Table des matières

Ressources

Il existe de très nombreux sites dédiés au PHP ! Il faut au moins le manuel (notamment pour les fonctions) : http://www.php.net/manual/fr/index.php et http://nexen.net/index.php

Le Langage PHP

Introduction

Le langage PHP a été créé par Rasmus Lerdorf en 1994 pour des besoins personnels (***P**ersonnal **H**ome **P**age*)

En 1997, le projet devient un travail d'équipe et l'interpréteur est réécrit par Zeev Suraski et Andi Gutmans pour donner la version PHP3, version qui s'est rapidement imposée et devient PHP (***H**ypertext **P**re**P**rocessor*).

La version PHP 4 date de 2000. Elle intègre en mode natif le moteur Zend (société privée créée par Suraskyi et Gutmans). PHP4 s'avère plus rapide, plus fiable et plus complet. Les scripts sont désormais compilés puis exécutés.

La dernière version en date est PHP5 (2004) et représente une évolution majeure : **nouveau modèle objet, support d'XML (*eXtended Markup Language*) et des services web, base de données locale, ...**

PHP est utilisé par plus d'un site web sur trois dans le monde. Environ 500.000 entreprises l'utiliseraient en 2003 selon l'AFUP (Association Française des Utilisateurs de PHP). En France particulièrement, 78% des entreprises du CAC 40 ont recourt à cette technologie, la plupart pour leur site web uniquement.

Caractéristiques

Les principales caractéristiques sont :

- Langage de script côté serveur
- Embarqué dans les pages HTML
- Syntaxe héritée du C et du Perl
- Extensible (nombreuses bibliothèques et fonctions) ;

Remarque:
PHP est distribué sous licence libre (licence BSD pour PHP5).

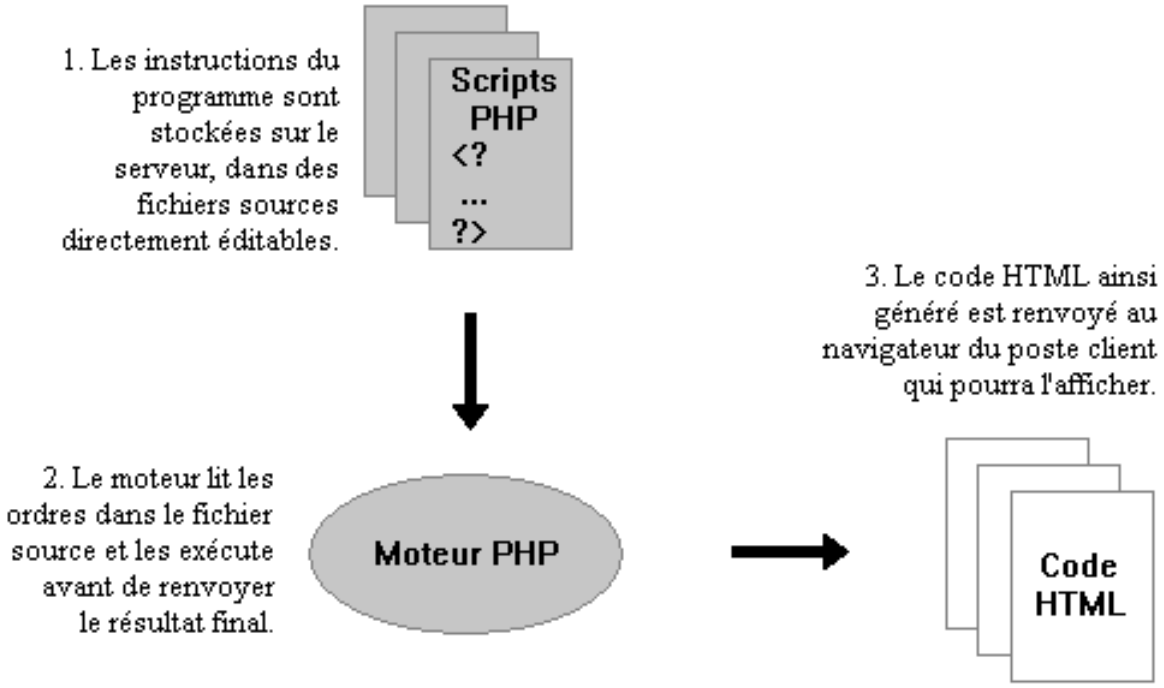
Versions

On distingue 3 versions majeures pour PHP :

