PHP

Olivier Aubert

Introduction

- PHP = Hypertext PreProcessor
- Site officiel : `http://www.php.net`
- Créé en 1994 par Rasmus Lerdorf (Personal Home Page Tool)
- 1995 : ajout de la gestion des formulaires
- Première version : ensemble de scripts perl
- PHP3 : interprété
- PHP4 : compilé

Principe

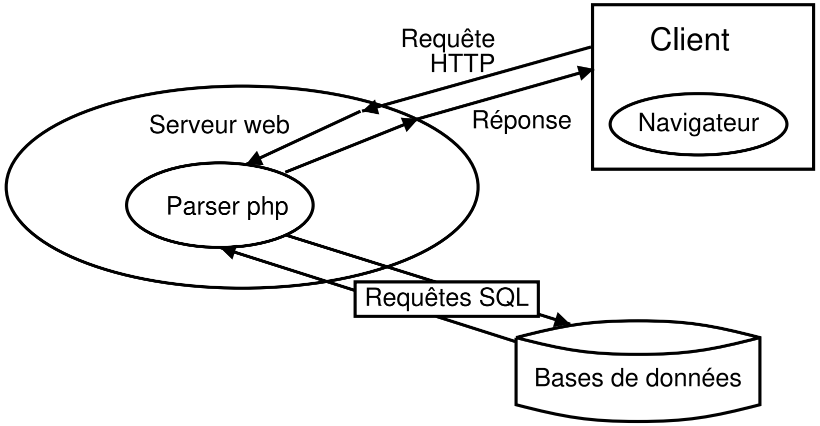
- PHP est un langage simple
- Syntaxe : inspirée de perl
- Fonctionnalités offertes :
  - Accès à des bases de données
  - Manipulation d'images
  - Accès à divers protocoles (SMTP, etc)
  - Programmation réseau
  - ...
- Mélange de code PHP et de code HTML
- Fonctionne sous UNIX, Windows, ...

Caractéristiques

- Module intégré à Apache
- Le code PHP est exécuté sur le serveur
- Dans un fichier PHP, on trouve plusieurs natures de code :
  - code PHP
  - code HTML
  - code Javascript
- Accès aux bases de données avec une limitation sur le nombre de requêtes simultanées.

Principe (II)

- Le navigateur ne voit jamais le code PHP, il reçoit uniquement le code HTML généré.



Le code PHP

Plusieurs manières d'activer du code PHP

- `<?php echo "Hello world"; ?>` (Format préféré)
- `<? echo "Hello world"; ?>` (selon configuration du serveur)
- `<script language="php"> echo "Hello world" </script>`
- `<% echo "Hello world" %>` (format ASP selon configuration du serveur)
  `<%= $variable %>` (raccourci pour echo)

PHP et shell

- Il est possible d'exécuter un script PHP dans une fenêtre de terminal
- Syntaxe : `php -q fichier.php3`
- L'option `-q` évite l'affichage du type du résultat
  `Content-type : text/html`

La syntaxe

- Les instructions sont terminées par `;`
- Typage des données faible et dynamique
- Commentaires à la C, C++ ou Shell :

```
/* Commentaire (sans imbrication) */
// Commentaire
# Commentaire
```

Les constantes

- Constantes prédéfinies :
  - `__FILE__` Fin de fichier
  - `__LINE__` Fin de ligne
  - `TRUE`, `FALSE`
- Définition par l'utilisateur avec la fonction `define`

```
define ("FICHIER", "script.log");
```


Les chaînes de caractères

- Une chaîne de caractères est délimitée par `"`, `'` ou par la syntaxe *here-document*
- Les séquences d'échappement standard sont `\n \r \t \\ \$ \" \'`
- `"` ⇒ expansion des variables
- `'` ⇒ pas d'expansion des variables et séquences d'échappement désactivées
- Syntaxe *here-document*

```
$str = <<EOS
Chaine de caracteres
sur plusieurs lignes
EOS
```

- Opérateur de concaténation : le point (`.`)

Les opérateurs

- Arithmétiques : `+ - * / % ++ -`
- Affectation : `= += -= *= /= %=`
- Comparaison : `== != < <= => >`
- Comparaison référence (php4) : `=== !==`
- Si-alors-sinon : `? :`
- Logiques : `and or xor not && ||`
- Concaténation : `.=`

Les structures de contrôle

- `if`, `else`, `elseif`
- `for`

```
for ($n = 1; $n <= 10; $n++) { print $n; }
```

