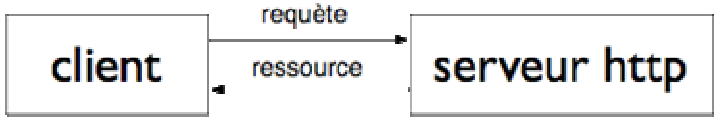
Séance 7
Introduction à PHP

19 mars 2007

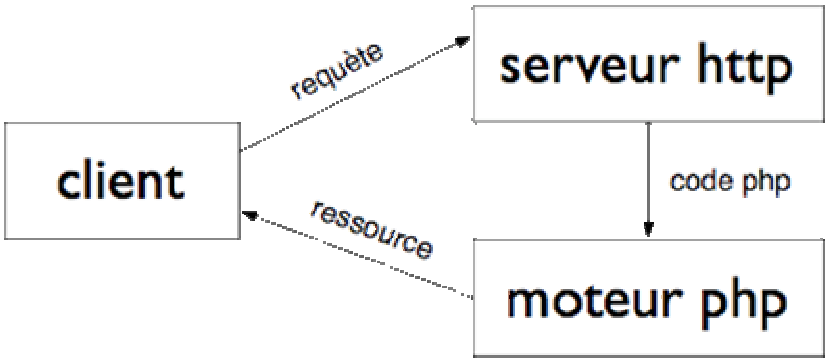
1. PHP : Fonctionnement

- Contenu "statique" vs. contenu "dynamique"
- Du point de vue du développeur
  - passage de la requète en paramètre
  - génération d'une page fonction de la requète

page statique



page générée



2. PHP : Principe

- Embarquer du code dans une page HTML
  - exécuté par le serveur
  - remplacé par son résultat

- Exemple trivial
  - Côté serveur : Il est acuellement <?php print date("H:i:s"); ?>.
  - Côté client : Il est acuellement 8:45:34.

3. PHP : Généralités

- Extension du fichier : *.php*
- Délimiteurs possibles
  - <? ... ?> **<?php ... ?>** <script language="php"> ... </script>
- Variables : précédées de “$”, pas de typage explicite

entiers : int

réels : float, double

chaînes de caractères : string

booleen : boolean

4. PHP : Généralités / exemples

```
$reponse = 42; # entier
$pi = 3.14159; # réel
$verbe = "lire"; # chaîne
$phrase = "J'aime $verbe"; # chaîne avec interprétation
$phrase = 'It cost $100'; # chaîne sans interprétation
$ceci = $cela; # variable
$E = $m*$c*$c; # expression
```

La variable est toujours précédée de $

Initialisation :

```
$tab = array("h", "1", "test") ;
```

Des éléments peuvent être ajoutés dynamiquement :

```
$tab[3] = "suite" ;
$tab[] = "fin" ; // ajout en fin de tableau
```

Tester si une variable est un tableau :

```
boolean is_array(var) ;
```

5. PHP : Tableaux et parcours

- Initialisation :

```
$tab[1] = valeur1;
$tab = array(1,2,3);
$tab["cle1"] = valeur1;
$tab = array("cle1"=>valeur1, "cle2"=>valeur2);
```

- Fonctions de parcours:

reset, end, prev, next : parcours linéaire

key($tab) : retourne la clé de l'élément courant.

each($tab) : retourne la paire (clé, valeur) suivante du tableau associatif.

6. PHP : Tableaux et parcours / exemples

**count($tab**) : retourne le nombre d'éléments du tableau.reset($tab) ;
**end($tab**) : place le pointeur interne sur le premier (dernier) élément et le retournenext($tab) ; prev($tab) : place le pointeur interne sur l'élément suivant (précédent) et le retournecurrent($tab) : retourne l'élément courant.
split($sep, $chaine) : transforme une chaîne en tableau ($sep étant le séparateur).
join($sep, $tab) : transforme un tableau en chaîne de caractères

Exemples :

```
$tab = array("a", "b", "c") ;
$c = count($tab) ; // $c vaut 3
$res = end($tab) ; // $res vaut "c"
$res = prev($tab) ; // $res vaut "b"
```