- A l'heure actuelle, PHP 3 n'est pratiquement plus utilisé (à part pour des solutions de serveur web embarqué)
- PHP 4 est encore la version la plus utilisée notamment par les hébergeurs de site web (essentiellement pour des raisons de stabilité/sécurité et de migration lourde)
- PHP 5 est la toute dernière version et la plus prisée pour les entreprises qui désirent migrer ou démarrer avec ce type de technologie

Fonctionnement

Un script PHP est un simple fichier texte ASCII contenant des instructions incluses dans du code HTML à l'aide de balises spéciales et stocké sur un serveur HTTP disposant d'un interpréteur PHP.



Ce fichier script doit avoir une extension reconnue par le serveur (le plus souvent .php ou tout autre extension défini sur le serveur).

En résumé:
Pour qu'un script PHP soit interprété par le serveur, les conditions nécessaires sont :

- Le fichier est un simple **fichier texte ASCII**
- Le fichier contenant le code doit avoir la bonne extension (**.php** et non *.html*)
- Le code php contenu doit être délimité par les balises **<?php et ?>**

Pour des raisons de conformité avec certaines normes (XML par exemple), plusieurs balises peuvent être utilisées pour délimiter un code PHP :

```
1.  <?php et ?>
2.  <? et ?>
3.  <script language="php"> et </script>
4.  <%php et %>
```

Remarque: il est également possible d'utiliser les balises <?= ... ?> qui produise un affichage direct, par exemple <?= $message ?> est équivalent à <?php echo $message; ?>

Exemple

Script **`hello.php`** (2 versions possibles):

```
<html>
     <head>
      <title>Exemple</title>
     </head>
     <body>
      <?php
          echo "Hello world";
      ?>
     </body>
</html>
```

```
<?php
echo "<html>
     <head>
      <title>Exemple</title>
     </head>
     <body>";
echo "Hello world";
echo "</body>
</html>";
?>
```

Le client utilise un navigateur et l'URL : http://localhost/hello.php (en local) ou http://adresse_IP/hello.php et obtient : Hello world

Si maintenant on regarde le source de la page Web côté client, on y lit :

```
<html>
 <head>
  <title>Exemple</title>
 </head>
 <body>
   Hello world
 </body>
</html>
```

Remarques:

La syntaxe de PHP est directement héritée du langage C et du perl, par exemple :
- le séparateur d'instructions est le `;`
- les commentaires :

```
/* ...mes commentaires... */
// ...mes commentaires...,
# ...mes commentaires....
```

Les variables

Le langage PHP supporte les types de données suivants :

- Scalaires (entier, flottant, chaînes de caractères)
- Tableaux et tableaux associatifs
- Objets

Tous les noms de variable sont précédés d'un **$**.

Comme pour la plupart des langages de scripts, il n'est pas obligatoire de déclarer les variables et il suffit de leur assigner une valeur pour en définir le type :

- `nbr = 10; //un entier`
- `fl = 3.141; //un réel`
- `str1 = "L'étoile"; //une chaîne de caractère`
- `str2 = 'brille'; //une chaîne de caractère`

Remarque: il existe aussi le type booléen (*true* ou *false*)

Les tableaux stockent des données sous forme de liste. Les données contenues dans la liste sont accessibles grâce à une **clé** (ou **index**, indifféremment un **entier** ou une **chaîne de caractères**).

Contrairement à des langages tels que le C, il est possible de stocker des éléments de types différents dans un même tableau.

Pour créer un tableau, on peut utiliser :

- la fonction **`array()`** ;
- affecter directement les valeurs au tableau.

```
// tableau simple :
$tableau[0] = 2002;
$tableau[1] = "BTS IRIS";
$tableau[] = 11.12;// donc en [2]

// tableau à 2 dimensions :
$tab[0][0] = 12;

// tableau associatif :
$toto["Age"] = 12;

// avec array() :
$prenom = array('robert',‘roger',‘germain',‘fernand');
echo $prenom[0];

$animaux = array(1 => 'chien','chat','vache','cochon');
echo $animaux[1];
```

La portée d'une variable dépend du contexte dans lequel elle est définie.

