) {
3         echo 'Identifiant de session : ' .
4             $_COOKIE['id_internaute'];
5     }
6 ?>
```

permet de retrouver l'internaute même des années après! (depuis le même navigateur)

Durée de vie d'un cookie

Syntaxe : `setcookie(cle, valeur, date_expiration)`

```
1 setcookie('id', '12345', time()+365*24*3600); // un an  
2 setcookie('pref', 'site_mobile', time()+3600); // une heure
```

Supprimer un cookie : en indiquant une date passée

```
1 setcookie('id', '', time() - 3600); // date passée !
```

Que stocker dans un cookie ?

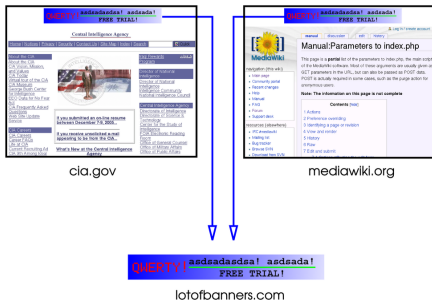
- identifiant de l'internaute (permet de rester connecté-e)
- préférence de navigation
- contenu du panier sur un site de commerce (de plus en plus remplacé par une base de données à partir de l'identifiant de l'internaute)
 - en réalité : origine des cookies!
- etc.

Danger des cookies

- Identification de visites répétées à un site même sans être identifié·e
 - le site garde en mémoire toutes les pages visitées par ce même identifiant « anonyme »
- Croisement avec votre véritable identité si l'on s'identifie une seule fois au site
 - le site ajoute à votre historique de navigation (stocké dans sa base de données) l'intégralité des visites qu'il a tracées via votre identifiant anonyme : fusion de l'historique de votre identifiant anonyme avec l'historique de votre identifiant de connexion
- Menace sur la vie privée... et bien plus encore en cas de cookies tierce partie

Les cookies tierce partie

Cookie tierce partie : déposé ou lu par un **autre site** que celui sur lequel l'Internaute est connecté·e



Exemples d'utilisations massives :

- Google Analytics
- Publicités Google (ou autres)
- Bouton « Like » : Facebook peut ainsi vous suivre sur (presque) tout Internet sans que vous ne vous rendez jamais sur le site de Facebook!

Les cookies tierce partie : résistances

- De nombreux navigateurs acceptent les cookies tierce partie par défaut, mais ils peuvent être désactivés via les préférences du navigateur
- Google a cessé d'utiliser les cookies tierce partie en 2022
 - ...mais il existe de nombreuses autres façons d'identifier un·e Internaute de façon (quasi) unique

Danger des cookies et navigation privée

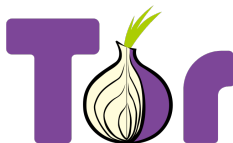
La **navigation privée** permet (notamment) de supprimer les cookies à la fin d'une navigation

- Le navigateur supprime tout (cookies, historique...) en fin de navigation
- N'empêche pas entièrement le traçage par les sites Web pour autant!

⚠ Limites de la navigation privée

La navigation privée **n'empêche pas du tout** votre fournisseur d'accès (et donc toute entité étatique) de connaître les sites que vous visitez!

Seule l'utilisation d'un réseau tel que **TOR** permet (a priori) une navigation véritablement privée.



Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web**
 - Requêtes GET
 - Requêtes POST
 - Cookies
 - Sessions**
- 6 Interactions avec les bases de données

Sessions

Session en PHP :

- période de temps où un serveur est en communication authentifiée avec un·e internaute
- une session ne concerne qu'un·e seul·e internaute à la fois
- durée variable, typiquement **une heure**
- contient habituellement des informations liées au compte d'un·e internaute
- permet de faire **persister** des informations entre plusieurs pages (du même serveur)

Pas d'infos confidentielles

Les informations confidentielles (par exemple authentification par mot de passe, numéro de carte bancaire...) **ne doivent pas être conservées** au sein d'une session

Utilité des sessions

- Mémoriser temporairement des informations de l'internaute
- Pré-remplir un formulaire avec les informations précédemment envoyées (par exemple : en cas d'erreur de saisie et réaffichage du même formulaire)
- Utilisation en conjonction avec les cookies

Sessions : exemple simple