Conversion chaîne/tableau :

```
$chaine = join(":", $tab) ; // $chaine vaut "a:b:c"
$tab2 = split(":", $chaine) ; // $tab2 == $tab
```

Parcours de tableau:

```
$tab = array("un"=>1, "deux"=>2, "trois"=>3) ;    // premier parcours
for(reset($tab) ; $cle = key($tab) ; next($tab)) {
        $val = current($tab);
        print("$cle = $val<BR>\n");
}
// second parcours
reset($tab);
while(list($cle, $valeur) = each($tab)){
        print "$cle = $valeur<BR>";
}
```

7. PHP : Opérateurs et expressions

- Opérateurs usuels "à la C"

  + - * / == && || ...

  ++ : pre ou post incrémentation

  -- : pre ou post d´ecrémentation

  += : incrémentation + affectation

  -= : d´ecrémentation + affectation

- Opérateurs sur les chaînes

  . : concaténation

  .= : concaténation + affectation

8. PHP : Opérateurs et expressions / exemples

```
function double($i) {
        return $i*2;
}
$b = $a = 5;        /* Assigne la valeur 5 aux variables $a et $b  */
$c = $a++;          /* Post-incrmentation de la variable $a et assignation
                       de la valeur  la variable $c */
$e = $d = ++$b;     /* Pre-incrémentation, et assignation de la valeur aux variables $d et $e  */

/*  A ce niveau, les variables $d et $e sont gales  6 */

$f = double($d++);      /* assignation du double de la valeur de $d  la variable $f
                                ($f vaut 12), puis incrémentation de la valeur de $d  */

$g = double(++$e);      /* assigne deux fois la valeur de $e après incrémentation, 2*7 = 14 to $g */

$h = $g += 10;      /* Tout d'abord, $g est incrémente de 10, et donc $g vaut 24. Ensuite, la valeur
                       de $g, (24) est assigne  la variable $h, qui vaut donc elle aussi 24. */
```

Associativité et précédence

```
$a = 3 * 3 % 5; // (3 * 3) % 5 = 4
$a = 1;
$b = 2;
$a = $b += 3; // $a = ($b += 3) -> $a = 5, $b = 5
```

9. PHP : Structures de contrôle

- if, else, elseif, for, while, switch . . .
  - syntaxe similaire à la syntaxe C
- include :
  - inclusion d’un autre script
  - warning si le script n’est pas présent
- require :
  - inclusion d’un autre script
  - erreur fatale si le script n’est pas présent

10. PHP : Structures de contrôle / exemples

Plusieurs syntaxes sont possibles selon la longueur du segment HTML généré conditionnellement:

```
<body>
<?php
  $a = 3 ;
  if ($a ==3) {
    echo "A vaut 3 <br />'' ;
  } else {
    echo "A ne vaut pas 3 <br />" ;
  }
?>
</body>
```

Ou bien:

```
<body>
<?php $a = 3 ; if ($a ==3) : ?>
A vaut 3 <br />
<?php else : ?>
A ne vaut pas 3 <br />
<?php endif; ?>
</body>
```

D'autres variantes sont possibles: gardez votre code lisible !

11. PHP : Fonctions

- Syntaxe à la pascal (cf exemple)
  - sans typage, comme les variables !
- Passage de paramètres
  - par défaut, par valeur
  - passage par référence (à la C++) possible avec &
    - soit à l'appel
    - soit à la déclaration

12. PHP : Fonctions / exemples

```
<?php
  function plus_un($x) {
          return $x+1 ;
  }
?>
```

Passage par référence à la déclaration:

```
<?php
  function incremente(&$x) { $x += 1 ; }
  $a = 1;
  incremente($a);
  print $a;
?>
```

Passage par référence à l'appel:

```
<?php
  function incremente($x) { $x += 1 ; }
  $a = 1;
  incremente(&$a);
  print $a;
?>
```

13. PHP : portée des variables

- Une variable est visible jusqu'à la fin de la *page* ou de la *fonction* englobante
  - Une page forme un seul "bloc" PHP
  - L'inclusion se fait à la manière de "cpp" (préprocesseur C/C++)
  - Accès aux variables globales : "global $var"
    Attention: pas comme en C !
- Durée de vie : chargement de la page.

