Introduction aux Outils pour le Web 2.0
Cours 5 - Php

Laurent Henocque
Polytech Marseille
Département Informatique
Mis à jour en Décembre 2013

laurent.henocque.com

Licence Creative Commons

PHP

- PHP signifie PHP: Hypertext Preprocessor
- PHP est un langage de scripts s'éxécutant du côté du serveur
- Les scripts PHP servent à la génération de pages HTML
- PHP fonctionne avec de nombreuses bases de données databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)
- PHP est open source
- PHP est gratuit à télécharger et utiliser

Le fichier PHP

- Les fichiers PHP peuvent contenir du HTML (tags, css, texte, javascript ...) et du code PHP
- Les fichiers PHP sont interprétés du côté du serveur, et leur sortie est retournée au navigateur comme du HTML
- Les fichiers PHP ont normalement l'extension".php" (parfois ".php3",...,".phtml")

MySQL

- MySQL est le SGBD le plus populaire de PHP
- MySQL permet une certaine montée à l'échelle, pour servir des applications importantes
- MySQL est conforme au standard SQL
- MySQL est disponible sur de nombreuses machines
- MySQL est gratuit à télécharger et utiliser

Pourquoi PHP?

- PHP fonctionne sur de nombreuses plate formes (Windows, Linux, Unix, etc.)
- PHP est disponible chez la quasi totalité des fournisseurs d'hébergement
- PHP est compatible avec la plupart des serveurs web modernes (**Apache**, IIS, etc.)
- PHP est facile à apprendre et à utiliser
- PHP s'exécute rapidement, les scripts pouvant être pré-compilés en cache

Le serveur WEB

- Pour utiliser PHP, il faut un serveur de pages web actif sur une machine (Apache par exemple)
- Ce serveur est normalement accessible sur le port 80 (port par défaut du protocole http)
- Le serveur va intercepter toutes les requêtes http de type http://monsite.fr/mapage
  - (rappel: monsite.fr est traduit par un DNS en l'adresse IP de la machine - ex: 123.65.123.2)

Accès aux fichiers

- L'adresse d'une page sur un site ne comporte souvent pas d'extension (.html, .php, etc.)
- Le serveur web est paramétré pour faire une recherche par priorité.
- Etant donné une page nommée 'mapage', il va par exemple en premier lieu chercher le script 'mapage.php', puis 'mapage.php3' (et autres), enfin 'mapage.html'

Quand l'URL désigne un dossier

- Dans ce cas, le serveur considère que la page s'appelle '**index**'
- Si aucun fichier 'index.html', 'index.php' etc. n'existe, le serveur affiche le contenu du répertoire
- Avec Apache, un fichier spécial '**`.htaccess`**' permet d'empêcher l'affichage des contenus de répertoire sur le site entier

```
Options -Indexes
```

- Un moyen technique simple d'empêcher cet affichage consiste à toujours placer un fichier index.html (même vide)

Le Source PHP

- Un source HTML valide est un fichier php valide. Il suffit de modifier l'extension en '.php'
- La machine virtuelle php du serveur resservira ce contenu sans modification, comme si c'était du html standard.
  - En fait, php traite tout ce qui ne le concerne pas comme des affichages (par la fonction '**`echo`**')
- Les fragments de programme inclus dans le fichier sont encadrés par les balises **`<?php`** et **`?>`**, ou **`<?`** et **`?>`**

Bonjour tout le monde

```
<!DOCTYPE html>
<html>
<body>
<?php
        echo "Bonjour tout le monde";
?>
</body>
</html>
```

Commentaires en PHP

```
<?php
du code // un commentaire fin de ligne
du code # un commentaire fin de ligne
/*
Un commentaire multi lignes
*/
et puis /* un commentaire */ dans du code
?>
```

Les variables PHP

- ne sont pas déclarées
- ne sont pas typées
- ont pour portée le bloc qui les déclare
- sont préfixées par le caractère '**$**'

```
<?php
   $message="Salut!";
   $num=7;
?>
```

Raccourci pour echo

```
<!DOCTYPE html>
<html>
 <head>
 <title><?=$titre_de_la_page;?></title>
 </head>
 <body>
 <p>Salut!</p>
 </body>
</html>
```

Les Entiers

- Entiers (machine dépendants): 32 bits signés
- Notations décimales, octales, hexadécimales

```
<?php
$a = 1234; // nombre decimal
$a = -123; // négatif
$a = 0123; // forme octale (= 83)
$a = 0x1A; // forme hexadecimale (= 26)
?>
```

