Tema 2. Sesiones en PHP

HTTP es un protocolo "stateless": El protocolo no mantiene el estado entre 2 transacciones.
Si un usuario solicita una primera página, y a continuación una segunda al mismo servidor web, HTTP no proporciona una manera de saber si las 2 peticiones provienen del mismo usuario/cliente.

Por ello, los lenguajes de servidor proporcionan manejo de SESIONES, con el objetivo de mantener la comunicación entre un navegador y el servidor Web para un conjunto de páginas Web.

Una sesión es una secuencia de comunicaciones sucesivas entre un navegador y un sitio Web a las que se les asigna un identificador común

Sesiones en PHP

Las sesiones en PHP permiten al usuario almacenar información de forma persistente en el servidor, mientras el usuario se mantenga navegando dentro del mismo sitio web
(Sin necesidad de recurrir a almacenamiento permanente en ficheros o bases de datos
Y Sin necesidad de enviar en todas las peticiones la información a conservar con campos ocultos, get, …)
La Sesiones en PHP funcionan creando un ID único (SID) para cada visitante y almacenando un conjunto de variables asociadas a ese ID, el cual es almacenado en una cookie o bien, es propagado en la URL del sitio visitado.

1. Iniciando/Recuperando una Sesión en PHP

Para utilizar sesiones en PHP, lo primero hay que inicializarlas con:

```
session_start();
```

session_start() comprueba si existe una sesión en curso.

- Si no existía la crea, asignándole un id (identificador de sesión)
- En caso de ya existir la retoma, es decir, recarga todas las variables de la sesión existente.

Antes de comenzar a operar con la información de una sesión PHP, debemos, explícitamente, indicar el inicio de la sesión, en cada página que creemos.

El inicio de sesión debe ser la PRIMERA instrucción en cada página, por ejemplo

```
<?php
session_start();

/*  Código PHP   */
?>
<html>
<body>
. . .
</body>
</html>
```

El código anterior registra la sesión del usuario en el servidor, asignando un ID de sesión. A partir de esto, ya podemos operar con la información almacenada en la sesión.

2. Almacenando Valores en una Variable de Sesión

La manera de definir variables de sesión en PHP es utilizando la variable/array **$_SESSION**

$_SESSION es uno de los arrays superglobales asociativos de PHP, y almacena correspondencias de nombres de variables (clave) con valores.

```
<?php
        session_start();
        // Crea variable de sesión 'visitas' con un valor
        $_SESSION['visitas'] = 1;
?>
<html>
<body>
        <?php
        // Lee el contenido de la variable de sesión 'visitas'
        echo "Visitas: ".$_SESSION['visitas'] ;
        ?>
</body>
</html>
```

Este código desplegará:
Visitas: 1

Para hacer que nuestra variable de sesión actúe a modo de contador de visitas, haremos que la variable de sesión 'visitas' se cree con valor 1 o se incremente, en función de su existencia previa.

```
<?php
        session_start();
        if( ! isset($_SESSION['visitas']))
        {
                $_SESSION['visitas'] = 1;
        }
        else
        {
                $_SESSION['visitas'] = $_SESSION['visitas'] + 1;
        }
?>
```

3. Eliminando una variable de sesión

Si deseamos eliminar una variable de sesión, al igual que cualquier variable, utilizaremos el método php unset (como con todas las variables)

```
unset (variable);
```

P.e: **unset($_SESSION['visitas]);**

*¡Cuidado! *unset[$_SESSION]*, destruiría todas las variables de la sesión

4. Destruyendo una sesión

Si deseamos eliminar solamente algunos datos de la sesión, podemos utilizar la función unset(). Pero para eliminar toda la sesión, debes utilizar la función:

```
session_destroy();
```

```
<?php
    session_start();
    // unset() es utilizada para liberar una variable específica
    unset($_SESSION['visitas']) ;
    // session_destroy() destruye la sesión de forma completa
    session_destroy();
?>
```

session_destroy() reiniciará completamente la sesión y se perderán todos los datos almacenados hasta el momento en sus distintas variables.