On distinguera les variables à portée :

- **Globale** (lorsqu'une variable est déclarée à l'extérieur de toute fonction ou de tout bloc d'instruction, elle est accessible (visible) de partout dans ce code)
- **Locale** (Lorsque une variable est déclarée à l'intérieur d'un bloc d'instructions ou d'une fonction, sa portée est alors locale à ce bloc ou cette fonction)

Remarque : les noms de variable sont sensibles à la casse (mais pas les noms de fonction !).

Les fonctions

Définition :

```
<?php
function nomfonction($param1, ..., $paramN)
{
          // code PHP de la fonction
          return $variable_ou_valeur ; // facultatif
}
?>
```

Explication :

- L'instruction `function` qui déclare une fonction.
- Le nom de la fonction `nomfonction`. (non sensible à la casse !)
- (...) La liste des paramètres séparés par une virgule, les arguments sont facultatifs, tout dépend du but de votre fonction.
- Une instruction `return` facultative qui permet de renvoyer le contenu d'une variable ou une valeur ... Si l'instruction `return` est absente, la fonction est une procédure.

Appel :

```
$res = nomfonction($var1, $val2, $varN);
```

Remarque : ici passage des paramètres par valeur

Paramètres par valeur ou par référence :

Les paramètres d'une fonction peuvent être passés de 2 façons différentes :

- Par **valeur**, seule la valeur est transmise à la fonction, si la variable subit des modifications à l'intérieur de la fonction, ces modifications ne seront pas perçues dans le programme principal.

- Par **référence**, avec le signe **&** avant la variable (ex : `&$var1`), l'adresse mémoire de la variable est passée à la fonction et toute modification de cette variable dans la fonction aura des répercutions à l'extérieur du programme.

Remarque :

Le PHP autorise de déclarer des valeurs par défauts pour les paramètres : function isHumanAge($age=0) {...}

Exemple :

```
function isHumanAge($age)
{
          if (($age < 0) || ($age > 120)) { return false  ; }
          else { return true  ; }
}
```

```
$age = 133;
if(!isHumanAge($age))
{
          echo("Vous ne pouvez pas avoir " . $age . " ans !");
}
```

Remarque: avec les chaînes de caractères, l'opérateur **.** réalise la **concaténation**

Variables globales :
Les variables globales déclarées dans un script sont visibles sur l'ensemble du script, mais pas directement au sein des fonctions. Pour utiliser une variable globale au sein d'une fonction, on doit le préciser à l'aide de l'instruction global, pour faire référence à la variable globale du même nom.
On peut accéder à cette variable par son nom ou à l'aide du tableau **`$GLOBALS`**.

Remarque : il existe aussi des variables super globales ! (voir plus loin)

Rappel : les variables déclarées et utilisées au sein d'une fonction sont locales à la fonction.

```
<?php
function foo()
{
 global $cpt; // permet l'accès à la variable globale cpt
 $cpt++; //ou directement avec $GLOBALS["cpt"]
}

$cpt = 0;
foo();
echo $cpt; // affiche 1
?>
```

Variables `static` :
Une variable déclarée à l'intérieur d'une fonction à l'aide de l'instruction `static` permet à une variable de garder sa valeur à chaque appel de la fonction. L'initialisation d'une variable *static* se fait au début de la fonction et à chaque appel de la fonction dans le script elle gardera la valeur du dernier appel.

Les constantes :
Les constantes sont définies grâce à la fonction **`define()`,** dont la syntaxe est la suivante :

```
          define("MA_CONSTANTE", "Bonjour") ;
          echo MA_CONSTANTE ; # affiche Bonjour (et ne pas mettre de $)
```

On conseille de toujours utiliser des majuscules pour les noms de constante.

Opérateurs et Structures de contrôle

Les opérateurs et structures de contrôle sont identiques au langage C.

Seule particularité :

PHP4 définit une boucle **foreach**, comme en Perl, pour réaliser une boucle sur les éléments d'un tableau.