Les Flottants

- Flottants machine dépendants
- Trois notations:

```
<?php
   $a = 1.234;
   $b = 1.2e3;
   $c = 7E-10;
?>
```

- Valeur spéciale **`NaN`** (not a number), testée avec la fonction **`is_nan(num)`**

Les Booléens

Sont convertis en 'FALSE':

- l'entier 0 (zero)
- le réel 0.0 (zero)
- la chaine vide ("") et la chaine "0"
- tout tableau de zéro éléments
- un 'Object' sans données membres (php4)
- la valeur NULL (dont celle de valeurs non initialisées)

Le type Resource

Les données de type 'resource' correspondent à des ressources externes

- Bases de données
- Images
- Fichiers

Null

- On utilise '**`null`**' pour spécifier qu'une variable est vide
- **`null`** est aussi la valeur par défaut des variables non initialisées
- **`null`** est évalué à FALSE dans les expressions booléennes

Portée des variables

- Les variables PHP sont locales ou globales
  - Les variables utilisées dans les fonctions, et les paramètres des fonctions sont locales
  - Les variables utilisées en dehors de toute fonction sont globales
- Pour faire référence à une globale dans une fonction, il faut la déclarer globale

Portée locale

```php
<?php
  $a = 7; // globale
  function foo() {
    echo $a; // (locale)
  }
  foo(); // n'affiche rien
?>
```

Accès aux globales

```php
<?php
  $a = 7;
  $b = 13;
  function bar() {
    global $a, $b;
    $b = $a + $b;
  }
  bar();
  echo $b; // affiche 20
?>
```

Variante: le tableau $GLOBALS

```php
<?php
$a = 7;
$b = 13;
function baz(){
    $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
}
baz();
echo $b; // affiche 20
?>
```

Rémanence: variables Static

- Il est parfois nécessaire de préserver la valeur d'une variable locale entre plusieurs exécutions d'une fonction

```
static $variableRemanente;
```

Chaînes de caractères

- Opérateur de concaténation: "."

```
<?php
   $txt1="Polytech Marseille";
   $txt2="la meilleure formation
     d'informatique de Marseille!";
   echo $txt1 . " : " . $txt2;
?>
```

Opérations sur les chaînes

- **`strlen (str)`**: calcule le nombre de caractères d'une chaîne
- **`strpos (meule, aiguille)`**: renvoie la position du premier caractère d'une sous chaîne dans une autre (ou FALSE si absent)
- Les positions dans les chaînes de caractères sont numérotées à partir de zéro

Opérateurs

- Arithmétiques: +,-, - unaire, *, /, %
- Affectation: =, +=, -=, *=, /=, .=
- Incréments pré et postfixes: ++, --
- Comparaison: == (avec conversions), === (identité), != (et <>), !===, <, >, >=, <=
- Logiques: ! (non unaire), && (et 'and'), || (et 'or'), xor (l'un des deux est vrai mais pas les deux)
- Tableaux: + (union), == (mêmes paires clé/valeur), === (==, dans le même ordre et de même types, != (<>), !==)

Tableaux

- Les tableaux PHP sont comparables en fonctionnalités aux tableaux javascript
- Les tableaux numériques sont indexés à 0 et épars (peuvent contenir des trous)
- Les tableaux associatifs stockent des paires clé/valeur

```
$pays=array("FR","UK,","US");
$prix=array("FR"=>3,"UK"=>5,"US"=>12);
echo $prix[$pays[0]]; // affiche 3
```

Tableaux à plusieurs dimensions

```
$tel = array (
   "FR"=>array("+33", 1.2),
   "US"=>array("+1", 5.4)
);

echo $tel['FR'][1]
```

Structures de contrôle

- PHP offre les structures de contrôle habituelles des langages de programmation

```
if (condition) {...} else {...}
switch (exp) {case:val;...break;...}
while (condition){...},
do {...} while (condition)
for (init; condition; incrément){...}
foreach ($array as $value){...}
```

FOREACH

```
<html><body>
<?php
$pays=array("FR","US","UK");
foreach ($pays as $p) {
  echo $p . "<br />";
  }
?>
</body></html>
```

Fonctions

```
<?php
function foo($parametre, ...)
{
    ...

    function bar_dans_foo()
    {
        ...
    }
    return ...;
}
?>
```

Fonctions

- Depuis la 5.3, PHP permet l'utilisation de fonctions anonymes

```
<?php
$bonjour = function {  return 'Bonjour'; }
$polytech = function(){
    return "Polytech";
}
echo $bonjour().' '.$polytech();
?>
```

