Cookies y sesiones

Realizado por A.Garay (dpto.de Informática)

Resumen de contenidos

- **Cookies**
  - Definición y conceptos
  - Ubicación y manejo
  - Creación
  - Acceso a información de cookies.
- **Sesiones**
  - Definición y conceptos
  - Inicio de sesión
  - Directivas en “php.ini”
  - Variables de sesión $_SESSION
  - Destrucción y cierre de sesión

Cookies: Definición y conceptos

Una cookie es un mini archivo de información sobre el usuario, que el servidor coloca en el equipo cliente con diferentes fines:

- Diferenciar a los usuarios y ofrecer a cada uno el contenido que corresponda
- Autenticar al usuario de forma directa, es decir, solo la primera vez es necesario escribir el login y la contraseña
- Establecer nuestras preferencias y opciones, por ejemplo, al personalizar nuestro método de búsquedas en Google.

El uso de cookies se justifica porque HTTP es un protocolo sin estado.

Cookies y seguridad

- Las cookies se asocian a una combinación **máquina+usuario_del_so+navegador**. Por tanto, son accesibles para todos la personas que compartan esta combinación, de modo que es muy recomendable no guardar información sensible, como datos personales, contraseñas, etc. según establece la LOPD (Ley Orgánica de Protección de Datos).
- La UE obliga actualmente a avisar a los clientes del uso de cookies en un sitio web.
- Las cookies pueden contener información sobre los hábitos de navegación de los usuarios en un sitio web, habitualmente para ser utilizados con fines publicitarios.
- Un sitio puede redireccionar a otros servidores (de forma transparente al cliente) para que éstos creen sus propias cookies (cookies de terceros). Por eso, ciertos sitios nos ofertan productos que hemos visitado anteriormente en otro sitio web

Cookies: Ubicación y manejo *

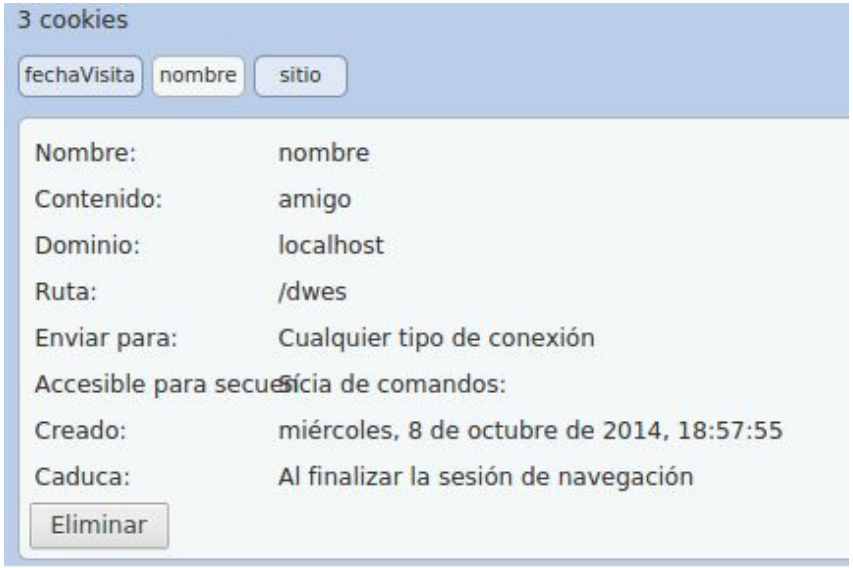
- En Firefox:
  - **Preferencias ➔ Privacidad ➔ Cookies**
- En Google Chrome podemos verlas y administrar su uso desde:
  - **chrome://settings/siteData**
  - Se puede hacer un listado de las cookies por servidor, consultar su contenido, borrarlas, y de manera global deshabilitarlas o habilitarlas

Cookies: Formato y creación en PHP

- Cada cookie está formada por un pareja ("nombre", "contenido") de tipo texto.
- Para crear una cookie en PHP se utiliza la función
  - **setcookie ("nombre", ["contenido"], [...])**
- En el script que crea la cookie, es imprescindible colocar la función setcookie() antes de generar cualquier salida, porque si no el intérprete producirá un Warning 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 el intérprete PHP encuentra la función setcookie(), se produce el warning y no se crea la cookie.