Las sesiones terminan de modo explícito con la función session_destroy(), pero también de manera implícita cuando se cierra el navegador

** Es conveniente que estas funciones aparezcan al comienzo del fichero, antes de cualquier otra instrucción PHP, etiqueta HTML o línea en blanco*

*Existe como alternativa al uso directo del array $_SESSION el uso de las funciones **session_register()**, **session_unregister()**, **session_is_registered()** y **session_unset()** para operar sobre la información de la sesión. Sin embargo, existen varios problemas en su uso que pueden hacernos introducir fallos. Estas funciones solo trabajan cuando la directiva register_globals de nuestro **php.ini** está activada, cosa que no es nada recomendable por motivos de seguridad, (Desde PHP 4.2.0 en adelante el valor por defecto de register_globals es off por motivos de seguridad).*

¿Cómo se **propagan** las sesiones?

Los valores almacenados en la sesión son serializados en un archivo al cerrar cada script y al continuar la sesión en otra página son de nuevo cargados en el array $_SESSION.

El archivo de la sesión suele estar en el directorio /tmp y tiene por nombre el session_id del visitante. (Podemos configurar dónde queremos serializar las sesiones con la directiva *session.save_path* de nuestro **php.ini** o en tiempo de ejecución mediante la función **session_save_path()**)

La cuestión es, si *pagina1.php* crea una sesión con ciertas variables ¿Cómo llega **session_start()** en *pagina2.php* a conocer que el visitante ya tiene una sesión y continuarla? La única forma que tiene de hacerlo es que le proporcionemos de alguna forma el session_id del visitante→ el session_id debe ser propagado de alguna manera

Existen 2 maneras de propagar la sesión:

- A través de cookies
- Por URL

Propagación de sesiones a través de cookies

La forma habitual de propagar las sesiones es a través de cookies, ya que por defecto, el archivo php.ini tiene la directiva *session.use_cookie=1*
Si queremos desactivar la propagación del session_id por cookie debemos cambiar *session.use_cookie=0* en nuestro **php.ini**, o si no tenemos acceso al **php.ini** podemos hacerlo en nuestra página mediante la función **ini_set()** de la siguiente forma:

```
ini_set("session.use_cookie", "0");
```

además de

```
ini_set("session.use_only_cookies",0);
```

La propagación de sesión por cookie funciona así:
session_start() buscará en un cookie el *session_id* del visitante. Si no lo encuentra creará una nueva sesión, creará un cookie para la sesión y almacenará en él el *session_id* del visitante. Una próxima llamada a session_start(), leerá el cookie que contiene el *session_id* y podrá continuar la sesión correspondiente.

Como vemos, **session_start()** se encarga de crear y leer el cookie, así que el proceso es automático y no tendríamos que preocuparnos por él.

Pero el usuario/navegador tiene la posibilidad de no aceptar cookies. Para ese caso PHP dispone de una opción alternativa que permite la propagación de la sesión a través de la URL aún en el caso de que las cookies estén desactivadas.

Propagación de sesión por URL

La propagación del *session_id* por URL implica escribir el session_id en la URL de cada enlace interno de nuestro sitio, como en este ejemplo:

```
<a href="http://localhost/blog/pagina2.php?PHPSESSID=F513fad624vDx3">
    Visita la siguiente página
</a>
```

donde la variable PHPSESSID se encarga de pasar el session_id de la sesión a pagina2.php.