Closures

```
public function Total($taxes){
   $total = 0.00;
   $callback =
      function ($qte, $prod) use ($taxes, &$total){
          $total += ...;
      };
   array_walk($this->products, $callback);
   return round($total, 2);
}
```

Classes et Objets

- PHP supporte une définition moderne des classes depuis la 5.3

```
class foo extends bar{
 function __construct() {
  $doo = "une donnée membre";
 }
 public function methode(){
  return $doo;
 }
}
```

Statiques

```
class Foo{
    public static $maStatique = 'foo';
    public function valStatique() { return self::$maStatique; }
}
class Bar extends Foo{
    public function fooStatic() { return parent::$maStatique; }
}

print Foo::$maStatique . "\n";
print new Foo()-> valStatique() . "\n";
$classname = 'Foo';
print $classname::$maStatique . "\n"; // depuis PHP 5.3.0
print Bar::$maStatique . "\n";
print new Bar()->fooStatic() . "\n";
```

Formulaires

- PHP est conçu notamment pour exploiter par programme les données transmises par un utilisateur via les formulaires (**`<form>`**)
- Les données sont transmises:
  - via l'URL (paramètres 'GET')
  - invisibles à l'utilisateur dans une zone de données d'au plus 64 K (paramètres 'POST')

PHP et les formulaires GET

- Le formulaire suivant lancera l'URL

```
http://monsite.fr/bonjour.php?nom=...&age=...
```

```
<html><body>
<form action="bonjour.php" method="get">
Nom: <input type="text" name="nom" />
Age: <input type="text" name="age" />
<input type="submit" />
</form></body></html>
```

Traitement des données: le fichier bonjour.php

- Lorsqu'on utilise la méthode GET, les données du formulaire sont récupérées en PHP via le tableau prédéfini **$_GET**

```
<html><body>
Bonjour <?= $_GET["nom"]; ?>!<br />
Vous avez <?= $_GET["age"]; ?> ans.
</body></html>
```

PHP et les formulaires POST

- Le formulaire suivant lancera l'URL

  http://monsite.fr/bonjour.php/

  (les paramètres sont transmis sans être visibles dans l'URL)

```
<html><body>
<form action="bonjour.php" method="post">
Nom: <input type="text" name="nom" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>
</body></html>
```

Traitement des données: le fichier bonjour.php

- Lorsqu'on utilise la méthode GET, les données du formulaire sont récupérées via le tableau **$_POST**

```
<html><body>
Bonjour <?= $_POST["nom"]; ?>!<br />
Vous avez <?= $_POST["age"]; ?> ans.
</body></html>
```

Limitations de $_GET

- En utilisant la méthode 'get', les paramètres du formulaire sont transmis via l'URL

  ```
  <form action="hello" method="get">
  ```

- produit:

  ```
  http://monsite.fr/hello?nom=Bob&age=37
  ```

- La méthode GET n'est pas appropriée pour le transfert de mots de passe, ni pour des données importantes ou codées, ni donc pour le téléchargement de fichiers
- Elle permet d'archiver les sites et leur référencement

Formulaire de Transfert de Fichiers

- L'upload de fichiers vers un site distant se fait avec la méthode 'post', en spécifiant l'encodage.

```
<form action="telecharger.php"
    method="post"
    enctype="multipart/form-data">
<label for="file">Filename:</label>
<input type="file" name="fic" id="fic"><br>
<input type="submit" name="submit"
value="Submit">
</form>
```

Principes de la récupération des fichiers

```php
<?php
$leFic= $_FILES["fic"];
if ($leFic["error"] > 0){
  echo "Erreur: " . $leFic["error"] . "<br>";
}else{
  echo "Téléchargement de: " . $leFic["name"] . "<br>";
  echo "Type: " . $leFic["type"] . "<br>";
  echo "Taille:" . ($leFic["size"]/1024) . "kB<br>";
  echo "Stocké dans: " . $leFic["tmp_name"];
}
?>
```

Récupération de fichier avec contrôle des types

```
<? $leFic= $_FILES["fic"]; $leType=$leFic["type"];
$extensions = array("jpg", "jpeg", "gif", "png");
$lext = end(explode(".", $leFic["name"])); //par ex "jpg"
if (((($leType == "image/jpeg")||.../*gif,png,pjpeg*/))
    && ($leFic["size"] < 20000)
        && in_array($lext, $extensions)){
  if ($leFic["error"] > 0){ ... } else {
    echo "Téléchargement de: " . $leFic["name"] . "<br>";
    ...
  }
} else { echo "Fichier invalide";} ?>
```

