Introduction au langage PHP

BIENVENUE

- Présentation
- Organisation
- Tour de table
- Programme

ORGANISATION

- Horaires

10h00 – 12h30

13h30 – 17h00

Une pause le matin et l'après-midi

Merci de mettre vos portables en mode silencieux et de ne pas les utiliser pendant la formation.

TOUR DE TABLE

- Qui êtes-vous, votre fonction-métier ?
- Vos connaissances sur le sujet
- Quels sont vos objectifs ?

PROGRAMME

- Introduction
  - Définition
  - Documentation PHP
- Prérequis
  - Fonctionnement d'un serveur
  - Schéma d'exécution d'un script PHP
  - Serveur local
  - Editeur avancé
- Installation
  - Installation de Xampp
- Premier pas en PHP
  - Syntaxe générale du langage
  - Le Hello World
  - Les commentaires
- Les variables et les types
  - Les variables
  - La fonction Echo
  - La concaténation
  - Les types
  - Les conversions des types
  - Conversion automatique
- Les opérateurs arithmétiques et logiques
- Les structures de contrôles
- Les structures itératives
- Les fonctions

PROGRAMME

- Passage de variable en $_GET
- Formulaires
- Base de données
  - Base de données
  - Gestion des erreurs
- CRUD
- Sessions
- Sécurité
- Mise en ligne
- Conclusion

INTRODUCTION

DÉFINITION

- PHP (Hypertext Preprocessor), créé en 1994 par Rasmus Lerdorf, est un langage de scripts libre principalement utilisé pour être exécuté par un serveur HTTP, mais il peut fonctionner comme n'importe quel langage interprété de façon locale, en exécutant les programmes en ligne de commande.

DOCUMENTATION PHP

- Version en ligne
  - http://www.php.net/manual/fr/preface.php
- Version hors ligne (téléchargement)
  - http://fr3.php.net/download-docs.php
- Autre
  - http://www.nexen.net (recommandée)
- Logiciel sur mac : dash

PRÉREQUIS

FONCTIONNEMENT D'UN SERVEUR

Fonctionnement d'un serveur Web.

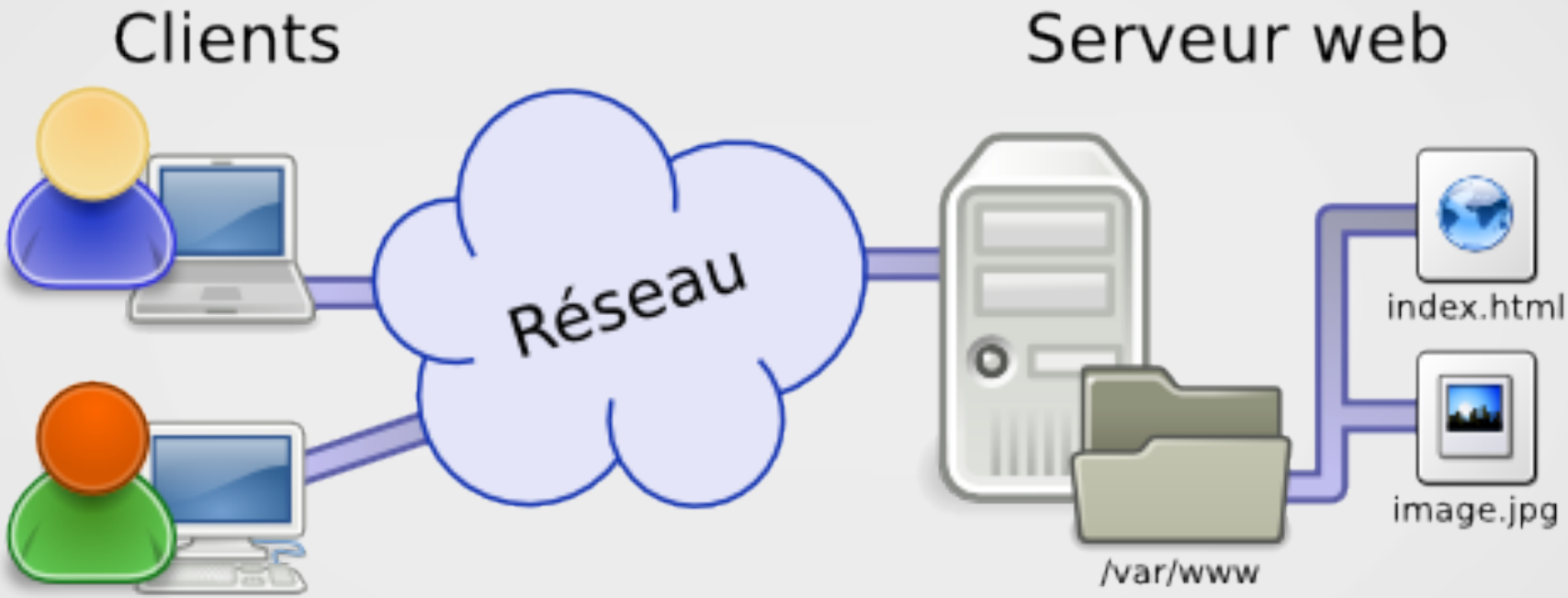
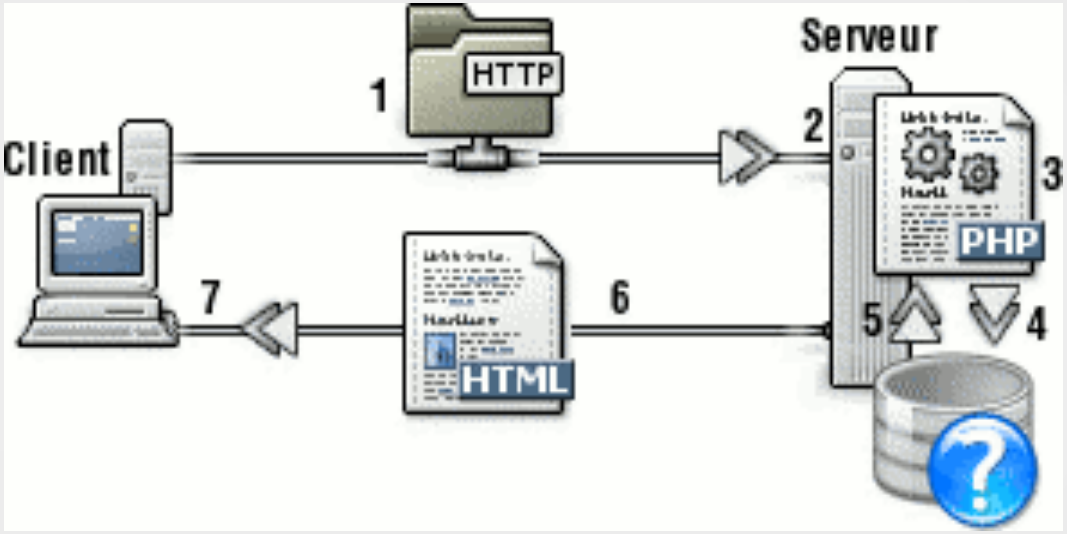


