Módulo 6: Cookies y Sesiones

Objetivos

- Conocer el concepto de galleta (*"cookie"*), su utilidad e implementación en páginas web.
- Conocer el concepto de sesión como método de mantenimiento de información común a una serie de páginas relacionadas.
- Saber cómo aplicar una sesión a un conjunto de páginas web.

Contenidos

En este módulo se expondrán algunos de los métodos que permiten mantener y disponer de una información común al proceso de navegación a través de una serie de páginas relacionadas.

Índice de apartados

Módulo 6: Cookies y Sesiones

Cookies

El primero de los mecanismos que vamos a estudiar, que nos permitirá mantener cierta información constante a lo largo del proceso de navegación por una web, serán las *cookies* (o si se prefiere, "galletas" en castellano).

Una *cookie* es una especie de variable que permite almacenar localmente informaciones diversas. Se almacenan en un fichero que puede contener diferentes *cookies*. Su finalidad es la de poder guardar información de un visitante, dejándola en su ordenador, para poderlas recuperar y utilizar en futuras visitas (si ya se ha visitado la página o no, fecha y hora de la última visita, etc.). Todas las *cookies* de una misma web se guardan secuencialmente en un mismo fichero de texto.

Acceso a las cookies

Dependiendo del Sistema Operativo en el que operemos y de su versión, y asimismo también teniendo en cuenta el navegador, encontraremos las *cookies* en un lugar u otro.

En el Windows XP, por ejemplo, el navegador "Internet Explorer" las guarda en *C:\Documents and Settings\NOMUSUARI\Cookies*.

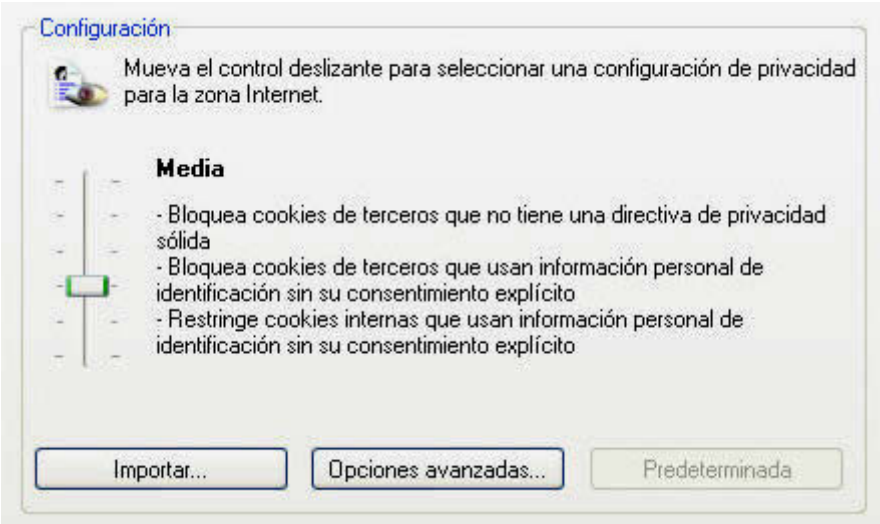


Figura 1. Internet Explorer

En el caso del "Mozilla Firefox", tanto para Windows como para Linux, se pueden ver accediendo a **"Editar | Preferencias | Privacidad | Cookies"**, y pulsando el botón "Ver Cookies", obtendremos una vista de las *cookies* almacenadas.