1. El nombre de la variable que porta el *session_id* es igual al nombre de la sesión, y por defecto es PHPSESSID (puede definirse un nombre diferente para la sesión con la directiva *session.name* de nuestro **php.ini** o en php por la función **session_name()**)
2. Para recuperar el id de nuestra sesión, podemos usar la función php **session_id()**

```
<?php $id= session_id() ?>
<a href=http://localhost/blog/pagina2.php?PHPSESSID= " <?php echo $id?>">
    Visita la siguiente página
</a>
```

3. Para simplificar el proceso anterior está disponible la constante **SID** que contiene la cadena completa "nombre_de_sesion=session_id", en nuestro caso "PHPSESSID=F513fad624vDx3"

```
<a href=http://localhost/blog/pagina2.php?<?php echo SID; ?>">
        Visita la siguiente página
</a>
```

Propagación automática del session_id en cada URL

Para hacerlo más fácil PHP puede encargarse de transformar por nosotros todos los enlaces internos de nuestro sitio para que incluyan el session_id.
Para ello la directiva *session.use_trans_sid* en nuestro **php.ini** debe estar activada (a "1").
Por motivos de seguridad viene con el valor "0" por defecto, de modo que para habilitarla deberemos editar nuestro **php.ini** o utilizar la función **ini_set()**.
ini_set("session.use_trans_sid", "1");

Si esta directiva está habilitada y PHP ha sido compilado con soporte para trans_id, un enlace escrito de la siguiente forma en nuestra página:

```
<a href="http://localhost/blog/pagina2.php">
    Visita la siguiente página
</a>
```

será transformado automáticamente por PHP al siguiente enlace:

```
<a href="http://localhost/blog/pagina2.php?PHPSESSID=%20F513fad624vDx3">
    Visita la siguiente página
</a>
```

propagando por nosotros el session_id en cada URL.

Cookies en PHP

Las cookies son un mecanismo de los servidores web para guardar información en el ordenador del cliente y recuperarla cada vez que el navegador les pide una página. Así pues, una cookie no es más que un pequeño archivo de texto que el servidor web deposita en el computador del cliente. Este archivo de texto está formado por parejas (nombre de la cookie → valor de la cookie).

Cuando el navegador solicita una página PHP a un servidor (un dominio) que ha guardado previamente cookies en ese ordenador, el navegador incluye en la cabecera de la petición HTTP todas las cookies (el nombre y el valor) creadas anteriormente por ese servidor.

El script PHP recibe los nombres y valores de las cookies en la matriz $_COOKIE

1. Crear una cookie

setcookie (nombre, valor, tiempo de expiración)

El tiempo de expiración debe indicarse en segundos, desde su creación
Si el tiempo de expiración es 0, la cookie desaparecerá al acabar la sesión

Ejemplo 1:

```
setcookie("usuario","Juan Perez",time()+3600);
```

→Crea una cookie llamada “usuario”, con valor “Juan Perez” y que expirará en una hora.

La creación de cookies debe estar ubicada en la parte superior del documento, antes de <html> (*)

(*) Hay que tener la precaución de utilizar la función setcookie() antes de empezar a escribir el contenido de la página, porque si no PHP producirá un aviso y no se creará la cookie. El motivo es que las cookies se crean mediante cabeceras de respuesta HTTP y las cabeceras se envían antes del texto de la página. Es decir, cuando PHP encuentra una instrucción que escribe texto, cierra automáticamente la cabecera; si a continuación PHP encuentra en el programa la función setcookie(), da un aviso porque ya se han enviado las cabeceras y no se crea la cookie

2. Leer el valor de una cookie

Para leer el valor de una cookie, en PHP utilizaremos el array $_COOKIE. En el ejemplo anterior:

```
echo $_COOKIE['usuario'];
```

visualizará *Juan Perez*

Es habitual, antes de usar una cookie, verificar su existencia, como en el siguiente ejemplo:

```
if(isset($_COOKIE['usuario']))
    echo "Bienvenido ".$_COOKIE['usuario']."! <br>";
else
    echo "Bienvenido Invitado";
```

3. Eliminar una cookie

Para eliminar una cookie, hay que ajustar la fecha de expiración de la misma al pasado

```
setcookie('usuario','',time()-3600);
```
→ // Ajustar la fecha de expiración hace una hora atrás