SCHÉMA D'EXÉCUTION D'UN SCRIPT PHP



SERVEUR LOCAL

- Nous avons besoin d'un serveur web Apache, un moteur PHP et d'un serveur de base de donnée MySQL qui soient installés sur nos machines.

- Il faut utiliser un logiciel qui permet de développer en local (sur votre propre ordinateur).

- C'est une plate-forme de développement Web pour des applications Web dynamiques (Par exemple : site développé en PHP).

- Existe sous Windows (Wamp, Easy PHP, Xampp) ou Mac (Mamp).

SERVEUR LOCAL

WAMP signifie :

- **W** indows
- **A** pache (le serveur web)
- **M** ySQL (base de données)
- **P** HP

MAMP signifie :

- **M** acintosh
- **A** pache (le serveur web)
- **M** ySQL (base de données)
- **P** HP

SERVEUR LOCAL

- **Apache**: est un serveur HTTP créé et maintenu au sein de la fondation Apache. C'est le serveur HTTP le plus populaire du World Wide Web.

- **MySQL**: est un système de gestion de bases de données relationnelles.

- **PHP**: est un langage de programmation libre principalement utilisé pour produire des pages Web dynamiques.

EDITEURS AVANCÉS

- NotePad, Notepad2, Notepad++,Emacs, Vim … (free).
- NuSphere PhpED, PHP Storm (IDE payant).
- Zend Development Environment (IDE payant).
- Dreamweaver (IDE payant).
- Sublime, Brackets …

INSTALLATION

INSTALLATION DE XAMPP

- Télécharger et installer Xampp (Xampplite).
  - http://www.apachefriends.org/fr/xampp.html

- Pour tester si Xampp est correctement installé:
  - Tapez « localhost » dans la barre d'adresse de votre navigateur
  - La page d'accueil de Xampp doit s'afficher

  - Si cela ne fonctionne pas, c'est peut-être que le port par défaut est déjà utilisé

INSTALLATION DE XAMPP

- Pour visualiser votre première page, créez dans votre éditeur avancé une première page puis enregistrez la page dans un dossier (par ex: projet1):
  - C:\xampp\htdocs**projet1\index.php**

- Ouvrez votre navigateur web, tapez l'adresse:
  - http://**localhost**/projet1/index.php

PREMIER PAS EN PHP

SYNTAXE GÉNÉRALE DU LANGAGE

```php
<?php

        //Corps du script

?>
```

index.php

- Il faut enregistrer notre document sous l'extension *.php* pour qu'il soit interprété par le serveur comme étant un document php.
- Bien que les balises *<?* et *?>* (au lieu de *<?php* et *?>*) puissent être également utilisées sur une configuration par défaut, elles sont peu recommandées car elles dépendent de la configuration du serveur.

LE HELLO WORLD

- PHP peut être imbriqué avec du HTML.

```
<html>
  <head>
        <title>Test page</title>
  </head>
  <body>
        <?php
                echo "<h2>Hello World :)</h2>";
        ?>
  </body>
</html>
```

index.php

LE HELLO WORLD

- Vous apporter du dynamisme en affichant la date du jour en tête de page à l'aide du code PHP suivant.

```
<?php
  echo "<h3> Aujourd'hui le ". date('d / M / Y H:m:s ')."</ h3><hr />";
  echo "<h2>Bienvenue sur le site PHP 5</h2>";
?>
```

LE HELLO WORLD

- Le code PHP est toujours incorporé dans du code HTML.
- Vous pouvez incorporer autant de scripts PHP indépendants que vous le souhaitez n'importe où dans du code HTML, du moment que ces parties sont délimitées par les balises ouvrantes et fermantes:
  - *<?php* et *?>*

LES COMMENTAIRES

- Un commentaire permet de vous y retrouver dans votre code.
- Ce texte est ignoré durant la génération de la page.
  - Commentaire une ligne: // Texte ou # Texte
  - Commentaire multi-lignes: /* Texte */

```php
<?php
  echo "Ceci est un test"; // Ceci est un commentaire sur une ligne

  /* Ceci est un commentaire
  sur plusieurs lignes*/
  echo "Ceci est encore un test";

  echo "autre commentaire"; # Ceci est un commentaire comme en Shell Unix
?>
```

index.php

INCLUSION D'UN CODE PHP

- PHP.NET – include
  - http://php.net/manual/fr/function.include.php

- La fonction include() permet d'insérer une page dans une autre.
  - On peut mettre du PHP dans la page incluse.
  - La page incluse est directement insérée dans le code

```php
<?php include('une_autre_page.php'); ?>
```

include.php

INCLUSION D'UN CODE PHP

- Il est possible d'inclure « menu.php » en haut de toutes les pages de mon site.
- Il est possible de séparer le haut « header.php » et le bas « footer.php ».

```
// CODE HTML DU HAUT (doctype, html, head, body)
<?php include('header.php'); ?>

<?php include('menu.php'); ?>
<div id="main-container">CONTENU</div>

// CODE HTML DU BAS (balise de fin: html, body)
<?php include('footer.php'); ?>
```

include.php

LES VARIABLES ET LES TYPES

LES VARIABLES

- En PHP, les variables sont représentées par le caractère dollar "$" suivi du nom de la variable.