- `while`
- `do ... while`
- `foreach` (php4)

```
foreach ($arr as $value) { echo "Valeur: $value"; }
foreach ($arr as $key => $value)
        { echo "Cl: $key; Valeur: $value"; }
```

- `break`, `continue`
- `switch`, `case`, `default`

Les tableaux

- Indexés numériquement...
- `$t[0] = 1; $t[2] = "toto";`

```
$t = array ("un", "deux", "trois");
```

- ... ou par des chaînes de caractères

```
$personne = array ("nom" => "Aubert",
                   "prenom" => "Olivier");
$personne["nom"] = "Aubert";
```


Les fonctions

- Pas de déclaration de la signature d'une fonction
- De 0 à n paramètres, de types quelconques
- Retour optionnel d'un résultat (instruction `return`)
- Transmission du résultat par copie

Exemples

```
function compare-majmin ($a, $b) {
  if (strtoupper ($a) == strtoupper ($b)) return 0;
  return (strtoupper ($a) < strtoupper ($b)) ? -1 : 1;
}
$tableau-trie = usort ($tableau, "compare-majmin");
$fp = fopen ($document);
if ($fp) {
 do { $ligne = fgets ($fp, 1024); ... }
    while (! feof ($fp));
}
```


Les formulaires

- Traitement des formulaires en php : on récupère les valeurs dans un tableau associatif appelé `$_REQUEST`.
-

```
<form action="formulaire.php3" method="POST">
Nom : <input type="TEXT" NAME="nom"><br>
Nationalite : <input type="TEXT" name="nationalite">
<input type="SUBMIT" name="OK"> </form>
```

- Code côté client

```
<?php
  echo $_REQUEST['nom'];
?>, <p>Vous avez la nationalite
<?php echo $_REQUEST['nationalite']; ?>
```

- Autres variables 'auto-globales' : `_GET`, `_POST`, `_COOKIE`, `_ENV`, `_SERVER`, `_SESSION`

Programmation modulaire

- S'effectue en réutilisant du code de bibliothèques existantes
- Les bibliothèques sont des fichiers PHP classiques
- Par convention, leur extension est `.inc`
- On inclut un fichier par `include` ou `require`
  - `include` : inclusion dynamique
  - `require` : inclusion avant l'exécution du code

Programmation objet

- Héritage simple
- Pas de méthodes ou attributs privés. Tout est public.
- Un objet instancié est une variable.
- Pas de destructeur

Bibliothèques

- En standard : bibliothèques d'accès à certaines fonctionnalités (envoi de mail, mySQL, calendrier, images, fichiers zip, ...)
- Autres : bibliothèques (voire frameworks) de plus haut niveau : phpNuke, phpGroupWare, spip, ...

Utilisation des SGBD

- Ajout du module adéquat dans `php.ini`
- Pour mysql par exemple :
  - `mysql_connect()` : connexion à une base de données
  - `mysql_pconnect()` : connexion persistente à une base de données
  - `mysql_list_*()` : retourne la liste des bases, tables ou champs disponibles sur le serveur
  - `mysql_select_db()` : sélectionne une base de données
  - `mysql_query()` : envoie une requête au serveur
  - `mysql_result()`, `mysql_fetch()` : exploitation du résultat
  - `mysql_close()` : fermeture de la connexion

Cookies

- Utilisation de la fonction `setcookie ()` :

```
int setcookie (string name, string value,
               int expire, string path,
               string domain, int secure);
```

- Tous les paramètres sauf le nom sont optionnels
- Les cookies sont passés dans l'entête HTTP. Il faut donc appeler la fonction `setcookie` avant de générer le code HTML.
- Les informations sont récupérables dans la variable `$_COOKIE` dans les documents suivants

Cookies - exemple

```
<?php
$nom = "test";
$valeur = "retest";
setcookie ($nom, $valeur, time() + 60);
?>
<html><body>
<h1>Utilisation de cookies</h1>
En consultant ce document, vous avez reu
sur votre navigateur un cookie appel <?=$nom?>
dont la valeur est <?=$valeur?>.<br>
<a href="co2.php">Vrification</a>
</body></html>
```


Cookies - exemple

```
<html>
<body>
<h1>Utilisation de cookies</h1>
La valeur du cookie test est <?=$test?>.
</body>
</html>
```


Session

- Notion introduite en standard dans php4.
- Ouverture de session :
  - attribution d'un identifiant au client
  - stockage sous forme de cookies
- Déclaration des variables à sauvegarder
- À la fin de chaque document, sauvegarde sur le serveur des variables spécifiées.
- À l'ouverture de chaque document, lecture de la valeur des variables associées à l'identifiant de session.