```
1 <?php
2 // Nécessaire pour utiliser les sessions
3 session_start();
4
5 // Création d'un compteur de visites (de la MÊME personne)
6 if (!isset($_SESSION['nb_visites'])) {
7     $_SESSION['nb_visites'] = 0;
8 } else {
9     $_SESSION['nb_visites']++;
10 }
11 echo $_SESSION['nb_visites'];
12 ?>
```

Comportement :

- le compteur va s'incrémenter à chaque rechargement de la page
- une autre personne (ou soi-même sur un différent navigateur, ou le même navigateur en navigation privée) aura sa propre valeur du compteur!

Sessions : quelques fonctions utiles

- Activation des sessions (avant toute génération de code HTML)

```
1 session_start ();
```

- Suppression de toutes les variables de session

```
1 session_unset ();
```

- Fermeture de la session en cours

```
1 session_destroy ();
```

Destruction de session après une durée donnée

```
1 session_start();
2 if (isset($_SESSION['debut']) && (time() -
   $_SESSION['debut'] > 1800)) {
3     session_unset();
4     session_destroy();
5     echo "session terminée";
6 }
7 $_SESSION['debut'] = time();
```

Comportement :

- la session est supprimée après 30 minutes sans rechargement
- ...mais elle est sinon automatiquement prolongée de 30 minutes à chaque rechargement

Destruction de session après une durée donnée

```
1 session_start();
2 if (isset($_SESSION['debut']) && (time() -
   $_SESSION['debut'] > 1800)) {
3     session_unset();
4     session_destroy();
5     echo "session terminée";
6 }
7 $_SESSION['debut'] = time();
```

Comportement :

- la session est supprimée après 30 minutes sans rechargement
- ...mais elle est sinon automatiquement prolongée de 30 minutes à chaque rechargement

Syntaxe alternative :

```
1 session_regenerate_id(true);
```

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données**

Motivation

- PHP permet de générer des pages Web dynamiques... mais ne permet pas (particulièrement) de stocker de l'information
- Solution : **base de données**
 - Base de recherche des moteurs de recherche
 - Posts et commentaires sur les réseaux sociaux
 - Stockage d'identifiants personnels
 - Stockage de l'historique de navigation d'un·e internaute par une régie publicitaire ou un réseau social
 - Courriels
 - etc.

MySQL

- Système de gestion de bases de données relationnelles
- Première version : 1995
- Distribué sous licence libre (et propriétaire)
- Très efficace en lecture (un peu moins en écriture)
- Concurrents de MySQL : Microsoft SQL server, Oracle, PostgreSQL



MySQL et PHP

- Couple PHP-MySQL très fréquent (mais pas exclusif!)
- Utilisé notamment au sein de l'ensemble LAMP
 - Linux, Apache, MySQL, PHP (ou Perl/Python)
 - Variantes / concurrents : LAMA, MAMP, WAMP...
- PHP offre un ensemble de fonctions en lien avec MySQL

Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données**
 - **Fonctions PHP pour MySQL**
 - Éviter les injections SQL

Connexion à une base de données MySQL

Syntaxe orientée objet (plus récente)

```
1 $mysqli = new mysqli('serveur', 'user', 'mdp', 'base');
2 // NOTE: peut aussi prendre le port et le socket en
   arguments optionnels
3 if ($mysqli -> connect_error) {
4     /* Gestion de l'erreur */
5 }
```

Syntaxe procédurale (plus courante?)