$Nom_de_la_variable

- Le nom est sensible à la casse (*$x* est différent de *$X*).
- Un nom de variable valide doit commencer par une lettre ou un underscore (_), suivi de lettres, chiffres ou soulignés.

LES VARIABLES

- Exemples de variables valide ou non valide.

Variable	
$name	Valide
$_name	Valide
$1name	**Non Valide**
$_1name	Valide
$NaMe3	Valide

LA FONCTION ECHO

- La fonction echo permet d'afficher un texte qui se trouve entre
  - " " (avec interpolation)
  - ' ' (sans interpolation)

```php
<?php
  echo "Ceci est une chaîne simple";
  // affiche : Ceci est une chaîne simple

  echo 'Arnold a coutume de dire : "I\'ll be back"';
  // affiche : 'Arnold a coutume de dire "I'll be back"

  echo 'Les variables ne seront pas $afficher $ici';
  // affiche : Les variable ne seront pas $afficher $ici
?>
```

echo.php

LA FONCTION ECHO

- Pour afficher la date et l'heure du jour.

```php
<?php
  $date_du_jour = date ("d-m-Y");
  $heure_courante = date ("H:i");

  echo 'Nous sommes le : ';
  echo $date_du_jour;
  echo ' Et il est : ';
  echo $heure_courante;
?>
```

LA CONCATÉNATION

- Pour concaténer une chaine nous utilisons:
  - Le point . (Indique à PHP que le nom de la variable s'arrête à un endroit précis)
- Le code HTML ou texte est placé entre:
  - '' simples quotes: aucune variable n'est remplacée, aucun caractère n'est échappé (plus rapide)
  - "" double quotes: les variables sont remplacées, les caractères échappés (plus lent)

```
<?php
  $date_du_jour = date ("d-m-Y");
  $heure_courante = date ("H:i");

  echo 'Nous sommes le: '.$date_du_jour. 'et il est : '.$heure_courante;
?>
```

LES TYPES

- PHP **ne nécessite pas de déclaration explicite** du type d'une variable.
- Le type d'une variable est déterminé par **le contexte d'utilisation**.

Exemple:

si vous assignez **une chaîne de caractères** à la variable *$var*, *$var* devient une chaîne de caractère.

Si vous assignez **un nombre entier** à la variable *$var*, *$var* devient un entier.

Le nom d'une variable en PHP est simplement une étiquette.

```
$nombre = 10; // sans préciser le type de la variable

$nombre = (int) 10; // cast (précise le type de la variable)
```

LES TYPES

1. Booléens

- PHP.NET – BOOLEAN
  - http://php.net/manual/fr/language.types.boolean.php
- C'est le type le plus simple. Un booléen exprime une valeur de vérité. Il peut prendre comme valeur soit *TRUE* soit *FALSE*.

```
<?php

  $foo = true; // assigne la valeur TRUE à la variable $foo

?>
```

index.php

LES TYPES

- Pourquoi utiliser la variable *foo* ?
  - Les termes *foobar* (/ f û b ɑː r /), ou *foo* et d' autres sont utilisés comme noms d'espace réservé (également appelés les variables metasyntactic) dans la programmation informatique ou de la documentation liée à l' informatique.
  - Ils sont utilisés pour nommer des entités telles que **des variables, des fonctions** et **des commandes dont l'identité exacte n'est pas importante et servent uniquement à démontrer un concept**.

LES TYPES

2. Entiers

- PHP.NET – integer
  - http://php.net/manual/fr/language.types.integer.php

- Un entier est un nombre: -2, -1, 0, 1, 2 ...

- Les entiers peuvent être spécifiés en base décimale (base 10), hexadécimale (base 16) ou octale (base 8).
- Les entiers peuvent être optionnellement précédés par le signe plus ou moins (- ou +).
- Pour utiliser la notation octale, vous devez préfixer le nombre avec un zéro; pour utiliser la notation hexadécimale, vous devez préfixer le nombre avec 0x .

LES TYPES

```php
<?php
  $a = 1234;        // nombre entier décimal (base 10)

  $a = -123;        // nombre entier négatif

  $a = 0123;        // nombre octal (équivalent à 83 en décimal)

  $a = 0x1A;        // nombre héxadecimal (équivalent à 26 en décimal)

  $a = 0b11111111; // nombre binaire (équivalent à 255 en decimal)
?>
```

index.php

LES TYPES

3. Float

- PHP.NET – float
  - http://php.net/manual/fr/language.types.float.php
- Nombres décimaux, (aussi connus comme nombres à virgule flottante, "floats", "doubles", ou "real numbers")

```php
<?php
  $a = 1.234;
  $b = 1.2e3;
?>
```

float.php

LES TYPES

4. Les chaines de caractères

- PHP.NET - string
  - http://php.net/manual/fr/language.types.string.php

- Les chaînes de caractères sont des séquences de caractères.
- En PHP, un caractère est un octet et il y en 256 de possibles.

```
<?php
   $firstName = "Jonathan";
   $lastName = "Miller";
   $fullName = $firstName." ".$lastName;

   echo strtoupper($fullName)."<br>"; // Renvoie une chaîne en majuscules
   // JONATHAN MILLER
   echo substr($firstName, 0, 3)."<br>"; // Retourne un segment de chaîne
   //Jon
?>
```

LES TYPES

5. NULL

- PHP.NET NULL - La valeur spéciale NULL représente une variable sans valeur. NULL est la seule valeur possible du type NULL.
  - http://php.net/manual/fr/language.types.null.php

- La valeur spéciale *NULL* représente l'absence de valeur.
- Une variable avec la valeur *NULL* n'a pas de valeur.

- $var = ''; // string
- $var = 'NULL'; // variable sans valeur

LES TYPES

6. Les tableaux

- PHP.NET array - Crée un tableau
  - http://php.net/manual/fr/function.array.php

- Un tableau PHP et un tableau HTML sont deux choses complètement différentes.
  - Un tableau PHP a pour fonction de stocker et manipuler des données.
  - Un tableau HTML sert à présenter des données sur un écran.