Cookies: Ejemplo y parámetros

```
setcookie ( "nombre", "amigo" );
setcookie ( "fechaVisita", "13/10/2014" );
setcookie ( "sitio", "www.java.es" );
```



- "**nombre**" = nombre de la cookie
- ["**contenido**"] = contenido de la cookie (puede ser vacío)
- ["**caduca**"] = fecha y hora de caducidad expresado en tiempo Unix, por defecto, al cerrar la ventana del navegador (no solo la pestaña)
- ["**ruta**"] = establece el directorio del servidor al que se asocia la cookie, que será enviada cuando el cliente acceda a dicho directorio (o subdirectorios)
- ["**dominio**"] = establece el dominio del servidor
- ["**enviar para**"] = un booleano que indica si la cookie se enviará en conexiones seguras https o en cualquier tipo de conexión
- ["**accesible**"] = establece si la cookie será accesible sólo por el servidor o mediante el navegador usando scripts

Sintaxis completa de setcookie(...)

setcookie(nombre, contenido, caducidad, ruta, dominio, https, soloHttp)

- **nombre:** nombre de la cookie
- **contenido:** contenido de la cookie
- **caducidad:** t.en segundos desde el EPOCH. (0 por defecto)

  * (ver nota al pie de página)
- **ruta:** Para ubicarla a otro nivel distinto del dir. actual

  ** (ver nota al pie de página)
- **dominio:** Para ubicarla en un dominio distinto al actual (cookies de terceros)
- **https:** booleano para indicar si sólo se envían por https
- **soloHttp (PHP>=5.2.0):** booleano para indicar que sólo se transmitan cookies vía http (evitar consulta por javascript)

Cookies: acceso a la información

- Mediante las variables superglobales $_COOKIE y $_REQUEST
- Con la función **isset (…)** controlaremos qué cookies hay y cuáles no.

```php
$nombre = isset ( $_COOKIE ["nombre"] ) ? $_COOKIE ["nombre"] : null;
$fechaVisita = isset ( $_COOKIE ["fechaVisita"] ) ? $_COOKIE ["fechaVisita"] : null;
$sitio = isset ( $_COOKIE ["sitio"] ) ? $_COOKIE ["sitio"] : null;
if ($nombre != null && $fechaVisita != null && $sitio != null) {
    echo "<h1>Hola $nombre</h1>",
    "Tu última visita fue el $fechaVisita",
    " redirigido desde $sitio";
} else {
    setcookie ( "nombre", "amigo" );
    setcookie ( "fechaVisita", "13/10/2014" );
    setcookie ( "sitio", "www.java.es" );
    echo "Bienvenido desconocido";
}
```

IMPORTANTE: *Antes del primer echo o cualquier instrucción de salida*

Borrar una cookie

- Dos opciones
  - **setcookie**('nombre', '', time() - 3600);
    - valor nulo y caducidad en el pasado
  - **setcookie**(‘nombre’);
    - nombre de cookie sin más parámetros

Sesiones: conceptos

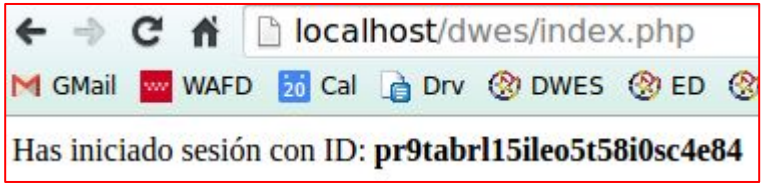
- Es un sistema que permite a los servidores guardar una relación entre conexiones, pero a diferencia de las cookies, la información se almacena en el servidor.
- HTTP es un protocolo sin estado, lo que significa que cada conexión es complementamente independiente de las demás.
- Esto implica que mediante HTTP no se puede guardar información que relacione una conexión con otra.
- La información de las sesiones se guarda durante un tiempo determinado, que puede ser el cierre de sesión por parte del usuario o un tiempo de expiración establecido previamente.

