Un peu plus loin avec PHP

Le problème de la continuité
le serveur web ferme la connexion
Qui est connecté ?
Quels sont les droits de l'utilisateur ?
Qu'a t'il déjà fait ?
Comment passer des informations de pages en pages ?
Comment assurer une navigation suivie ?

Les passages de variables entre pages

Solution formulaire (GET et POST) : les variables sont accessibles dans la page *cible* sous la forme d'un tableau global associatif ($_GET ou $_POST) (mais toutes les pages ne sont pas des formulaires).

Solution du lien href (en fait, on utilise la forme du GET) : On fabrique une *URL* qui contient les noms des variables et leur contenu, comme le fait le navigateur avec un formulaire GET : page.php?nom=val&nom2=val2&nom3=val3

(attention aux caractères sensibles : espace= %20)

Passage de Variables

Solution formulaire Champ caché : les variables sont stockés dans un formulaire dans un champ *hidden*. Grâce aux attributs *value* et *name*, on peut attribuer **une valeur et un nom** à ce champ caché, qui sera récupèré dans la page cible, comme n'importe quel autre champ... (faille de sécurité énorme cependant !Car modification possible par l'utilisateur un peu documenté)

Les Cookies

Le serveur web est *STATELESS* : Ferme la connexion aprés avoir servi la page (c'est un peu faux maintenant, mais ça ne change rien)

Idée d'origine pour suivre la connexion : le **cookie**

Le **cookie** porte un **nom**, et est enregistré sur le client. On y stocke des informations.

On peut en PHP **générer** un cookie, qu'on pourra ensuite **rappeler**, afin de savoir par exemple si l'utilisateur est déjà venu. Les cookies seront accessibles dans un tableau associatif en PHP, $_**COOKIE**[]

Exemple de cookie

```php
<?php
setCookie('GNU-Association','Nouvel adhérent',
time()+12*3600);
?>
```

Ici, je définie un cookie GNU-Association qui contient la valeur (nouvel adhérent). La validité de ce cookie est de 12 heures. (nom, contenu, validité)

Puis, plus tard, je tente de l'afficher..

```php
<?php
if (isset($_COOKIE['GNU-Association'])) {
   echo $_COOKIE['GNU-Association'];
}
?>
```

La malédiction du cookie

Tout d'abord, la gestion du cookie (le setcookie()) doit être fait **AVANT** tout envoi de texte, de contenu HTML. En effet, il s'agit d'une commande qui va générer une information dans **l'entête HTTP** (donc, si vous écrivez ne serait-ce qu'un espace, le serveur web aura été obligé d'envoyer l'entête HTTP.. D'ou un « *Headers already sent* »...)

Le problème du cookie :

- sur le poste client !!! Aucun contrôle de notre part !
- Certains utilisateurs les refusent..
- Les utilisateurs peuvent les modifier, en altérer le contenu.
- Est ce que je peux récupèrer des cookies d'autres sites ? (*par exemple, ceux déposés par un concurrent ?*)

Solution : les sessions

Il suffit *d'identifier* l'utilisateur (par exemple, avec un unique cookie qui contient une valeur aléatoire), et d'associer à cette valeur tout un ensemble d'informations (sur le serveur, dans un fichier, lié à cette valeur). La valeur a une durée de vie limitée (30 minutes aprés sa dernière utilisation).

C'est le principe des **Sessions**. Une seule info sur le client (un cookie, ou une variable GET), et tout est sur le serveur, sous notre contrôle...

Les commandes de SESSIONS

Les variables de sessions seront accessibles via un tableau associatif $_**SESSION**.

Si vous voulez pouvoir *manipuler* les variables de session (qui vous permettent donc de faire passer des informations de pages en pages), il faut **déclencher** le mécanisme :

```
<?php
   session_start(); // Création de la session
   $_SESSION['prenom'] = 'Popaul'; //
Sauvegarde dans la session créée de la variable
"prenom"
?>
```

Attention !

Le mécanisme des sessions utilise des informations dans l'entête HTTP (*voir plus loin les headers*) :

Il faut donc..... *!*

D'autre part, vous pouvez ensuite directement utiliser le tableau associatif $_SESSION.. Les fonctions habituelles sont obsolètes (session register et Cie)

Pour détruire la session (et donc perdre toutes les valeurs) : **session_destroy();**

Pourquoi faire ?

On peut utiliser les sessions par exemple :

- Pour *l'authentification* des utilisateurs (une variable de session contient l'id de l'utilisateur. Cette variable est remplie par une page spéciale *authentification.php*. A chaque entrée dans une page, on vérifie **l'existence** de la variable : Sinon, c'est une tentative d'accès illégale, on renvoit vers la page *authentification.php*)
- pour réaliser un panier (les choix du client sont stockés dans un tableau, variable de **session**)
-...