- Les tableaux, sont appelés *arrays* en anglais
  - Ce sont des types de données structurés permettant de grouper des informations ensemble. Les tableaux peuvent stocker une ou plusieurs valeurs à la fois (de types différents).

LES TYPES

6. Les tableaux

- Lors de la déclaration d'un tableau, il est inutile de préciser sa dimension et le type de données qu'il va contenir. PHP s'en charge automatiquement.
- Les tableaux sont dits dynamiques. A chaque nouvelle entrée enregistrée dans le tableau, PHP agrandit sa taille de 1 élément.
- Le langage PHP propose également deux types distincts de tableaux:
  - Les tableaux à index numériques
  - Les tableaux associatifs

LES TYPES

6. Les tableaux

- Un tableau vide.

```
// Tableau vide
<?php
  $legumes = array();

  print_r($legumes);
?>
```

tableau.php

LES TYPES

6. Les tableaux

- Un tableau à index numérique avec des **chaines de caractères** *string*.

```
// Tableau indexé numériquement
<?php
  $legumes = array(
        'carotte',
        'poivron',
        'aubergine',
        'chou'
  );
  echo $legumes[0]; // carotte (l'index commence toujours par 0)
?>
```

tableau.php

LES TYPES

6. Les tableaux

- Un tableau numéroté.

```php
<?php
  $fruits = array();

  $fruits[0] = 'Poire';
  $fruits[1] = 'Pomme';
  $fruits[] = 'Raisin'; // Ajoue à la suite
  $fruits[5] = 'Melon';

  echo $fruits[1];                // Affiche 'Pomme'
?>
```

tableau.php

LES TYPES

6. Les tableaux

- Un tableau à index numérique avec des **entiers** *integer*.

```
// Tableau indexé numériquement
<?php
  $nombre = array(
        1, 2, 3, 4, 5
  );
  echo $nombre[3]; // 4
?>
```

tableau.php

LES TYPES

6. Les tableaux

- Un tableau associatif.
- Il prend un nombre illimité de paramètres, chacun séparé par une virgule, sous la forme d'une paire *key => value*.
- On parle alors de tableaux associatifs.

```
// Déclaration d'un tableau associatif
<?php
  $tableau = array(
        'prenom' => 'mickael',    // Chaine de caractères (String)
        12 => true                // Entier (integer)
  );
  echo $tableau['prenom']; // bar
  echo $tableau[12];    // 1
?>
```

LES TYPES

```php
<?php
  // Déclaration d'un tableau associatif
  $identite = array(
      'nom' => 'Durand',
      'prenom' => 'Hugo',
      'age' => 19,
      'estEtudiant' => true
  );

  echo $identite['nom'].' '.$identite['prenom']; // Durand Hugo
?>
```

LES TYPES

- Comparaison de code de tableau.

```php
<?php
  // Ce tableau est identique à
  array(
         5 => 43, 32, 56, // Le premier index est 5
         "b" => 12)
  ;

  // Celui ci
  array(
         5 => 43,
         6 => 32,
         7 => 56,
         "b" => 12)
  ;
?>
```

LES TYPES

- 2eme exemple, on ajoute une variable et on met un print_r();
- Copier/ coller le array dans mon print_r();

```
<?php
  array(
        5 => 43, 32, 56, // Le premier index est 5
        "b" => 12)
  ;

  print_r(array(
        5 => 43, 32, 56, // Le premier index est 5
        "b" => 12)
  );
?>
```

LES TYPES

- Code de tableaux.

```php
<?php

  $arr = array(
         5 => 1,
         12 => 2,
  );

  $arr[] = 56;     // Ceci revient à $arr[13] = 56;

  $arr["x"] = 42;  // Ceci ajoute un nouvel élément avec l'index "x"

  unset($arr[5]);  // Ceci efface un élément du tableau
  unset($arr);     // Ceci efface tout le tableau

?>
```

LES CONVERSIONS DES TYPES

1. **Pour afficher les informations d'une variable**

- PHP.NET var_dump - Affiche les informations d'une variable
  - http://php.net/manual/fr/function.var-dump.php

```
<?php
  $a = 5.5;        // $a contient 5.5 c'est un nombre à virgule (float).
  $b = (int) 8.5; // $b contient 8 c'est un entier (int).

  var_dump($a);           // float(5.5)
  var_dump((int) $a);     // int(5)
  var_dump($b);           // int(8)
?>
```

CONVERSION AUTOMATIQUE

- Exemple de conversion automatique.

Variable	Résultat décimal
$a = 5 + "5";	$a contient 10
$b = 5 + "5abc";	$b contient 10 (supprime le texte)
$c = 026; // nombre octale	$c contient 22
$e = 5 + "1.5";	$e contient 6.5

- Pour tester la conversion d'un nombre octale en décimal: http://www.aly-abbara.com/utilitaires/convertisseur/convertisseur_chiffres.html

LES OPÉRATEURS ARITHMÉTIQUES ET LOGIQUES

LES OPÉRATEURS ARITHMÉTIQUES ET LOGIQUES

Opérateurs	
++, --	Incrémentation / décrémentation
!	Négation
*, /	arithmétique
+, -	arithmétique
.	concaténation des chaînes de caractères
<, <=, >, >=	comparaison
==, !=	égalité et différence
AND ou &&	Et logique
OR ou ||	Ou logique