Enregistrement du fichier sur le serveur

```
$leNom= $_FILES["fic"] ["name"];
if (file_exists("upload/" . $_FILES["file"]["name"]){
  echo $leNom . " existe déjà.";
} else {
  move_uploaded_file(
    $leFic["tmp_name"],
    "upload/" . $leNom);
  echo "Stored in: " . "upload/" . $leNom;
}
```

Fonctions utiles sur les tableaux

- Les fonctions suivantes sont utiles pour exploiter les paramètres GET et POST

```
bool array_key_exists (mixed $k, array $search )
bool isset (mixed $v [, mixed $... ] )
array array_keys ( array $input [, mixed $search_value = NULL [, bool $strict = false ]] )
bool in_array (mixed $needle, array $haystack [, bool $strict = FALSE ] )
bool property_exists (mixed $c, string $prop )
```

Ajax: partie HTML

```
<body>
<p>Entrez le début d'un prénom</p>
<form>
  prénom: <input type="text"
    onkeyup="suggestions(this.value)">
</form>
<p>
Suggestions: <span id="hint"></span>
</p>
</body>
```

Ajax: partie javascript

```
<head><script> function suggestions(str){
  var hint= document.getElementById("hint");
  if (str.length==0){ hint.innerHTML=""; return;}
  xmlhttp=new XMLHttpRequest();
  xmlhttp.onreadystatechange=function() {
    if (xmlhttp.readyState==4 && xmlhttp.status==200)
      hint.innerHTML=xmlhttp.responseText;
  }
  xmlhttp.open("GET","gethint.php?q="+str,true);
  xmlhttp.send();
} </script></head>
```

Ajax: partie PHP
le programme gethint.php

```
<?php $a=array("Anna", "Bob", ...);
$q=$_GET["q"]; // récupérer la valeur de 'q'
if (strlen($q) > 0) { // construire les suggestions
  $hint="";
  for($i=0; $i<count($a); $i++) {
    if (strtolower($q)==strtolower(substr($a[$i],
0,strlen($q)))) {
      if ($hint=="") $hint=$a[$i];
      else $hint=$hint." , ".$a[$i];
 }}}
if ($hint == "")  echo "Aucune suggestion";
else echo $hint; ?>
```

Variables de variables

- Il est parfois utile de manipuler des variables de variables

```
<?php
   $a = 'bonjour';
   $$a = 'polytech';
?>
```

- Nous avons ici deux variables: $a et $bonjour

```
<?php echo "$a ${$a}"; ?>
```

- produit la même trace que :

```
<?php echo "$a $bonjour"; ?>
```

En guise de Conclusion

Qu'avons nous vu?

- Quatre langages pour le Web 2:
  - HTML : le contenu
  - CSS: l'apparence
  - Javascript: la dynamique
  - PHP: le serveur

Que n'avons nous notamment pas vu?

- Tout HTML, CSS, Javascript, PHP
- Les autres langages pour l'apparence et la dynamique - les langages génériques
- Les autres langages pour le serveur
- Les bibliothèques pour la dynamique
- Les bases de données
- Les frameworks et CMS côté serveur

Autres langages pour la dynamique

- Adobe Flash (propriétaire)
- Dart (traduction vers javascript)
- Java (traduction vers javascript)
- ...

Autres langages pour le serveur

- Java (J2EE)
- Python
- Ruby ↓
- Go (Google) ↓
- Dart (Google) ↓
- Javascript (NodeJS) ↑

Bibliothèques javascript

- Apportent de la portabilité sur tous les navigateurs (mécanisme des 'polyfills')
- Etendent les fonctionnalités
- Apportent la fonction d'accès '$'

```
getElementById("bob") => $("bob")
```

Bibliothèques Javascript

- JQuery
- Mootools
- Prototype.js
- script.aculo.us
- ExtCore
- jsPHP
- ...

Les frameworks

- Les frameworks simplifient l'utilisation des bases de données (mysql): **objets actifs**, **scaffolding**
- Simplifient la génération et le décodage des URL
- Organisent le code en trois groupes de fonctionnalités:
  - les contrôleurs : pages php ouvertes par le client
  - le modèle : accès et modification des bases de données
  - les vues : génération des pages affichées

Frameworks

- Yii (PHP)
- CodeIgnitor (PHP)
- CakePHP
- Flex (Adobe)
- ...

Systèmes de gestion de contenu gratuits PHP

- Wordpress
- Joomla
- Drupal
- CMSMS (CMS made simple)
- et TextPattern, Contao ...

Fin du document

Et voila ! A bientôt