3.2 SESIONES EN PHP

¿Qué es eso de las sesiones? Las sesiones son simplemente **variables** que mantienen su valor para cada usuario a lo largo de toda la visita a nuestra página web, no como las variables normales, que sólo están activas durante la ejecución del script. Por ejemplo, si tenemos la variable *$nombre = 'Jose';* ésta seguirá activa durante toda la ejecución del script, pero al cambiar de página ésta desaparecerá. En cambio una variable de sesión permanecerá activa y con el valor que le hayamos asignado durante toda la visita del usuario. ¿y cómo es eso?. Vamos a verlo.

Una definición más formal sería que una sesión es un mecanismo de programación de las tecnologías de web scripting que permite conservar información sobre un usuario al pasar de una página a otra. **A diferencia de una cookie, los datos asociados a una sesión se almacenan en el servidor y nunca en el cliente**. Es decir, son variables que, en este caso, se guardan en el servidor.

En la mayoría de las tecnologías de web scripting, las sesiones se implementan utilizando una cookie que almacena un valor que identifica al usuario en el servidor web cada vez que pasa de una página web a otra. En el servidor web están almacenados todos los datos de la sesión y se accede a ellos cada vez que se pasa de página gracias al identificador almacenado en la cookie.

En Mozilla Firefox podemos ver la cookie PHPSESSID que se emplea en PHP para almacenar el identificador de una sesión. Además, el siguiente código de PHP muestra el identificador de una sesión y el nombre de la cookie que almacena la sesión:

```
<?php
  session_start();
  echo 'session_id(): ' . session_id();
  echo "<br />\n";
  echo 'session_name(): ' . session_name();
  echo "<br />\n";
  print_r(session_get_cookie_params());
?>
```

Para nuestro día a día, **utilizar variables de sesión** es tan sencillo como llamar a ***session_start()*** al principio de la página, antes de cualquier salida por pantalla(antes de <html> y cualquier otra cosa que genere una salida por pantalla. Esto es así ya que la información viaja en las cabeceras del mensaje http), y usar la variable superglobal ***$_SESSION*** para almacenar valores en el servidor.

Para **destruir una sesión** y toda la información que contiene lo mejor es borrar explícitamente toda la información contenida en el array $_SESSION, borrar la cookie que contiene el identificador de la sesión y por último destruir la sesión:

```
<?php
  // Borra todas las variables de sesión
  $_SESSION = array();
  // Borra la cookie que almacena la sesión
  if(isset($_COOKIE[session_name()])) {
```

```
    setcookie(session_name(), '', time() - 42000);
  }
  // Finalmente, destruye la sesión
  session_destroy();
?>
```

Para controlar el acceso a una zona privada dentro de una web, se suele emplear una (o varias) variable/s de sesión que inicializamos con un cierto valor en la página de control de acceso; en las páginas donde se quiere controlar si el usuario tiene permiso para acceder se consulta el valor de la variable de sesión para ver si tiene el valor esperado. Si no contiene el valor esperado, lo normal es mostrar una página con un mensaje de error o redirigir al usuario a la página principal del sitio web. Lo practicaréis enseguida...

Vamos a hacer un ejemplo sencillo con dos páginas, en una definiremos una variable de sesión y en la otra veremos su valor.

Página 1:

```
<?php
  session_start();
  $_SESSION['nombre'] = 'Juan';
?>
```

Página 2:

```
<?php
  session_start();
  echo "El nombre es: ".$_SESSION['nombre'];
?>
```

Guardamos el código en pagina1.php y pagina2.php respectivamente.

Siempre tenemos que llamar a session_start() en un script antes de usar las sesiones, ya sea para acceder a ellas o para almacenar valores. Si ejecutamos la primera página veremos que aparece en blanco, pero si seguidamente vamos a la página 2 veremos que aparece "El nombre es: Juan", sin embargo, "Juan" no aparece por ninguna parte en el código de la pagina2.php, todo es obra de las sesiones, junto con php, claro.

Esto nos puede ser útil por ejemplo para identificar a un usuario. En una aplicación completa, como la que puede que hagamos en el siguiente trimestre, primero comprobaríamos en la base de datos si el usuario y la contraseña introducidos son correctos y, si es así, haríamos algo del tipo $_SESSION['logged'] = true (por ejemplo. Como veis, $_SESSION es una array asociativo, donde nosotros elegimos el índice; este caso hemos puesto *logged*, y el valor en este caso *true*). Luego sólo tendríamos que comprobar esta variable de sesión, tal como se comenta más arriba. Aunque se suele aprovechar una misma variable de sesión, como esta que se ha creado para, de paso, almacenar algún valor útil para la aplicación; por ejemplo el nick del usuario. Entonces haríamos $_SESSION['logged'] = 'Juan'. Así, con una sola variable, podemos saber si alguien se ha logueado correctamente (sólo me haría falta comprobar si existe

la variable $_SESSION['logged'] para saber si hay alguien logueado), y además tengo un dato que me puede hacer falta, en este caso su 'nick'.

SISTEMA DE AUTENTIFICACIÓN EN PHP

Ahora vamos al grano y a lo importante: ponerlo en práctica.Veréis como os queda todo más claro. En el siguiente enlace de desarrolloweb.com, tenemos un manual de autentificación en PHP. Leed con atención y **realizad los pasos del 1 al 6** (salir de la aplicación segura de PHP). Os explica el funcionamiento de un sistema de autentificación en PHP usado sesiones. Probad su funcionamiento:

Haced el ejercicio que os plantea, de los puntos del 1 al 6 (al menos), y personalizadlo un poco, que se note que es vuestro. Y por favor, no copies y peguéis, escribid el código y entended lo que estáis haciendo. Esta es una de las tareas que tenéis que entregarme en este tema, pero es realmente importante que os quede claro.

P.D. Hay más información interesante, como ***Diferentes formas de cerrar sesión en PHP***, y ***Cierre de sesión al cerrar el navegador en PHP***. Los dejo como material de ampliación, para los que os apetezca y queráis profundizar más.

Nota: otro ejemplo sencillo de empezar con las sesiones:
https://diego.com.es/sesiones-en-php