En PHP3 on peut réaliser l'équivalent avec une boucle **`while`** et les fonctions **`list()`** et **`each()`.**

Exemple :

```
foreach ($tableau as $cle => $valeur)
{
          echo "$cle => $valeur, " ;
}
```

L'affichage

PHP fournit plusieurs fonctions permettant d'envoyer du texte au navigateur :

`echo, print` et **`printf`**

Après traitement par le moteur PHP, le flux est envoyé via HTTP vers le client :
`echo "Bienvenue $login !";` // c'est ce qui rend possible la création de pages dynamiques

Remarques :
L'insertion de code HTML dans des scripts PHP posent régulièrement des problèmes au programmeur en terme de cohérence, maintenance et portabilité. On cherche donc à séparer l'affichage (HTML) de la partie programmation (PHP), vu que d'autre part ce ne sont le plus souvent pas les mêmes personnes qui créent ces différentes parties (designer/développeur).
Les solutions les plus utilisées sont : encapsulation des fonctions d'affichage dans des classes/fonctions ou utilisation des *templates (*par exemple *smarty)*.

Pour un déboggage simple, on utilise souvent la fonction `var_dump($mavariable)` pour afficher des informations structurées (type, valeur) sur une variable (voir aussi `print_r, var_export,` ...)

Ressources

Il existe de très nombreux sites dédiés au PHP ! Il faut au moins un accès au manuel (notamment pour les fonctions) : http://www.php.net/manual/fr/index.php et http://nexen.net/index.php

Séquence 1 : les bases

Objectifs

S'initier à la programmation de base du langage PHP en manipulant des variables, structures de contrôle et fonctions.

Moyens disponibles

Le logiciel ***Qanta+*** ou l'éditeur de texte ***vim*** pour la réalisation de la page HTML/PHP ou un autre éditeur
Un navigateur (***Konqueror***, ***Mozilla***, ***Firefox*** ou ***Opéra***) et un serveur Web (***Apache***)
Le manuel sur les fonctions PHP

Remarques

Pour réaliser un développement PHP, il vous faut la chaîne complète **client/serveur**. Plusieurs solutions s'offrent à vous :

① le serveur est présent sur votre machine de développement (*localhost*). Le plus souvent sous Linux, la racine des documents se trouvent en **/var/www/html/** . Et l'accès par le navigateur se fait à l'adresse: http://localhost/ ou http://votre_adresse_ip/
② le serveur est présent sur l'intranet de votre structure de développement (entreprise, école, université, communauté, ...). Le serveur de la section est configuré pour un accès pour chaque compte. La racine se trouve dans votre répertoire personnel **$HOME/public_html/** et l'accès client se fait à l'adresse: http://192.168.52.83/~$LOGIN/
③ le serveur est présent sur l'internet le plus souvent chez un hébergeur. Dans ce cas, il faut transférer les documents de votre poste de développement vers le serveur Internet.

Pour le TP, vous utiliserez la solution 1 ou 2.

Manipulations

1 . Ecrire votre premier script PHP **manip_s1_1.php** et vérifier que celui-ci fonctionne ce qui permet de vérifier que la chaîne client/serveur fonctionne :

```
<html>
     <head>
       <title>Manipulation 1.1</title>
     </head>
     <body>
      <?php
           echo "Hello world !";
      ?>
     </body>
</html>
```

Avant de continuer, il est intéressant de tester aussi la fonction **`phpinfo()`** qui affiche un tas d'informations importantes, soit le script **manip_s1_info.php** :

```
<html>
```

```
     <head>
       <title>Manipulation 1.0</title>
     </head>
     <body>
      <?php
           phpinfo();
      ?>
     </body>
</html>
```

PHP offre un certain nombre de variables d'environnement dites **super globales** qui contiennent des informations importantes et souvent utilises, par exemple **manip_s1_0.php** :

```
<html>
     <head>
       <title>Manipulation 1.0</title>
     </head>
     <body>
      <?php
       echo "<pre>"; print_r($_SERVER); echo "</pre>";

       echo "Adresse IP du serveur: ";
       echo $_SERVER["SERVER_ADDR"];
       ?>
     </body>
</html>
```


2 . Ecrire un script **manip_s1_2.php** qui utilise une fonction PHP et manipule des variables.

```
<html>
     <head>
       <title>Manipulation 1.2</title>
     </head>
     <body>
      <?php
      $date = date("l j F Y");
      $heure = date("H:i:s");
      $message = "Bonjour,<br />on est le ".$date." et il est ".$heure."<br />";
      echo $message;
      ?>
     </body>
</html>
```

Remarque: notez l'utilisation de l'opérateur **.** qui permet la concaténation.