```
1 $connexion = mysqli_connect('serveur', 'user', 'mdp',
   'base');
2 if (!$connexion) {
3     /* Gestion de l'erreur */
4 }
```

Gestion des erreurs de connexion

Ne pas afficher les erreurs par défaut **mais** les gérer depuis PHP :

```
1 /* Désactiver les rapports d'erreur */
2 mysqli_report(MYSQLI_REPORT_OFF);
3
4 /* Le '@' permet de supprimer les warnings */
5 $mysqli = @new mysqli('serveur', 'user', 'mdp', 'base');
6
7 if ($mysqli->connect_error) {
8     /* Gestion de l'erreur */
9     echo ('Erreur de connexion : ' .
10         $mysqli->connect_error);
11     exit(1);
12 }
```


Requête MySQL en PHP

Syntaxe orientée objet :

```
1 /* Requête pour récupérer 5 noms de films (table 'Films') */  
2 $resultat = $mysqli->query("SELECT nom FROM Films LIMIT 5");  
3 printf("%d résultats retournés.\n", $resultat->num_rows);
```

Syntaxe procédurale

```
1 $resultat = mysqli_query($connexion, "SELECT nom FROM Films  
    LIMIT 5");  
2 printf("%d résultats retournés.\n",  
    mysqli_num_rows($resultat));
```

Exemple complet PHP et MySQL (orienté objet)

```
1 $mysqli = new mysqli('serveur', 'user', 'mdp', 'base');
2 if ($mysqli -> connect_error) {
3     /* ... Gestion de l'erreur ... */
4 }
5
6 $resultat = $mysqli->query("SELECT nom, annee FROM Films
7     LIMIT 5");
8 printf("%d résultats retournés.\n", $resultat->num_rows);
9
10 while ($ligne = $resultat->fetch_assoc()) {
11     echo $ligne['nom'] . ' a été réalisé en ' .
12         $ligne['annee'];
13 }
14
15 $resultat -> free_result();
16
17 $mysqli -> close();
```

Exemple complet PHP et MySQL (procédural)

```
1 $connexion = mysqli_connect('serveur', 'user', 'mdp',  
    'base');  
2 if (!$connexion) {  
3     /* ... Gestion de l'erreur ... */  
4 }  
5  
6 $resultat = mysqli_query($connexion, "SELECT nom, annee  
    FROM Films LIMIT 5");  
7 printf("%d résultats retournés.\n",  
    mysqli_num_rows($resultat));  
8  
9 while ($ligne = mysqli_fetch_array($result)) {  
10     echo $ligne['nom'] . ' a été réalisé en ' .  
    $ligne['annee'];  
11 }  
12  
13 mysqli_free_result($resultat);  
14  
15 mysqli_close($connexion);
```

Codages de caractères

Attention, il faut définir :

- un codage en PHP
- un codage en MySQL
- un codage en HTML

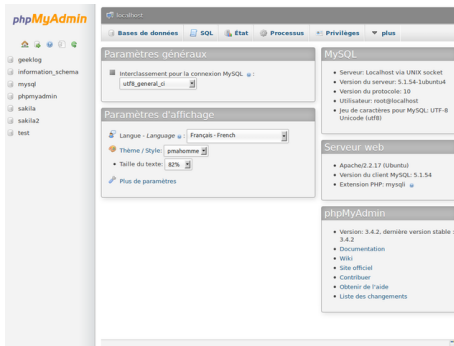
Le mieux : tout en **UTF-8**!

Fonctions utiles en PHP :

```
1 /* Pour MySQL: définir la base en 'utf8_unicode_ci' */
2
3 /* Puis préciser le codage lors de la connexion */
4 $connexion->set_charset("utf8");
5
6 /* Pour renvoyer un fichier (ex : HTML) au bon format */
7 header('Content-type: text/html; charset=utf-8');
```

phpMyAdmin

- Application Web (en PHP) pour la **gestion de bases MySQL**
- Permet des requêtes (création de tables, export, accès...) de façon intuitive et graphique
- Logiciel distribué sous **licence GNU GPL**
- Souvent installé par défaut sur les serveurs et hébergeurs (par exemple dans le cadre d'une solution LAMP)



Outline

- 1 Introduction
- 2 Syntaxe de base
- 3 Les fonctions
- 4 PHP et orientation objet
- 5 Interactions avec le Web
- 6 Interactions avec les bases de données**
 - Fonctions PHP pour MySQL
 - Éviter les injections SQL**

Attention aux injections SQL!

Attaque par injection SQL

Toute interaction entre PHP et MySQL peut présenter un risque d'attaque par injection SQL.

Idée : l'attaquant-e va chercher à insérer du code dans un formulaire, qui sera ensuite utilisé pour effectuer des requêtes néfastes (obtention de droits admin, récupération d'informations confidentielles, destructions...).

Injection SQL par commentaire

Code PHP :