LES STRUCTURES DE CONTRÔLES

LES STRUCTURES DE CONTRÔLES

1. L'instruction IF

- PHP.NET if - Permet l'exécution conditionnelle d'une partie de code.
  - http://php.net/manual/fr/control-structures.if.php

```php
<?php
  if ($a > $b)
  {
        echo "a est plus grand que b";
  }
?>
```

if.php

LES STRUCTURES DE CONTRÔLES

- Exemple plus complet.

```php
<?php
  $a = 2;
  $b = 1;

if ($a > $b)
  {
        echo $a." est plus grand que ".$b;
  }
?>
```

if.php

LES STRUCTURES DE CONTRÔLES

2. L'instruction ELSE

- PHP.NET else - Si une condition n'est pas remplie.
  - http://php.net/manual/fr/control-structures.if.php
- *else* fonctionne après un *if* et exécute les instructions correspondantes au cas où l'expression du *if* est *FALSE*.

```php
<?php
  if ($a > $b) {
        echo "a est plus grand que b";
  } else {
        echo "a est plus petit que b";
  }
?>
```

else.php

LES STRUCTURES DE CONTRÔLES

3. L'instruction SWITCH

- PHP.NET switch - Équivaut à une série d'instructions if.
  - http://php.net/manual/fr/control-structures.switch.php

```
<?php
  $dept = 75;
  if($dept == 75 || $dept == 'capitale') {
        echo "Paris";
  }
  if($dept == 91) {
        echo "Essonne";}
?>

  // Code page suivante …
```

switch.php

LES STRUCTURES DE CONTRÔLES

```php
$dept = 'capitale';


switch ($dept) {
        case 'capitale':
        case 75:
                echo  "Je suis à paris";
                break;
        case 1:
                echo "i égal 1";
                break;
        case 2:
                echo "i égal 2";
                break;
  }
?>
```

switch.php

- Instruction *switch* utilisant une **chaîne de caractères** *string*.

```php
<?php
  switch ($i) {
        case   "tarte":
                 echo "i est une tarte";
                 break;
        case  "glace":
                 echo "i est une glace";
                 break;
        case  "gateau":
                 echo "i est un gateau";
                 break;
  }
?>
```

switch.php

LES STRUCTURES ITÉRATIVES

LES STRUCTURES ITÉRATIVES

1. La boucle while

- PHP.NET while – Permet d'implémenter une boucle.
  - http://php.net/manual/fr/control-structures.while.php

```php
<?php
  while ($condition) {
  // instructions
}
?>

<?php
  $i = 1;
  while ($i <= 10) { // Tant que $i est inférieur ou égal à 10
  echo $i++;  /* La valeur affichée est $i avant l'incrémentation */
}
```

while.php

LES STRUCTURES ITÉRATIVES

- Second exemple de la boucle *while*.

```
<?php
  $i = 1;
  while ($i <= 10) :
        echo $i;
        $i++;
  endwhile;
?>
```

while2.php

LES STRUCTURES ITÉRATIVES

- Troisième exemple pour la boucle *while*.

```php
<?php
  // Définissez le nombre(numérique) et limites de tables de multiplication
  $num = 11;
  $upperLimit = 10;
  $lowerLimit = 1;

  // La boucle et se multiplie pour créer la table
  while ($lowerLimit <= $upperLimit)
  {
        echo $num." x ".$lowerLimit." = " . ($num * $lowerLimit);
        $lowerLimit++; // 1, 2, 3, 4 ... 10.
        // resultat: 11 x 1 = 11, 11 x 2 = 22 ...
  }
?>
```

while3.php

LES STRUCTURES ITÉRATIVES

2. La boucle do-while

- PHP.NET do-while – Contrairement à *while*, l'expression est testée à la fin de chaque itération plutôt qu'au début.
  - http://php.net/manual/fr/control-structures.do.while.php

- Avec la boucle *do-while* on est sûre que les instructions sont exécutées, au moins, une fois.

```php
<?php
  $i = 0;
  do { // Faire
        echo $i++."<br>";
  } while ($i > 1); // Tant que
?>
```

dowhile.php

LES STRUCTURES ITÉRATIVES

3. La boucle for

- PHP.NET for – Exécution de plusieurs instructions.
  - http://php.net/manual/fr/control-structures.for.php

```php
<?php
  for (expr1; expr2; expr3) {
        // instructions
  }
?>

<?php
  for (initialize counter; conditional test; update counter) {
        // instructions
  }
?>
```

for.php

LES STRUCTURES ITÉRATIVES

- Exemple pour la boucle *for*.

```
// Pour afficher 10 fois 'Ceci est une boucle for en PHP'
<?php
  for ($i=0; $i<10; $i++) { //$i démarre à 0, jusqu'à 9, incrémente de +1
        echo 'Ceci est une boucle for en PHP';
  }
?>

// Pour afficher les chiffres de 5 à 8
<?php
  for ($i=5; $i<9; $i++) { //$i démarre à 5,  jusqu'à 8, incrémente de +1
        echo $i;
  }
?>
```

for.php

LES STRUCTURES ITÉRATIVES

3. La boucle foreach

- PHP.NET foreach – Pour parcourir des tableaux. http://php.net/manual/fr/control-structures.foreach.php
- Fonctionne uniquement pour les tableaux et objets.

```php
<?php
  foreach (array_expression as $value){
        //commandes
  }

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

foreach.php

LES STRUCTURES ITÉRATIVES

- Exemple de boucle Foreach.

```
$lettres = array (
  "un" => 1,
  "deux" => 2,
  "trois" => 3,
  "dix-sept" => 17
);
  foreach ($lettres as $lettre => $chiffre) {
         echo "\$a[$lettre] => $chiffre.\n"; // \n pour un saut de ligne
  }

$chiffres = array(1, 2, 3, 4);
  foreach ($chiffres as $value) {
         echo "$value <br>";
  }
```

LES FONCTIONS

LES FONCTIONS

- PHP.NET les fonctions
  - http://php.net/manual/fr/functions.user-defined.php
- Les fonctions:
  - Permettent de découper le programme en petits blocs réutilisables
  - Augmentent la modularité et la lisibilité du code

```
<?php
  function nom_de_la_fonction($param1, $param2...)
  {
        // Du code
  }
?>
```

foreach.php

LES FONCTIONS

- Exemple de fonction pour afficher le contenu d'un tableau.

```php
<?php
  function print_array($array)
  {
        echo 'Tableau :<br />';

        foreach($array as $key => $value)
        {
                echo '$key => $value <br />';
        }
  }
?>
```

foreach.php

LES FONCTIONS

- Exemple de fonction pour afficher le contenu d'un tableau.