3 . Ecrire un script **manip_s1_3.php** qui permet d'afficher un message personnalisé et qui utilise la structure conditionnelle **if**. Ce script montre tout simplement que la langage PHP permet d'écrire des pages à contenu dynamique.

```
<html>
     <head>
       <title>Manipulation 1.3</title>
     </head>
     <body>
      <?php
      $heure = date("H"); // pour les tests modifier manuellement $heure
      if($heure >= 18)
            $message = "Bonsoir,<br />";
      else  $message = "Bonjour,<br />";
      echo $message;
      ?>
     </body>
</html>
```

Bilan n°1: écrire un script **bilan1.php** qui permet d'afficher la saison actuelle soit **"**C'est le printemps !**"** ou **"**C'est l'hiver !**"** ou **"**C'est l'automne !**"** ou **"**C'est l'été !**"**. Tester.

Remarque: On prendra les dates suivantes pour 2007
Printemps: 21 mars 2007 – Eté: 21 juin 2007 – Automne: 23 septembre 2007 – Hiver: 22 décembre 2007

Pour obtenir les dates des saisons:
http://www.imcce.fr/fr/ephemerides/astronomie/Promenade/pages4/439.html

4 . Ecrire un script **manip_s1_4.php** qui reprend le script manip_s1_2.php mais qui donne le résultat en français en utilisant les variables tableaux.

```
<html><head><title>Manipulation 1.4</title></head><body>
```

```
<?php
$jour = array('dimanche', 'lundi', 'mardi', 'mercredi', 'jeudi',
 'vendredi', 'samedi'); // 'dimanche' est l'index 0 de ce tableau

$mois = array();
$mois[1] = 'janvier';
$mois[2] = 'février';
$mois[3] = 'mars';
$mois[4] = 'avril';
$mois[5] = 'mai';
$mois[6] = 'juin';
$mois[7] = 'juillet';
$mois[8] = 'août';
$mois[9] = 'septembre';
$mois[10] = 'octobre';
$mois[11] = 'novembre';
$mois[12] = 'décembre';

$js = date("w"); $j = date("j"); $m = date("n");
$message = ucfirst($jour[$js])." ".$j." ".$mois[$m]." ".date("Y")."<br />";
echo $message;
?>
</body></html>
```

Remarque: ici l'intérêt était de montrer l'utilisation des variables tableaux sinon pour formater des dates dans d'autres langues, utilisez les fonctions **setlocale()** et **strftime()**.

5 . Ecrire le script **manip_s1_5.php** qui fait la même chose mais en utilisant un tableau associatif.

```
<html><head><title>Manipulation 1.5</title></head><body>
<?php
$date["jour"] = array('dimanche', 'lundi', 'mardi', 'mercredi', 'jeudi',
'vendredi', 'samedi');
$date["mois"] = array('janvier', 'février', 'mars', 'avril', 'mai', 'juin',
'juillet','aout','septembre','octobre','novembre','décembre');

$js = date("w"); $j = date("j"); $m = date("n");
$message = ucfirst($date["jour"][$js])." ".$j." ".$date["mois"][$m-1]."
".date("Y")."<br /><br />";
echo $message;

// Affichage de debuggage (très utile)
echo "<pre>"; print_r($date); echo "</pre>";
//echo "<pre>"; var_dump($date); echo "</pre>";
?>
</body></html>
```

6 . Ecrire un script **manip_s1_6.php** qui affiche le contenu d'un tableau (normal et associatif)

```
<html><head><title>Manipulation 1.6</title></head><body>
      <?php
    $date["jour"] = array('dimanche', 'lundi', 'mardi', 'mercredi', 'jeudi',
'vendredi', 'samedi');
    $date["mois"] = array('janvier', 'février', 'mars', 'avril', 'mai', 'juin',
'juillet','aout','septembre','octobre','novembre','décembre');
    echo "Le tableau associatif \$date:<br /><br />";
      foreach ($date as $cle => $valeur)
      {  for($i=0; $i<count($valeur);$i++)
         {
            if(!Empty($valeur[$i]))
               echo "\$".$cle."[".$i."] => ".$valeur[$i]."<br />";
         }
         echo "<br />";
      }
      ?>
     </body></html>
```

Remarques: pour un tableau associatif, on utilise la boucle **foreach** et pour un tableau normal on utilise une simple boucle **for**.