Sesiones: inicio y consulta en PHP

- Inicio de sesión**: session_start** ()
  - Esta función comprueba si hay una sesión abierta, si no la hay creará una. Si la hay reanudará la sesión con el mismo ID.
  - Cuando un script PHP crea una sesión, el servidor asocia el navegador del usuario con el archivo de sesión ubicado en el servidor.
  - El identificador se guarda en el usuario en forma de cookie, si no se permite el uso de cookies se añade el ID en la dirección de la página.
  - Igual que con las cookies el ID de sesión se envía con las cabeceras, por tanto, la función session_start() debe ir antes del contenido de la página.
  - Devuelve true si la sesión se inicia satisfactoriamente.
- Consulta de id de sesión**: session_id** ()

Sesiones: inicio (ejemplo)

```php
<?php
session_start ();
echo    "Has iniciado sesi&oacute;n con ID: <b>",
        session_id (),"</b>";
?>
```



Sesiones: directivas en el “php.ini”

La información de cada sesión se almacena según estas directivas

- **session.save_handler** = files (en forma de archivos)
- **session.save_path** = "\xampp\tmp" (ruta de almacenamiento)
- **session.use_cookies** = 1 (usar cookies para el identificador)
- **session.use_only_cookies** = 0 (solamente usar cookies)
- **session.name** = PHPSESSID (nombre de la cookie de sesión)

Sesiones: variables de sesión

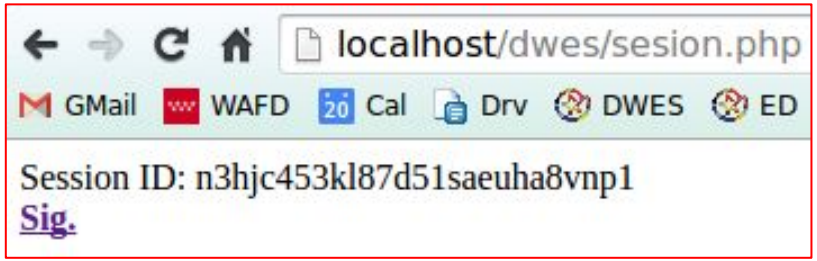
- Para permitir que una variable pertenezca a una sesión y pueda consultarse desde diferentes páginas, es preciso registrarla mediante la variable superglobal $_SESSION:
  - **$_SESSION** [‘variable’] = valor;
- Para dejar de registrar una variable :
  - **unset** ($_SESSION[‘variable’])
- Para dejar de registrar todas las variables
  - **session_unset** ( )

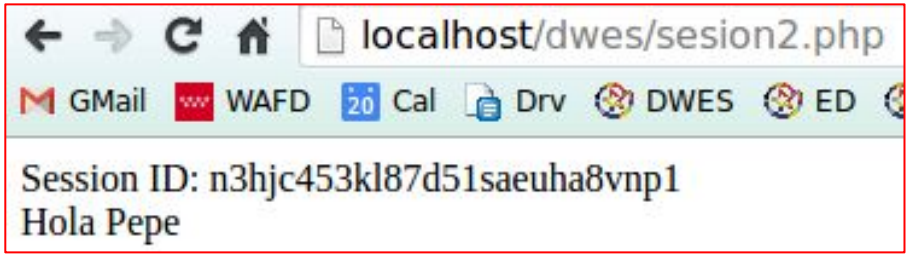
Sesiones: $_SESSION (ejemplo)

sesion.php → **sesion2.php**

```php
<?php
session_start ();
echo "Session ID: ", session_id (), "<br/>";
$nombre = "Pepe"; // Podría provenir de una BD
$_SESSION ['nombre'] = $nombre;
?>
<b><a href="sesion2.php">Sig.</a></b>
```

```php
<?php
session_start ();
echo "Session ID: ", session_id (), "<br/>";
echo "Hola {$_SESSION ['nombre']}";
?>
```