14. PHP : Variables prédéfinies

- **$_SERVER** : Les variables fournies par le serveur web
- **$_GET ; $_POST ; $_COOKIE** : Les variables fournies par le protocole HTTP en méthode GET, POST, ou dans les cookies
- **$_FILES** : Les variables fournies par le protocole HTTP, suite à un téléchargement de fichier.
- **$_ENV** : Les variables fournies par l'environnement.
- **$_SESSION** : Les variables qui sont actuellement enregistrées dans la session attachée au script.

15. PHP : Variables prédéfinies / exemples

– form.html

```
<form action="traite.php" method="GET"/>
        <input type=text name="adresse">
        <input type="submit">
</form>
```

– traite.php

```
<p>L’adresse est <?php echo $_GET['adresse'] ?></p>
```

16. PHP : Envoi de fichiers (1/2)

- Utilisation d'un formulaire
  - type MIME "multipart/form-data"
  - côté serveur, utilisation de la variable $_FILE
  - le fichier est automatiquement
    décodé et stocké par PHP

- Contenu du fichier envoyé
  dans une requète POST

17. PHP : Envoi de fichiers (1/2) / exemples

Exemple de formulaire:

```
<form enctype="multipart/form-data" action="upload.php" method="POST">
        Fichier à envoyer : <input name="userfile" type="file" />
        <input type="submit" value="Envoyer" />
</form>
```

18. PHP : Envoi de fichiers (2/2)

- Côté serveur:
  Reception de la reqête HTTP :
  - Décodage du corps MIME de la requête
  - Création du fichier dans un répertoire temporaire
  - Récupération du fichier par une variable super globale, $_FILE, un tableau associatif :
    - Clé : nom du champs du formulaire
    - Valeur : un tableau associatif
      name : nom du fichier chez le client
      type : type MIME si positionné par le navigateur
      size : taille du fichier uploadé en octets
      tmp_name : nom du fichier sur le serveur
      error : code d'erreur

19. PHP : Envoi de fichiers (2/2) / exemples

Exemple de script "upload.php":

```
<?php

$uploaddir = $_ENV['HOME'] + ’/tpi_html/upload/’;

$uploadfile = $uploaddir . $_FILES[’userfile’][’name’];

if (move_uploaded_file($_FILES[’userfile’][’tmp_name’], $uploadfile)) {
        print ("Chargement correctement effectué:\n") ;
        print_r($_FILES);
        print("$uploadfile\n") ;
} else {
        print "Erreur:\n";
        print_r($_FILES);
}

?>
```

20. PHP : Cookies

- Envoi : utilisation de la fonction prédéfinie :
  seul le nom est obligatoire

  bool setCookie (string name , string value,
  int expire, string path, string domain, bool secure)

- Attention : setCookie() doit être appelé *avant* de commencer la page HTML !

  rappel: les cookies sont envoyés dans les en-têtes HTTP

- Réception : accès aux cookies par le tableau associatif super global $_COOKIE

21. PHP : Cookies / exemples

Un cookie qui expire dans 1 heure

```
<?php
        setCookie("test", 3, time()+3600) ;
?>
<html> ... </html>
```

Heure d'expiration :

- Nombre de secondes depuis le 1er Janvier 1970
- Utilisation de la fonction time()
- Utilisation de mktime()

```
        mktime(h,m,s,mois,jour,année) ;
```

22. PHP : Cookies complexes

- Stockage de données de type tableau :
  trois solutions
  - convertir le tableau en chaîne
    en utilisant join() et split()
  - autant des cookies que de cases de tableau
  - utiliser la sérialisation (conversion de n'importe quel objet en chaîne de caractères)
    avec serialize() et unserialize()