7 . Ecrire un script **manip_s1_7.php** qui détecte si l'année en cours est bissextile ou non en utilisant une fonction.

```
<html><head><title>Manipulation 1.7</title></head><body>
      <?php
    function estAnneeBissextile($annee)
      {
         $estMultipleDeQuatreCent = ( ($annee % 400) == 0);
         $estMultipleDeQuatre = ( ($annee % 4) == 0);
         $estPasMultipleDeCent = ( ($annee % 100) != 0);
         return ( $estMultipleDeQuatreCent || ( $estMultipleDeQuatre &&
$estPasMultipleDeCent ) );
      }
      $y = date("Y"); // pour les tests modifier manuellement $y
      if(estAnneeBissextile($y))
            $message = $y." est une année bissextile !<br />";
      else  $message = $y." n'est pas une année bissextile !<br />";
      echo $message;
      ?>
     </body></html>
```

Remarque: la fonction estAnneeBissextile a été remaniée selon les bonnes pratiques de programmation. Elle est lisible et pourtant elle ne comporte aucun commentaire.

Bilan n°2: écrire un script **bilan2.php** qui permet d'afficher le nombre d'années bissextiles que vous avez vécu depuis votre naissance en indiquant lesquelles. Exemple de résultat :

```
Vous avez vécu 10 années bissextiles : 1968 1972 1976 1980 1984 1988 1992 1996 2000 2004
```

Séquence 2 : passage de paramètres dans l'URL

Objectifs

Cette séquence a pour objectif de montrer comment on récupère des données passées en paramètres de l'url. Cette technique est très utilisée dans la réalisation de scripts PHP.

Remarques

Jusqu'à la version PHP 4.2.0, les paramètres passés dans l'URL étaient automatiquement connus sous forme de variables globales du script destinataire. Mais depuis la version 4.2.0, ce n'est plus le cas à cause du changement de la valeur par défaut (auparavant à On et désormais à Off) du paramètre **`register_globals`** du fichier de configuration php.ini du serveur. Ce changement impose de recourir désormais aux tableaux dit **super globaux** de PHP (`$_GET[], $_POST[]`, etc ...). Ces variables super globales sont accessibles de partout dans un script php (ne pas mettre `global`) . Exemple :

exemple.php?id=4 alors **`$_GET['id']`** sera égal à 4

Manipulations

1 . Ecrire un script **manip_s2_1.php** qui détecte si l'année passée en paramètre de l'url est bissextile ou non.

```
<html><head><title>Manipulation 2.1</title></head><body>
      <?php
      function estAnneeBissextile($annee)
      {
         $estMultipleDeQuatreCent = ( ($annee % 400) == 0);
         $estMultipleDeQuatre = ( ($annee % 4) == 0);
         $estPasMultipleDeCent = ( ($annee % 100) != 0);
         return ( $estMultipleDeQuatreCent || ( $estMultipleDeQuatre &&
$estPasMultipleDeCent ) );
      }
      //récupère le paramètre y
      $y = $_GET["y"];
      //teste le paramètre y
      if(!Empty($y))
      {  if(estAnneeBissextile($y))
            $message = $y." est une année bissextile !<br /><br />";
         else  $message = $y." n'est pas une année bissextile !<br /><br />";
         echo $message;
      }
      else
      {
         echo "Paramètre y manquant !<br /><br />";
      }
      echo "Essayez :<br /><br />";
      echo "<a href=\"manip_s2_1.php?y=2000\">2000</a><br />";
      echo "<a href=\"manip_s2_1.php?y=2007\">2007</a><br />";
      ?>
    </body></html>
```

