SESIONES

Expliquemos paso a paso cómo usar las sesiones en PHP para validación de usuario (inicio de sesión) y para enviar datos entre las páginas del sitio Web. Expliquemos cómo crear una sesión, cómo crear variables de sesión y cómo cerrar una sesión en PHP.

Inicio de la sesión en PHP con session_start

La función PHP **session_start** crea una sesión o reanuda la actual basada en un identificador de sesión pasado mediante una petición GET o POST, o pasado mediante una cookie.Para usar una sesión nominada hay que utilizar la función *session_name* antes de llamar a *session_start*.

Cómo usar session_start en nuestro sitio web

En todas las páginas de nuestro sitio Web donde necesitemos usar algún dato del usuario que ha iniciado sesión o mostrar alguna opción especial para el usuario deberemos añadir el siguiente código, siempre al principio del fichero (**antes** de cualquier etiqueta HTML de la página):

```
<?
  session_start();
?>
```

Con el procedimiento "session_start()" iniciaremos la sesión, si ya está iniciada no hará nada.

Almacenar una variable de sesión. Establecer un valor para una variable de sesión en PHP

Para registrar una variable de sesión y establecer un valor usaremos **$_SESSION["nombre_variable"]**.

Añadir un enlace para que el usuario pueda cerrar la sesión en cualquier momento

Cuando el usuario ha iniciado sesión, a veces es recomendable añadir un enlace para que éste pueda cerrar la sesión en cualquier momento de forma segura. Así evitaremos accesos indebidos por otros usuarios que usen el mismo equipo que el que inició sesión. En teoría, siempre que se cierre el navegador Web se cerrará la sesión automáticamente, pero a veces es aconsejable añadir este enlace para que el usuario decida cuándo cerrar la sesión.

Para añadir el enlace de "Cerrar sesión", podemos insertar el siguiente código PHP donde queramos que aparezca el enlace:

```
if (! empty($_SESSION["nombre_usuario"]))
{
  echo "<a href="cerrarsesion.php" title="Cerrar Sesión">Bienvenid@</a>" .
  $_SESSION["nombre_usuario"];
}
```

Explicamos el código anterior:

- Con *if(! empty($_SESSION["nombre_usuario"]))* comprobamos si el usuario ha iniciado sesión, si lo ha hecho la variable de sesión *nombre_usuario* tendrá un valor, por lo que se ejecutará el código del if.

- Con el código que hay dentro del if, lo único que hacemos es mostrar algo así: *Bienveni@ fulanito*

  Donde:

  - "Bienveni@" será un enlace a la página "cerrarsesion.php" que describiremos más adelante.
  - "fulanito": será el nombre de usuario que haya iniciado la sesión.

Como ya hemos comentado, este código PHP lo pondremos en todas las páginas de nuestro sitio Web donde queramos que aparezca el enlace a cerrar sesión. Lo colocaremos, dentro de cada página, en el sitio donde queramos que aparezca. Puesto que PHP permite "mezclarse" con código HTML no tendremos problema, por ejemplo:

```
<td style="background-image:url(img/ejemplo.png)">

<?
 if (! empty($_SESSION["nombre_usuario"]))
 {
   echo "<a href="cerrarsesion.php" title="Cerrar sesión">Bienvenid@</a>" .
      $_SESSION["nombre_usuario"];
 }
?>

<table border="0" align="right" cellpadding="0" cellspacing="0">
<tr>
```

Fichero "cerrarsesion.php" para cerrar sesión en PHP con session_destroy

session_destroy destruye toda la información asociada con la sesión actual. No destruye ninguna de las variables globales asociadas con la sesión, ni destruye la cookie de sesión. Para volver a utilizar las variables de sesión se debe llamar a session_start().

Para destruir la sesión completamente, como desconectar al usuario, el id de sesión también debe ser destruido.

Uso de la función session_destroy para cerrar la sesión en PHP

Crearemos un fichero de texto plano sin formato con el siguiente contenido, guardándolo con el nombre "cerrarsesion.php":

```
<?
 session_start();
 unset($_SESSION["nombre_usuario"]);
 unset($_SESSION["nombre_cliente"]);
 session_destroy();
 header("Location: index.php");
 exit;
?>
```

A continuación explicamos cada línea del fichero:

- session_start: función ya explicada más arriba, puesto que la sesión ya está iniciada, no hará nada.
- unset($_SESSION["nombre_usuario"]) y unset($_SESSION["nombre_cliente"]): liberarán las variables de sesión registradas, en el ejemplo liberamos dos variables de sesión: nombre_usuario y nombre_cliente.
- session_destroy: libera la sesión actual, elimina cualquier dato de la sesión.
- header("Location: index.php"): tras liberar la sesión con los métodos anteriores, esta línea vuelve a mostrar la página "index.php" de nuestro sitio web.

Nota 1: si tenemos muchas variables de sesión y queremos liberarlas todas podemos usar este código PHP:

Otras funciones

Conviene dejar claro que el método de sesiones explicado no es el más profesional y seguro. Existen muchas más funciones para el tratamiento de las sesiones en PHP como son:

- session_cache_expire: devuelve la caducidad de la caché actual.
- session_cache_limiter: obtener y/o establecer el limitador de caché actual.
- session_commit: alias de session_write_close.
- session_decode: decodifica la información de sesión desde una cadena.
- session_destroy: destruye toda la información registrada de una sesión.
- session_encode: codifica la información de la sesión actual como una cadena.
- session_get_cookie_params: obtener los parámetros de la cookie de sesión.
- session_id: obtener y/o establecer el id de sesión actual.
- session_is_registered: averiguar si una variable global está registrada en una sesión.
- session_module_name: obtiene y/o establece el módulo de sesión actual.
- session_name: obtener y/o establecer el nombre de la sesión actual.
- session_regenerate_id: actualiza el id de sesión actual con uno generado más reciente.
- session_register: registrar una o más variables globales con la sesión actual.

- ⇨ session_save_path: obtener y/o establecer la ruta de almacenamiento de la sesión actual.
- ⇨ session_set_cookie_params: establecer los parámetros de la cookie de sesión.
- ⇨ session_set_save_handler: establece funciones de almacenamiento de sesiones a nivel de usuario.
- ⇨ session_start: inicializar información de sesión.
- ⇨ session_unregister: deja de registrar una variable global de la sesión actual.
- ⇨ session_unset: libera todas las variables de sesión.
- ⇨ session_write_close: escribir información de sesión y finalizar la sesión.

Bien usadas todas estas funciones (o las necesarias) podrán dotar a nuestro sitio Web de mayor robustez y seguridad. Por supuesto, lo explicado aquí no es lo más seguro pero sí, tal vez, lo más sencillo de implementar.

Por ejemplo, podríamos usar la función session_id para obtener el ID de sesión del usuario actual y guardarlo en una tabla de una base de datos, de esta forma podríamos saber cuántos usuarios han iniciado sesión en nuestro sitio Web, qué usuarios y en qué día y hora:

```
$session_id = session_id();
$sql = "INSERT INTO sesiones (user_id, session_id) VALUES ('" .
   $userid . "', '" . $session_id . "')'";
$resultado_sql = mysql_query($sql);
```