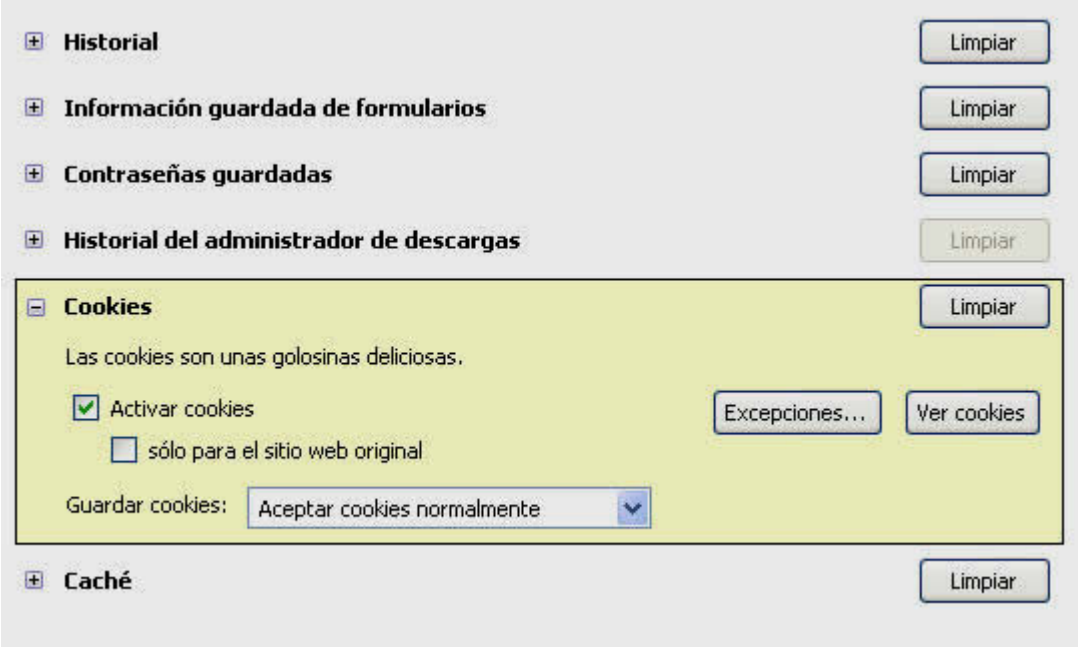


Figura 2. Firefox Mozilla

si pulsamos el botón " Ver cookies", dentro de la sección "Cookies" de la sección "Privacidad" en "Preferencias".

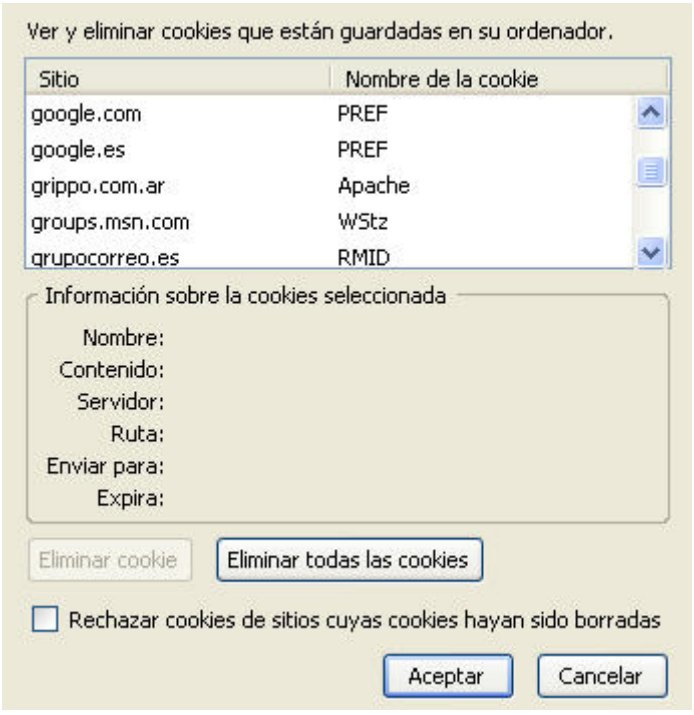


Figura 3. Cookies en Firefox Mozilla

Ejemplo de uso de cookies

La creación de *cookies* en PHP se hace a través de la función **setcookie()**. La descripción completa de su sintaxis, que conviene detenerse a leer, puede encontrarse en php.net .

En suma, para crear una *cookie* nos basta con darle un *nombre*, un *contenido* y un *momento de expiración* de la misma.

Tal y como se indica en la descripción de esta función, hay que tener presente que las cookies deben enviarse *antes* de mandar cualquier otra cabecera (esta es una restricción de las cookies, no de PHP). **Esto requiere que sitúe las llamadas a esta función antes de cualquier etiqueta *<html>* o *<head>*.**

El siguiente ejemplo pretende ilustrar su definición. Cópialo, guárdalo como *practica61.php* y ejécutalo en el navegador.

```
<?php
/*-------------------------------------------------------------
* Módulo: 6 Práctica: 1 Fichero: practica61.php
* Descripción: Creación de cookies
* Pre condi.: Cookies activadas en el navegador
* Post cond.:
-------------------------------------------------------------*/

// Momento de expiración de la cookie
// que será dentro de un año a partir de la hora
// actual que devuelve la función 'time()'

$expira = 365*24*3600;

// Creamos la cookie

setcookie("prueba","este es el contenido",time()+$expira);

echo "<h2>\"Cookie\" creada </h2><br<";
echo "Comprueba que la cookie ha sido creada";

?>
```

Como complemento al script propuesto, implementamos ahora otro que acceda a la *cookie*.

Cópialo, guárdalo como *practica61_b.php* y ejécutalo en el navegador.

```
<?php
/*-------------------------------------------------------------
* Módulo: 6 Práctica: 1 Fichero: practica61_b.php
* Descripción: Acceso a cookies
* Pre condi.: Cookie "prueba" creada
* Post cond.:
-------------------------------------------------------------*/

// Recuperamos el nombre de usuario de 2 formas

// 1) mediante el array '_COOKIE'

echo "Hola ".$_COOKIE["prueba"]." <br><br>";

// 2) utilizando la variable del mismo nombre directamente.

// NO FUNCIONARÁ POR EL PROBLEMA DE USO DE VARIBLES GLOBALES.

echo "Hola $prueba <br><br>";
 ?>
```

Cookies y Sesiones

La **sesión** en PHP aparece como tal, a partir de la versión 4. En este apartado, vamos a estudiarla, y veremos su utilidad y aplicación en el contexto de una web.