23. PHP : Cookies complexes / exemples

tableau.php --

```
<?php
  $a = array(1,2,3) ; setCookie(test, join(",", $a)) ;
?>
<html>
  <body>
    Verification du <a href="verif-tableau.php">cookie</a>
  </body>
</html>
```

verif-tableau.php --

```
<html>
  <body>
    Valeur brute reçue : <?php echo $_COOKIE['test']; ?> <br />
    Valeur décodée :
    <?php
      $a = split(",",$_COOKIE['test']) ; print_r($a) ;
    ?>
  </body>
</html>
```

24. PHP : Sessions, principe

- Besoin :
  - identifier chaque utilisateur de manière unique
  - utiliser des données persistantes pour chaque utilisateur (ex: login, panier...)
  - fiable et sûr (au contraire des cookies)
- Implémentation :
  - identifiant de session unique, stocké dans un cookie (nommé PHPSESSID par défaut)
  - base de données qui associe un ensemble de variables à un identifiant

25. PHP : Sessions, utilisation

- session_start() :
  - démarre une nouvelle session si elle n'existe pas (pas de cookie de session)
  - restaure la session courante si elle existe

- Création et utilisation d'une variable de session
  - Utilisation de la super globale :
    $_SESSION['variable']
    isset($_SESSION['variable'])
  - Suppression d'une variable de session :
    unset($_SESSION['variable'])

26. PHP : Sessions, utilisation / exemples

page1.php --

```
<?php
  session_start();
  echo 'Bienvenue à la page numéro 1';
  $_SESSION['favcolor'] = 'vert';
  $_SESSION['animal']   = 'chat';
  $_SESSION['time']     = time();
  // Fonctionne si le cookie a été accepté
  echo '<br /><a href="page2.php">page 2</a>';
?>
```

page2.php --

```
<?php
  session_start();
  echo 'Bienvenue sur la page numéro 2<br />';
  echo $_SESSION['favcolor']; // vert
  echo $_SESSION['animal'];   // chat
  echo date('Y m d H:i:s', $_SESSION['time']);
?>
```

27. PHP : Classes et objets

- Définition des membres d'une classe : **propriétés et méthodes**

```
class MyClass {
  public $var = 0;
  public function method() {
    ...
  }}
```

- **Héritage**

```
class SubClass extends BaseClass { ... }
```

- **Création / desctruction d'instances**

```
$obj = new A();  $obj = null;
```

- **Accès aux membres** : $obj->var = 'bla'; $obj->method();
- **Visibilite**

public	accès par tous
protected	accès par les classes filles
private	aucun accès

28. PHP : Communication avec MySQL

- Principe :
  – émission de requêtes en texte
  – réception de lignes tableau en retour

- Fonctions principales :

mysql_(p)connect	connection (persistante) au serveur MySQL
mysql_select_db	choix de la base
mysql_query	émission d'une requête
mysql_fetch_assoc	récupération d'une ligné résultat dans un tableau associatif
mysql_free_result	nettoyage
mysql_close	fermeture de la connection

29. PHP : Communication avec MySQL / exemples

```
<?php
// Connexion et sélection de la base
$link = mysql_connect("mysql_hote", "mysql_utilisateur", "mysql_mot_de_passe")
    or die("Impossible de se connecter");
 echo "Connexion réussie";
mysql_select_db("my_database") or die("Could not select database");

// Exécuter des requêtes SQL
$query = "SELECT * FROM my_table";
$result = mysql_query($query) or die("Query failed");

// Afficher des résultats en HTML
echo "<table>\n";
 while ($line = mysql_fetch_assoc($result)) {
    echo "\t<tr>\n";
    foreach ($line as $col_value) {
        echo "\t\t<td>$col_value</td>\n";
    }
    echo "\t</tr>\n";
 }
 echo "</table>\n";
mysql_free_result($result);          // Libération des résultats
mysql_close($link);                  // Fermeture de la connexion
?>
```

30. SQL : Rappels

- Création / destruction d'une table

```
CREATE TABLE groupe1.clients ( numero INTEGER NOT NULL DEFAULT 0
PRIMARY KEY AUTO_INCREMENT, nom VARCHAR(255) NOT NULL,
prenom VARCHAR(255) NOT NULL, email VARCHAR(255) NOT NULL );

DROP TABLE groupe1.clients;
```