Les entêtes HTTP

Protocole HTTP : il permet l'échange de document : une *entête* HTTP caractérise le document qu'elle décrit : Son *contenu* (text/html, image/jpg), sa *taille*, sa *date de création*, etc etc...

(*ne pas confondre avec le head du html, qui est une entête du contenu de la page hypertexte*)

Il est possible en PHP de **jouer** avec les **entêtes** http... Sous réserve de bien générer ces informations AVANT tout contenu (*que ce soit un simple espace, même avant la balise <html>*)

<? header("HTTP/1.0 404 Not Found"); ?>

Exemple de header

Un header intéressant de page « de traitement »:

```
<?php
/* plein de traitement dans la base de données*/

/* Redirige le client vers une autre page*/
header("Location: accueil.php");
?>
```

Autre exemple, un page qui ne doit pas être mise en cache :

```
<?php
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Mon, 03 Jul 1999 06:00:00 GMT"); // Date du passé
/* des traitement ensuite
?> <html><body>.....
```

Exemple de header reçu...

HTTP/1.x 302 Found
Date: Fri, 04 Apr 2008 07:32:17 GMT
Server: Apache/2.2.3 (Debian) mod_python/3.2.10 Python/2.4.4 PHP/4.4.4-8+etch4 mod_perl/2.0.2 Perl/v5.8.8
X-Powered-By: PHP/4.4.4-8+etch4
Set-Cookie: ***PHPSESSID****=30fd278b33040f38d65274f6a6c64cbd; path=/*
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate,
Pragma: no-cache
Location: index.php?demande=identification
Content-Length: 4224
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=ISO-8855-1

On voit ici que le serveur me définit un cookie qui s'appelle **PHPSESSID** (à présenter ensuite à chaque échange) (plugin firefox : *livehttpheaders*)

Exemple de valeurs Content-Type....

Content-Type: *image/jpeg*
Content-Type: *text/css*
Content-Type: *image/png*
Content-Type: *text/html*; *charset=ISO-8955*-1'
Content-Type: *text/plain*' // plain text file
Content-Type: *application/zip* // ZIP file
Content-Type: *application/pdf* // PDF file
Content-Type: *audio/mpeg* // Audio MPEG (MP3,...) file
Content-Type: *application/x-shockwave-flash* // Flash animation

Exemple d'headers utiles

```
// Erreur du serveur
header('HTTP/1.1 500 Internal Server Error');

// Redirection (déjà vu plus haut)
header('Location: http://www.gnu.org/');

// Redirection temporisée
header('Refresh: 10; url=http://www.gnu.org/');
print 'Dans 10 secondes : le GNU';

// faisable aussi en html....
// <meta http-equiv="refresh" content="10;http://www.gnu.org/ />

// Modifier les messages informatifs
header('X-Powered-By: Myself/0.8.a');

// Langage (en = English)
header('Content-language: en');
```

Suite headers...

```
// date de dernière modification (pour la gestion du cache)
$time = time() - 60; //
header('Last-Modified: '.gmdate('D, d M Y H:i:s', $time).' GMT');

// Dire au client que le contenu n'a pas été modifié
header('HTTP/1.1 304 Not Modified');

// Headers pour permettre le téléchargement d'un fichier
header('Content-Type: application/octet-stream');
header('Content-Disposition: attachment;
  filename="exemple.zip"');
header('Content-Transfer-Encoding: binary');
// et on balance le contenu  (le client posera alors une question)
readfile('exemple.zip');
```

Afficher une image stockée en Base de données

Dans un champ *BLOB*, vous avez le contenu *binaire* d'une image. Pour l'afficher, il faut un script php qui écrit le contenu de cette colonne. Cependant, le navigateur va prendre cela pour du *texte*, et l'afficher tel quel...

Pour qu'il considère cela comme une image, il suffit de lui donner le **header** correspondant..

On appelle le script de la façon suivante :