```
1 $resultat = $mysqli->query("SELECT nom, annee FROM Films  
WHERE realisateur = '" . $_POST['realisateur'] . "'");
```

- 😊 Si l'attaquant·e entre « Tarantino » dans le formulaire :
 - La requête SQL est « SELECT nom, annee FROM Films WHERE realisateur = 'Tarantino' »

Injection SQL par commentaire

Code PHP :

```
$resultat = $mysqli->query("SELECT nom, annee FROM Films  
WHERE realisateur = '" . $_POST['realisateur'] . "'");
```

- 😊 Si l'attaquant·e entre « Tarantino » dans le formulaire :
 - La requête SQL est « SELECT nom, annee FROM Films WHERE realisateur = 'Tarantino' »
- ☹️ Si l'attaquant·e entre « Tarantino'; drop table Films; - - » dans le formulaire :
 - La requête SQL devient « SELECT nom, annee FROM Films WHERE realisateur = 'Tarantino'; drop table Films; - -' »
 - Rappel : la syntaxe « - - » en MySQL correspond aux commentaires : le reste est ignoré

Injection SQL par disjonction (OR)

Code PHP :

```
$resultat = $mysqli->query("SELECT id FROM users WHERE nom = '" . $_POST['nom'] . "' AND mdp = '" . $_POST['mdp'] . "'");
```

- 😊 Si l'attaquant·e entre « anissa » et « Mdp?1234@ » dans le formulaire :
- La requête SQL est « SELECT id FROM users WHERE nom = 'anissa' AND mdp = 'Mdp?1234@' »
 - Va fonctionner uniquement si « Mdp?1234@ » est le bon mot de passe

Injection SQL par disjonction (OR)

Code PHP :

```
$resultat = $mysqli->query("SELECT id FROM users WHERE nom = '" . $_POST['nom'] . "' AND mdp = '" . $_POST['mdp'] . "'");
```

- 😊 Si l'attaquant·e entre « anissa » et « Mdp?1234@ » dans le formulaire :
 - La requête SQL est « SELECT id FROM users WHERE nom = 'anissa' AND mdp = 'Mdp?1234@' »
 - Va fonctionner uniquement si « Mdp?1234@ » est le bon mot de passe
- 😞 Si l'attaquant·e entre « admin » et « hihi' OR 1; - - » dans le formulaire :
 - La requête SQL est « SELECT id FROM users WHERE nom = 'admin' AND mdp = 'hihi' OR 1; - -' »
 - À cause du « OR 1 », la 2^e partie fonctionne toujours : l'attaquant·e obtient les droits de l'utilisatrice sans connaître son mot de passe!

Éviter les injections SQL en PHP

Empêcher les injections SQL grâce aux fonctions PHP

(Début de) solution : utiliser la fonction `mysqli_real_escape_string` (ou `mysqli::real_escape_string` en syntaxe orientée objet)