```php
<?php
  function affiche_bjr()
  {
        for ($i = 0; $i < 5; $i++) echo "Bonjour! <br> »;
        //Afficher 5 fois « Bonjour! ».
        //Rq : la fonction ne possède ni arguments ni valeur de retour.
  }
  affiche_bjr(); //Appel à la fonction
?>
```

foreach.php

PASSAGE DE VARIABLE EN $_GET

- Passage de variables:
- On peut passer des variables comme paramètres à une page PHP
- Il suffit de les insérer dans l'adresse de la page
  - http://exemple.com/ma_page.php?variable=valeur

```
// Si on appelle une page: /index.php?nom=alfred&age=12


<?php
      echo $_GET['nom'];       // Affiche "Alfred"
      echo $_GET['age'];       // Affiche "12"

?>
```

- Comment récupérer les variables d'une Url ?
  - Les variables d'une Url sont contenues dans un tableau nommé $_GET

```php
<?php
  $ma_variable = $_GET['ma_variable'];
?>
```

FORMULAIRES

FORMULAIRES

- Les formulaires permettent :
  - Entrer des données
  - De les envoyer au serveur
  - D'afficher des contrôles : zones de textes, menus déroulants, boutons, etc.

FORMULAIRES

- Méthode de conception
  - Les formulaires sont conçus en XHTML
  - Les données sont récupérées et interprétées par PHP

FORMULAIRES

- Contrôles dans une balise <form>
  - Method : type de transmission des données
    - Post : passage par paramètre invisible
    - Get : passage par l'Url (déconseillé)
- Action : fichier cible du formulaire

```
<form method="post" action="save.php »>
  <!-- Contrôles du formulaire -->
</form>
```

FORMULAIRES

- Contrôles communs

```
<input type="text" />                    // ligne de texte (une seule ligne)
<textarea>                               // zone de texte (multilignes)
<input type="submit" />                  // bouton de validation
<select>                                 // menu déroulant

// Contrôles de mise en forme
<fieldset>                               // groupe de contrôles
<legend>                                 // titre du fieldset
```

FORMULAIRES

- Exemple de formulaire :

```
<form method="post" action="form.php”>
  <input type="text" name="nom" value="" />
  <input type="text" name="nom" value="" />
  <input type=“submit" value="Envoyer" />
</form>
```

FORMULAIRES

- PHP – traitement des données
  - Les données envoyées par un formulaire sont contenues dans le tableau $_POST
  - $_POST s'utilise de la même façon que $_GET
  - Si $_POST est vide, le formulaire n'a pas été envoyé

FORMULAIRES

- Méthode classique de traitement des données:

1. On regarde si $_POST contient des données
2. Si oui, on traite les données (affichage, enregistrement...)
3. Si non, on affiche le formulaire

FORMULAIRES

- Exemple de traitement des données.

```
<?php
  if ( !empty($_POST) ) {
        echo $_POST['nom'];
        // Traiter les données
        echo 'Données traitées correctement !';
  } else {
?>
        <form method="post" action="">
                <!-- Contrôles du formulaire -->
        </form>
<?php
        }
?>
```

BASE DE DONNÉES

BASES DE DONNÉES

- Utilité des bases de données
  - Elles permettent de... classer des données
  - Mais surtout de rechercher selon des critères précis
  - Elles peuvent également être mises en relation

BASES DE DONNÉES

- PHP et les base de données
  - PHP comporte de nombreuses fonctions pour communiquer avec une base de données

- base de données communes
  - Une des plus connues est MySQL (libre et gratuit)
  - Autres bases de données: Oracle, SQL Server, Firebird...

BASES DE DONNÉES

- Communiquer avec une BDD
  - Une BDD reçoit des requêtes écrites en langage SQL
  - SQL est proche du langage naturel

- Administrer une BDD
  - On peut administrer une BDD uniquement en envoyant des commandes (comme un FTP sous Linux)
  - Cependant des interfaces rendent cette tâche plus intuitive (comme phpMyAdmin)

BASES DE DONNÉES

- PhpMyAdmin
  - Application écrite en PHP
  - Permet d'administrer facilement une base MySQL

- Accès à PhpMyAdmin
  - En localhost, tapez : http://localhost/phpmyadmin/

BASES DE DONNÉES

- Vocabulaire :
  - Base : « armoire » contenant des tiroirs (les tables)
  - Table : Casier contenant des informations classées – le nom et l'age des visiteurs, par exemple
  - Champs : colonnes d'une table
  - Entrées : lignes d'une table

BASES DE DONNÉES

- Créons une nouvelle table dans PhpMyAdmin :
  - Créer une base nommée 'test’
  - Créer une nouvelle table dans cette base, nommée 'users', avec 3 champs
  - Créer les trois champs : nom, âge, courriel
  - Remplir cette table avec deux ou trois entrées

BASES DE DONNÉES

- Lire des données avec PHP
  - Cela ressemble à peu à la communication avec un FTP
  - Il faut d'abord se connecter, puis lancer sa requête, et récupérer la réponse de MySQL

BASES DE DONNÉES

- Etablir la connexion
  - Avant de communiquer avec MySQL, il faut « attirer son attention »
  - Nous allons démarrer une conversation avec MySQL
  - On utilise pour cela la fonction mysql_connect()

```
<?php
  mysql_connect(
        $serveur,
        $login,
        $mot_de_passe
  );
?>
```

BASES DE DONNÉES

- Explications :
  - 'localhost' est le nom du serveur auquel on se connecte – ici, le serveur SQL est sur notre propre machine.
  - 'root' est le nom de l'utilisateur principal de MySQL
  - '' : par défaut, l'utilisateur root n'a pas de mot de passe

```
<?php
  $serveur = 'localhost';
  $login = 'root';
  $root = '';

  mysql_connect(
        $serveur,
        $login,
        $mot_de_passe
  );
  // ... suite
?>
```

BASES DE DONNÉES

- Que faire une fois connecté ?
  - D'abord, sélectionner la base que l'on veut utiliser
  - On se sert de la fonction mysql_select_db();
- Mysql_close(); pour arrêter la requête
- Si ce code n'affiche rien, tout s'est bien passé