<img src='recup_image.php?id=12'>

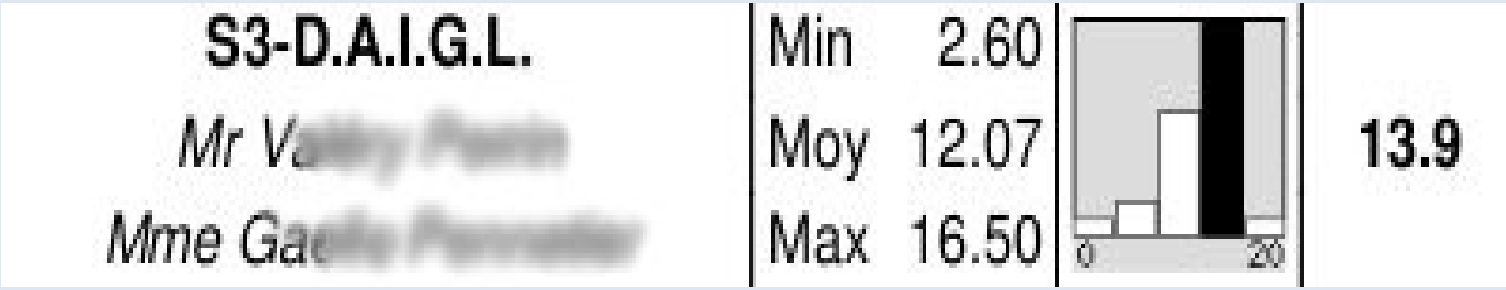
et le script recup_image.php

```
<?

 $connexion = mysql_connect("localhost", "moi", "monPss") or die
  (mysql_error());
      $database = mysql_select_db("maDb") or die (mysql_error());

      $apercu = mysql_query("SELECT  img FROM images WHERE id = ".
   $_GET[id]) or die (mysql_error ());

      $reponse = mysql_fetch_assoc($apercu);

        //On a le contenu binaire, il faut juste générer le header !!
         header ("Content-type: image/jpeg ");

         echo $reponse[img];
?>
```

Fabrication d'une image

Fonction qui dessine un petit graphique, calculé d'aprés les notes obtenues par un étudiant.

L'image est entièrement dynamique, fabriquée grace aux fonctions fournies par la bibliothèque GD (normalement dans PHP)..



Exemple fabrication d'une image

```
<?php

function graphe($notes,$pos,$X=60,$Y=80)
{/*************************************************
  cette fonction construit une image, et trace un histogramme selon le nombre de
   valeurs du tableau notes qui lui est passé...  De plus, il colorie la barre
   correspondante.
 ************************
  On passe le tableau de "$notes"
    Par exemple, on decide de faire 5 barres (0 a 4, 4 a 8, 8 a 12,12 a 16, et 16 a
   20). L'etudiant a 13,4 de moyenne, il fait donc partie de la barre n°3 (notée de 0
   a 4).
   On pourra aussi eventuellement indiquer une taille en pixel (imgX et imgY) pour
                                l'image. Sinon, l'image fera 80X60 pixels...

    exemple comme au dessus, avec une taille de 120 par 100, stockage dans
img1.png : graphe.php($nb,3,120,100,"img1.png"); */
```

Suite exemple image

```
// variables pour le tableau
  $NB=count($notes); //nombre d'elements du tableau
  $REDUC=0.5; //pourcentage representant la hauteur maxi

  $Message=""; //eventuel message d'erreur
//On commence a creer l'image...
  $im=imagecreate($X,$Y+10) or die("probleme GD");
  $backcolor=imagecolorallocate($im,220,220,220);//la première couleur =  fond

  //quelques calculs de preparation...
  //calcul de la valeur du point, Reduc du total
  //divise par la hauteur
  foreach($notes as $lanote) $total+=$lanote;

  //Enfin, derniere verif avant creation du contenu
  if ($pos<-1 or $pos>=count($notes) or $total==0) {
          $verif=false;
           $Message="Aucune note";
  }  else {$verif=true;}
```

Suite...

```
if($verif==true) //Si tout est ok, on fabrique le contenu..
{//batterie de couleurs
    $black=imagecolorallocate($im,0,0,0);
    $gris=imagecolorallocate($im,90,90,90);
    $blanc=imagecolorallocate($im,255,255,255);
   // le tour de l'image et la deco
     imagerectangle($im,0,0,$X-1,$Y-1,$black);
     imagestring($im,2,2,$Y-2,"0",$black);
     imagestring($im,2,$X-12,$Y-2,"20",$black);
```

Suite Image...

```
//au cas ou, je recalcule la reduction, si le coeff est trop fort
foreach($notes as $lanote)
                if ($lanote/$total>$REDUC) {$REDUC=$lanote/$total;}

$haut=$Y/($total*$REDUC);
for($i=0;$i<$NB;$i++)   {//on calcule les points de reference $DBx,$DBy
debut x et y $FNx,$FNy fin x et y
          $DBx=$i*($X/$NB);
          $DBy=$Y-($notes[$i]*$haut);
               $FNx=($i+1)*($X/$NB)-1;
          $FNy=$Y-1;
               if ($i==$pos)     {//c'est la note de l'eleve
                    imagefilledrectangle($im,$DBx,$DBy,$FNx,$FNy,$black);
               } else{//c'est une barre normale
```

Fin image...

```
                imagerectangle($im,$DBx,$DBy,$FNx,$FNy,$gris);
                imagefilledrectangle($im,$DBx+1,$DBy+1,$FNx-1,$FNy-
  1,$blanc);
                }
            }
        }  else  {
            imagestring($im,2,2,10,$Message,$black);// valeurs fournies sont
        incoherentes... On affiche le message d'erreur
        }
   //et on dessine
        header("Content_type: image/png");
        imagepng($im);

       imagedestroy($im);
 }//fin de la fonction graphe
?>
```