2 . Ecrire un script **manip_s2_2.php** qui récupère l'ensemble des paramètres passées à l'url et qui les affiche. Le passage de données dans l'url peut être délicat et nécessite d'utiliser des fonctions d'encodage et de décodage. Ce script montre un exemple d'utilisation des ces fonctions.

```
<html><head><title>Manipulation 2.2</title></head><body>
      <?php
      //echo "<pre>"; print_r($_SERVER["QUERY_STRING"]); echo "</pre>";
      $params = explode ('&', $_SERVER["QUERY_STRING"]);
      if(!Empty($params))
      {
         $i = 0;
         while ($i < count ($params))
         {
            $param = split ('=', $params[$i]);
          echo 'La valeur du paramètre ', htmlspecialchars(urldecode($param[0])),
               ' est ', htmlspecialchars(urldecode($param[1])), "<br  />\n";
            $i++;
         }
      }
      else
      {
         echo "Aucun paramètre reçu dans cette url !<br />";
      }
      echo "<br />Essayez :<br /><br />";
      echo "<a href=\"manip_s2_2.php\">manip_s2_2.php</a><br />";
      echo "<a href=\"manip_s2_2.php?y=2007\">manip_s2_2.php?y=2007</a><br />";
      echo "<a href=\"manip_s2_2.php?op=voir&y=2000\">manip_s2_2.php?
op=voir&y=2000</a><br />";
      $userinput = "un message avec des caractères à encoder !";
      $message = htmlentities(urlencode($userinput));
      echo "<a href=\"manip_s2_2.php?message=".$message."\">manip_s2_2.php?
message=".$userinput."</a><br />";
      ?>
     </body></html>
```

Bilan n°3: écrire un script **bilan3.php** qui affiche toutes les années depuis votre naissance sous forme de lien avec en paramètre l'année en question et qui permet de déterminer si cette année passée en paramètre est bissextile ou non. Exemple d'affichage :

2000 est une année bissextile !

1966 1967 1968 1969 1970 1971 1972 1973 1974 1975
1976 1977 1978 1979 1980 1981 1982 1983 1984 1985
1986 1987 1988 1989 1990 1991 1992 1993 1994 1995
1996 1997 1998 1999 2000 2001 2002 2003 2004 2005
2006 2007

Remarque: les lignes comportent seulement dix années

Séquence 3 : traitement de formulaire

Objectifs

Cette séquence a pour objectif de montrer comment on récupère des données envoyées par un formulaire. Cette technique est très utilisée dans la réalisation de scripts PHP.

Remarques

Jusqu'à la version PHP 4.2.0, les paramètres passés dans l'URL étaient automatiquement connus sous forme de variables globales du script destinataire. Mais depuis la version 4.2.0, ce n'est plus le cas à cause du changement de la valeur par défaut (auparavant à On et désormais à Off) du paramètre **`register_globals`** du fichier de configuration php.ini du serveur. Ce changement impose de recourir désormais aux tableaux dit **super globaux** de PHP (`$_GET[], $_POST[]`, etc ...). Ces variables super globales sont accessibles de partout dans un script php (ne pas mettre `global`). Par exemple:

```
<form action="" method="POST" name="form">
          <input type="hidden" name="id" value="4">
          <input type="submit" value="Envoyer">
</form>
```

Alors **`$_POST['id']`** sera égal à 4 . En phase d'apprentissage ou de débuggage, il est recommandé de faire un **`var_dump($_POST);`**

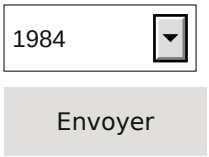
Manipulation

1 . Ecrire un script **manip_s3_1.php** qui formate un message de bienvenue en fonction du nom saisi dans le formulaire.

```
<html><head><title>Manipulation 3.1</title></head><body>
      <?php
       $nom = $_POST["nom"];
       if(!Empty($nom))
       {    $heure = date("H");
            if($heure >= 18)
              $message = "Bonsoir $nom,<br />";
            else  $message = "Bonjour $nom,<br />";
            echo $message;
       }
       else
       {  echo "Paramètre nom manquant !<br />";  }
       echo "<br />Essayez :<br /><br />";
     ?>
     <form action="" method="POST" name="form">
          <input type="text" name="nom" value="">
          <input type="submit" value="Envoyer">
     </form>
     </body></html>
```

Bilan n°4: écrire un script **bilan4.php** qui affiche si l'année envoyée dans le formulaire bissextile ou non. On utilisera une liste déroulante. Exemple :

1984 est une année bissextile !



Séquence 4 : conserver des données côté serveur

Objectifs

On a souvent besoin de lire, modifier ou écrire des données côté serveur. Il existe plusieurs mécanismes en PHP pour faire cela, voici les principaux :

- les sessions
- les fichiers
- les bases de données