```
<?php
  // ...
  mysql_select_db('test');
        // ici, on peut communiquer avec la BDD
  mysql_close(); // Pour terminer la requête immédiatement
?>
```

BASES DE DONNÉES

- Interroger la BDD
  - On « parle » à MySQL avec la fonction mysql_query()
    mysql_query($requete)
  - $requete est une chaîne de caractère contenant une commande SQL

BASES DE DONNÉES

- Le langage SQL
  - C'est une façon standardisée de communiquer avec les bases de données.
- Exemple de requête SQL :

```
<?php
  // Insérer le code de connexion à la BDD


$reponse = mysql_query('SELECT * FROM users');
?>


// SELECT: la commande à exécuter (une sélection)
// *: les champs que l'on veut récupérer (tous)
// FROM table: le nom de la table dans laquelle on veut chercher
```

BASES DE DONNÉES

- Comment traiter les réponses ?
  - La réponse renvoyée par MySQL est un recordset
  - Un recordset est un ensemble de valeurs brutes
  - On ne peut pas le manipuler directement
  - Il faut donc l'interpréter

BASES DE DONNÉES

- Transformer la requête en tableau
- mysql_fetch_array() transforme chaque ligne du recordset en un tableau associatif

```
<?php
  $ligne1 = mysql_fetch_array($reponse);
  echo $ligne1['nom'];    // Affiche le nom de la 1ère ligne
?>
```

Bases de données

- Boucle sur les résultats
  - mysql_fetch_array() renvoie chaque ligne l'une après l'autre
  - Lorsqu'il n'y a plus de ligne, elle renvoie false
  - On peut donc l'utiliser dans une boucle while() facilement

```
<?php
  $ligne = mysql_fetch_array($reponse);

  while($ligne != false) {
        echo $ligne['nom'];
        $ligne = mysql_fetch_array($reponse);
  }
  // Autre exemple plus concis
  //while($ligne = mysql_fetch_array($reponse)) {
  //     $ligne['nom'];
  //}
?>
```

BASES DE DONNÉES

- Exercice :
  - Récupérer toutes les données de la table 'users’
  - Les afficher sous forme d'un tableau Html

BASES DE DONNÉES

- Sélectionner des données
  - SQL permet de ne sélectionner que les données dont on a besoin, ou de les classer
  - Sélection : WHERE, FROM, LIMIT
  - Classement : ORDER BY

Bases de données

- La clause WHERE
  - Utilisée tout le temps
  - Elle restreint les lignes à la condition donnée
  - Note : il se peut qu'une requête ne renvoie aucune ligne, si la condition n'est remplie par aucune des entrées

```
SELECT * FROM users WHERE age > 16 // Tous les ages supérieur à 16

SELECT * FROM users WHERE nom='Berthe' // Tous les nom Berthe
```

Bases de données

- La clause LIMIT
  - Permet de restreindre le nombre de données renvoyées
  - Peu standard, mais souvent utilisée avec MySQL

```
SELECT * FROM users LIMIT 20 // Les 20 premières données
SELECT * FROM users LIMIT 5, 15 // Les données 5 à 15
```

BASES DE DONNÉES

- La clause ORDER BY
  - Permet de grouper les données reçues
  - On peut éventuellement spécifier un ordre de tri (ascendant ou descendant)
  - Exemples :

```
SELECT * FROM users ORDER BY age
SELECT * FROM users ORDER BY name
ASC
```

  - La première requête groupe les résultats par age
  - La seconde classe les résultats par ordre alphabétique

BASES DE DONNÉES

- Regroupement et combinaisons
  - On peut bien sûr combiner toutes ces clauses
  - Exemple :

```
SELECT *
    FROM users
    WHERE age > 16
        AND prenom = 'Jules'
    ORDER BY nom ASC
    LIMIT 20
```

GESTIONS DES ERREURS

- Gestion des erreurs
  - Lorsque une requête est mal formée ou échoue, MySQL renvoie un message d'erreur
  - On peut y accéder par la fonction mysql_error()
  - mysql_errror() renvoie le dernier message d'erreur généré par MySQL

```
<?php
  $sql = 'INSERT bad_data';
  mysql_query($sql) or die(mysql_error());
?>
```

CRUD

CRUD

- Modification des données
  - Les opérations les plus fréquentes sur les BDD sont résumées par l'acronyme CRUD
    - Create
    - Retrieve
    - Update
    - Delete
  - Certains frameworks pré-implémentent ces opérations
  - Voyons comment les réaliser avec des requêtes SQL

CRUD

- Create : Insertion dans une base de données
  - Pour ajouter une entrée dans une BDD, on utilise la requête INSERT
  - INSERT crée une nouvelle entrée à partir de rien
  - Structure :

```
INSERT
    INTO table(liste_des_champs)
    VALUES(liste_des_valeurs)
```

  - La liste des champs est optionnelle

CRUD

- Insertion dans une base de données
  - Exemple d'insertion :

```
<?php

    $sql = "INSERT INTO users(prenom, age)
VALUES              ('Paul', 51) ";

    mysql_query($sql) or die(mysql_error());

?>
```

CRUD

- Update : Mise à jour
  - Pour mettre à jour une entrée existante, on utilise la commande UPDATE

```
UPDATE
    table
    SET field = value
    WHERE condition
```

CRUD

- Mise à jour
  - Exemple : on veut changer l'âge d'un utilisateur

```
<?php

    $sql = " UPDATE users SET age = 21 WHERE
       name = 'Paul' ";

    mysql_query($sql) or die(mysql_error());

?>
```

CRUD

- Delete : Supression
  - Pour supprimer une entrée, on utilise la commande DELETE

```
DELETE
    FROM table
    WHERE condition
```

CRUD

- Suppression
  - Exemple : on veut supprimer des utilisateurs

```
<?php

   $sql = " DELETE FROM users WHERE name =
'Paul' ";
   mysql_query($sql) or die(mysql_error());

?>
```

  - Autre exemple :

```
   $sql = " DELETE FROM users WHERE age < 16 "
```

CRUD

- Suppression
  - Exemple : on veut supprimer des utilisateurs