```
1 /* Version orientée objet */
2 $requete = sprintf("SELECT id FROM users WHERE nom = '%s'
3     AND mdp = '%s'",
4     $mysqli->real_escape_string($_POST['nom']),
5     $mysqli->real_escape_string($_POST['mdp'])
6 );
7 $resultat = $mysqli->query($requete);
8
9 /* Version procédurale */
10 $requete = sprintf("SELECT id FROM users WHERE nom = '%s'
11     AND mdp = '%s'",
12     mysqli_real_escape_string($connexion, $_POST['nom']),
13     mysqli_real_escape_string($connexion, $_POST['mdp'])
14 );
15 $resultat = mysqli_query($connexion, $requete);
```

Les injections SQL vues par XKCD



Source : <https://xkcd.com/327/>

Alternative aux fonctions PHP vues ici : PDO

PDO = *PHP Data Objects*

- extension de PHP
- interface d'accès aux bases de données depuis PHP
- trois classes : `PDO`, `PDOStatement`, `PDOException`
- gère la connexion à la base de données, l'envoi des requêtes, la déconnexion
- permet de changer plus facilement de système de gestion de bases de données

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

Sources et références

Sources et références

Références

- **Robin Nixon** : Développer un site web en PHP, MySQL JavaScript jQuery, CSS3 et HTML5 : Un guide étape par étape pour créer des sites Web dynamiques (O'Reilly)


Sources

- Ce cours a bénéficié d'améliorations tirées des supports de cours de Thamer MECHARNIA (Université Sorbonne Paris Nord)


Crédits

Source des images utilisées I



Titre : warning sign
Auteur : ?
Source : https://commons.wikimedia.org/wiki/File:Warning_sign.png
Licence : 




Titre : Check Mark CSS Green
Auteur : Pigeon43
Source : https://commons.wikimedia.org/wiki/File:Check_Mark_CSS_Green.svg
Licence : 



Titre : The logo of Firefox
Auteur : Mozilla Corporation
Source : https://commons.wikimedia.org/wiki/File:Firefox_logo,_2019.svg
Licence : **moz://a**




Titre : Server-web-database
Auteur : RRZEicons
Source : <https://commons.wikimedia.org/wiki/File:Server-web-database.svg>
Licence : 




Titre : Vista Icons Toolbar
Auteur : VistaICO.com

Source des images utilisées II

Source : https://commons.wikimedia.org/wiki/File:1328101978_Web-page.png
Licence : 



Titre : Official PHP Logo
Auteur : Colin Viebrock
Source : <https://commons.wikimedia.org/wiki/File:PHP-logo.svg>
Licence : 




Titre : Apache HTTP server logo (2019)
Auteur : The Apache Software Foundation
Source : [https://commons.wikimedia.org/w/index.php?title=File:Apache_HTTP_server_logo_\(2019-2023\).svg](https://commons.wikimedia.org/w/index.php?title=File:Apache_HTTP_server_logo_(2019-2023).svg)
Licence : Licence Apache




Titre : Nginx logo
Auteur : Igor Sysoev
Source : https://commons.wikimedia.org/w/index.php?title=File:Nginx_logo.svg
Licence : 




Titre : Stack of choc-chip cookies
Auteur : Lisa Fotios
Source : https://commons.wikimedia.org/wiki/File:Cookie_stack.jpg
Licence : 

Source des images utilisées III




Titre : Third party HTTP cookies
Auteur : Tizio
Source : https://commons.wikimedia.org/wiki/File:Third_party_cookie.png
Licence : 




Titre : Tor logo.
Auteur : The Tor Project, Inc.
Source : <https://commons.wikimedia.org/wiki/File:Tor-logo-2011-flat.svg>
Licence : 



Titre : logo de MySQL
Auteur : en :MySQL.svg
Source : <https://fr.wikipedia.org/w/index.php?title=Fichier:MySQL.svg>
Licence : 



Titre : Screenshot of phpMyAdmin 3.4.2
Auteur : Roccivic
Source : <https://commons.wikimedia.org/wiki/File:PhpMyAdmin-main-fr.png>
Licence : 



Titre : Exploits of a mom
Auteur : XKCD
Source : <https://xkcd.com/327/>

Source des images utilisées IV

Licence : ©

Licence de ce document

Ce support de cours peut être réutilisé, modifié et republié selon les termes de la licence Creative Commons **Attribution-NonCommercial-ShareAlike 4.0 Unported (CC BY-NC-SA 4.0)**



<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Auteur : Étienne André

(Source \LaTeX disponible aux enseignant·e·s sur demande)

UNIVERSITÉ
SORBONNE
PARIS NORD

Institut
Galilée

Version : 1^{er} février 2023