- Insertion d'une ligne

```
INSERT INTO groupe1.clients ( nom, prenom, email )
VALUES ( 'John', 'Smith', 'john@smith.com' );
```

- Sélection

```
SELECT nom, email

FROM groupe1.clients

WHERE nom LIKE '%john%'

ORDER BY nom ASC;
```

31. Pour aller plus loin

- Classe d'abstraction PHP pour la communication avec des bd :
  - ADOdb : http://adodb.sourceforge.net/
  - Extention PHP Data Objects : http://fr2.php.net/pdo

- Framework PHP MVC :
  - Symphony : http://www.symfony-project.com/
  - CakePHP : http://www.cakephp.org/

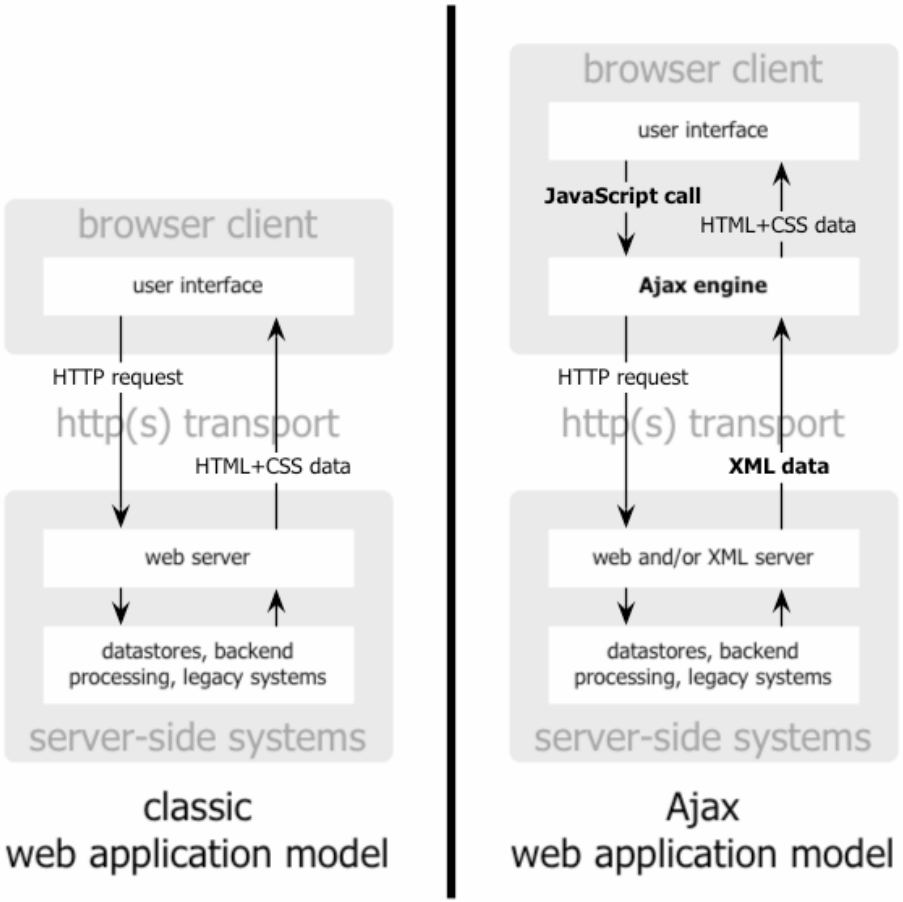
- Framework Ruby :
  - Ruby on Rails : http://www.rubyonrails.org/

- Framework Python
  - Django : http://www.djangoproject.com/

AJAX

AJAX n'est pas une technologie en elle-même, mais un terme qui évoque l'utilisation conjointe d'un ensemble de technologies couramment utilisées sur le Web :

- HTML (ou XHTML) pour la structure sémantique des informations
- CSS pour la présentation des informations
- DOM et JavaScript pour afficher et interagir dynamiquement avec l'information présentée
- l'objet XMLHttpRequest pour échanger et manipuler les données de manière asynchrone avec le serveur Web
- XML



http://www.adaptivepath.com/publications/essays/archives/000385.php