```
<?php

   $sql = " DELETE FROM users WHERE name =
'Paul' ";
   mysql_query($sql) or die(mysql_error());

?>
```

  - Autre exemple :

```
   $sql = " DELETE FROM users WHERE age < 16 "
```

CRUD

- TP : Réalisation
  - Quatre pages PHP principales : index.php, add.php, edit.php, delete.php
  - Un fichier de connexion à la BDD, bdd.php, inclus au début de toutes les pages principales
  - Vous êtes libres du reste :)

SESSIONS

sessions

- Les sessions
  - Permettent de conserver des données entre les pages
  - Gérés automatiquement par PHP
- Utilisation des sessions
  - Initialiser avec session_start()
  - Utiliser le tableau $_SESSION

```php
<?php
   session_start();
   echo 'Bienvenue à la page numéro 1';

   $_SESSION['favcolor'] = 'green';
   $_SESSION['animal']   = 'cat';
   $_SESSION['time']     = time();

   // Indique l'identifiant de session
   echo '<br /><a href="page2.php?' . SID . '">page 2</a>';
?>
```

SÉCURITÉ

SÉCURITÉ

- Sécurité
  - La question de la sécurité est primordiale
  - Dès que l'on interagit avec l'extérieur, on s'expose à des problèmes de sécurité :
    - Exécution de requêtes SQL malicieuses
    - Vol de données
    - Exécutions de commandes malicieuses sur le serveur
    - Vol d'identifiants (login/mot de passe)
    - Accès frauduleux au site

SÉCURITÉ

- Principe fondamental :
  - NE JAMAIS FAIRE CONFIANCE !
- Il faut toujours vérifier les données provenant de l'extérieur

SÉCURITÉ

- Principaux vecteurs d'attaques :
  - Injections de script
  - Injections SQL
  - Injection de HTML
  - Cross-Site Scripting (XSS)

SÉCURITÉ

- Injection de script :
  - Tentative d'injecter des variables (par l'Url : méthode GET)
  - Tentative d'exécuter des scripts sur le serveur (faille d'include)
- Pour se protéger :
  - Toujours initialiser les variables utilisées
  - Vérifier les variables avant de faire des include

SÉCURITÉ

- Injection SQL
  - Tentative d'injecter des bouts de requête SQL
  - On peut virtuellement exécuter n'importe quelle requête
- Exemple :
  - SELECT * FROM users WHERE login = '$name'
  - Si $name vaut :

    ' OR '1' = '1

    Tout nom d'utilisateur renverra une réponse valide.
- On pourrait ainsi récupérer des mots de passe, ou même effacer la base de données :

  '; DELETE FROM users WHERE '1' = '1

SÉCURITÉ

- Injection SQL : pour se protéger :
  - Utiliser mysql_real_escape_string()
  - Cette fonction échappe les guillemets malicieux : elle transforme les ' en \'
- Attention à magic_quotes_gpc
  - Option de php.ini permettant d'échapper automatiquement les chaînes
  - Si elle est activée, il ne faut pas échapper une seconde fois la chaîne !
  - On utilise alors stripslashes() avant mysql_real_escape_string()

SÉCURITÉ

- Injection HTML
  - Si l'utilisateur peut rentrer du texte affiché ensuite sur une page, il pourrait y mettre de l'Html
- Exemple :
  - Dans une news, on pourrait insérer :
    Voici ma news <!--
  - Le début de commentaire HTML briserait la maquette de la page lors de l'affichage
  - On pourrait aussi insérer du code Javascript malicieux

SÉCURITÉ

- Injection HTML : pour se protéger :
  - Utiliser la fonction htmlentities() avant d'afficher du texte
  - Elle transforme les balises Html en texte affichable
  - Le Html est alors affiché comme du texte brut

SÉCURITÉ

- Cross-Site Scripting (XSS)
  - Injection de Javascript renvoyant sur une autre page
  - Le Javascript peut voler des cookies ou des mots de passe, et les envoyer à une page Web distante

SÉCURITÉ

- XSS - Exemple :
  - On appelle la page :

    add.php?nom="/
    ><script>window.location='hack.php'</script><br
    &age=valeur_invalide

  - Cette page pourrait afficher un message d'erreur et le formulaire prérempli :

    <input name="nom" value="Nom" />

    deviendrait alors :

    <input name="nom" value="" />
    <script>....</script>

SÉCURITÉ

- XSS – Protection :
  - Vérifier que l'on affiche jamais de variable POST ou GET sans vérification préalable

MISE EN LIGNE

MISE EN LIGNE

- Comment mettre en ligne un site PHP ?
  - Il faut s'assurer que l'hébergeur dispose de PHP et d'une base MySQL (ce qui est fréquent)
  - Vérifier la version de PHP
    - PHP4 est normalement compatible avec PHP5
    - PHP5 n'est pas compatible avec PHP4
  - Vérifier dans la doc que les fonctions qu'on utilise sont compatibles avec la version de PHP du serveur
  - Vérifier les variables de configuration de PHP

MISE EN LIGNE

- Variables de configuration
  - PHP peut être configuré à l'aide du fichier php.ini
  - Il faut coder en gardant à l'idée que la configuration du serveur peut être différente
- Exemple :
  - magic_quotes_gpc (souvent à On)
    - Echappe automatiquement les chaînes
  - short_tags (par défaut à Off)
    - Autorise <? ?> au lieu de <?php ?>
  - register_globals (souvent à Off)
    - Place $_POST['clefs'] dans $clefs - dangereux

MISE EN LIGNE

- Déploiement
  - Le déploiement est souvent source de problèmes
  - Si possible, tester au fur et à mesure que l'on code
  - Prévoir du temps pour le déploiement

CONCLUSION

CONCLUSION

- À découvrir
  - Les Sessions en PHP
  - Les systèmes d'authentification (login)
  - Le PHP Objet
  - Les fonctionnalités de PHP5
  - Interactions AJAX/PHP

- Les Frameworks
  - Comment développer rapidement
  - Automatiser les tâches répétitives (comme les CRUD)
  - Formation aux Frameworks et PHP avancé en février