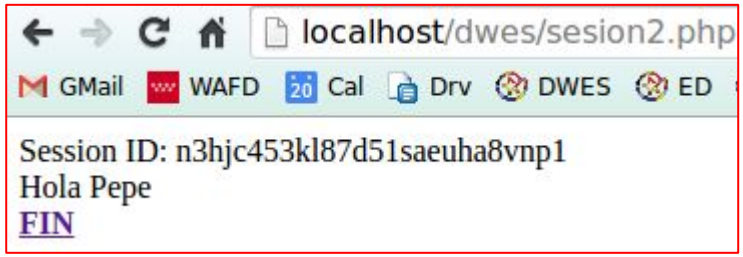


Sesiones: Destrucción y cierre

- Liberar variables: **session_unset()**
- Destruir la sesión: **session_destroy()**

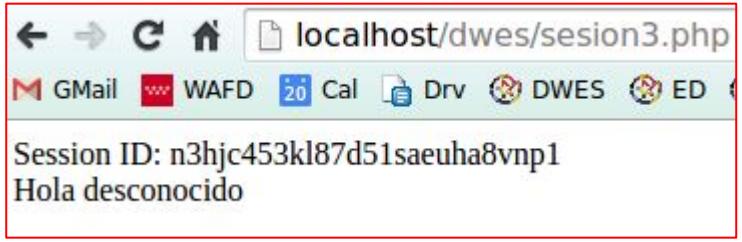
sesion2.php

```php
<?php
session_start ();
echo "Session ID: ", session_id (), "<br/>";
echo "Hola {$_SESSION ['nombre']}", "<br/>";
?>
<b><a href="sesion3.php">FIN</a></b>
```



sesion3.php

```php
<?php
session_start (); //recuperamos la sesión
session_unset (); //liberamos variables
echo "Session ID: ", session_id (), "<br/>";
if (isset ( $_SESSION ['nombre'] )) {
    echo "Hola {$_SESSION ['nombre']}"; }
else {
    echo "Hola desconocido<br/>"; }
session_destroy(); //destruimos la sesión
?>
```



¿Qué ocurre al destruir una sesión?

- En versiones antiguas de PHP, los ficheros de sesión se conservaban para siempre en el directorio temporal. Al destruir la sesión se desvinculaba del cliente borrando la cookie de sesión. El problema es que pasado cierto tiempo la carpeta temporal se llenaba, y había que vaciarla a mano, y además quedaba rastro de todo lo que hacían los usuarios a no ser que se hubiera tenido la precaución de hacer un `unset_session()`
- Actualmente `session_destroy()` BORRA el fichero temporal, pero mantiene la cookie, de manera que al volver a hacer un `session_start` se reutiliza el id pero sobre un nuevo fichero en blanco, por tanto la carpeta temporal no se llena.
- Si quisiéramos un nuevo id de sesión podríamos hacerlo borrando la cookie de sesión a mano (con `setcookie(...)` o bien utilizando `session_regenerate_id()`

Sesiones: ¿...sin cookies?

- El usuario puede decidir impedir el almacenamiento de cookies en su navegador, lo cual podría afectar al correcto funcionamiento de algunas aplicaciones.
- Ante esta situación el programador web, sólo tiene dos alternativas.
  - Mostrar una página de error, e indicar que se deben activar las cookies.
  - Apañarse con sesiones, obligando al navegador a enviar el ID de sesión cuando se navegue desde la página web generada por el script a cualquier otro sitio.
    - Lo consigue añadiendo campos hidden a cualquier formulario y añadiendo el envío del SID (vía GET) por cualquier etiqueta <a>
- Para conseguir esto último, bastará con modificar los siguientes parámetros en el "php.ini", o bien cambiarlos momentáneamente en el script con la función *ini_set($parámetro, $valor)*
  - **session.use_trans_sid = 1**
  - **session.use_cookies = 0**
  - **session.use_only_cookies=0** (desde PHP 4.3.0)
- Debe tenerse en cuenta que activar esta opción, es un potencial riesgo de seguridad ya que facilita un posible "secuestro de sesión".