En primer lugar, veremos todos los aspectos teóricos que se deben conocer, para pasar finalmente a elaborar un ejemplo que exponga el funcionamiento de las sesiones.

¿Qué es una sesión?

Existen casos de aplicaciones web en las que, dadas sus características, se hace necesario disponer de la posibilidad de mantener cierta información a lo largo del proceso de navegación entre las páginas que la integran.

Ejemplos de esto podrían ser tanto la banca on-line, en la que cada usuario dispone de un área de navegación personalizada, como el típico "carrito de la compra", en el que el usuario selecciona y acumula una serie de artículos en el proceso de navegación entre una serie de páginas-catálogo. En estos casos es necesario relacionar las páginas que forman parte de esa área de navegación dependiente de una información común.

Visto que las comunicaciones que se establecen a través del protocolo HTTP son independientes las unas de las otras, lo cual cierra la posibilidad de establecer cualquier tipo de comunicación entre una serie de páginas, las sesiones vienen a cubrir este tipo de requerimiento, ofreciendo un mecanismo para el mantenimiento ubícuo de esa información compartida, al margen del funcionamiento de este tipo de protocolo.

¿Cómo implementa PHP las sesiones?

Una sesión, a groso modo, no es más que un conjunto de variables (que pueden ser concebidas como variables "globales") que PHP gestiona de manera transparante al usuario.

Una sesión comienza cuando un usuario entra en la aplicación web y termina cuando la abandona. Durante ese espacio de tiempo se pueden definir y manipular una serie de variables pertenecientes a la sesión que permite mantener cualquier tipo de información común para todas la páginas.

Para conseguir mantener una serie de información común entre la ejecución de los distintos scripts PHP que generan las páginas de la aplicación web, será necesario que el gestor de sesiones guarde dicha información en un fichero. Como trabajamos en un entorno cliente-servidor es posible que dicho fichero se encuentre en cualquiera de las dos partes.

Cada sesión que se inicia debe diferenciarse y ser independente del resto (ya que una página puede ser accedida al mismo tiempo por diversos usuarios), por lo que PHP asigna un identificador por cada sesión que se inicie, siendo este identificador una secuencia alfanumérica generada automáticamente lo suficientemente grande para garantizar la seguridad de la aplicación.

El identificador de sesión es necesario que sea reenviado al servidor cada vez que se realice una petición de una página, ya que de esta manera el gestor de sesiones podrá localizar la información correspondiente con la sesión. Esto se puede hacer implícitamente o explícitamente, dependiendo del mecanismo empleado.

- Si el identificador de sesión se almacena mediante "cookies" en el cliente, cuando se realice una petición al servidor el propio navegador será el encargado de introducir automáticamente el identificador de sesión como información dentro de la petición al servidor.

  Problema: el usuario podría tenerlas desactivadas,

- En el caso de no emplear cookies, nos obligaría a tener que pasar la información "manualmente", o mediante el *query_string* (en etiquetas <A>) o mediante un campo de tipo "hidden" (si venimos de un formulario).

PHP, por defecto, utiliza un sistema mixto de almacenaje de la información relacionada con una sesión. El identificador se almacena en el cliente utilizando cookies. El resto de información se guarda en un fichero dentro del servidor cuyo nombre es el identificador de sesión precedido del prefijo sess_.

¿Cómo se gestionan las sesiones desde código?

La gestión de sesiones se realiza de manera sencilla a través de un juego de funciones.

- **bool session_start()**: crea una nueva sesión diferenciando 2 casos:
  - si es la 1ª solicitud de sesión:
    - se genera un identificador aleatorio.
    - se crea un fichero en el servidor cuyo nombre es el identificador precedido del prefijo **sess_.**
    - se envía una cookie al cliente con el identificador.

    .
  - en sucesivas solicitudes, se utiliza la información ya dispuesta.

- **bool session_destroy()**: elimina los datos de sesion.
  - se destruye el fichero en el servidor.
  - se conserva la cookie para futuras sesiones.

- **bool session_register(string nombre [, string nombre])**: crea una serie de variables globales registradas como variables de sesión.
  - se guardan en el fichero de sesión correspondiente.
  - comunes y operativas en todos los scripts implicados en la sesión.

- **bool session_unregister(string nombre [, string nombre])**: las variables de sesión pasan a ser variables normales y se limpia el contenido del fichero.

- **bool session_is_registered(string nombre)**: se comprueba la existencia de una variable de sesión asociada a “nombre”.

- **session_unset()**: se “vacía” el contenido de las variables de sesión.

- **string session_name([string nombre])**: tiene 2 comportamientos:
  - si se invoca sin parámetro devuelve el nombre de la variable interna que contiene el identificador de sesión (por defecto se asigna el valor de la directiva *session.name* de "php.ini").
  - si existe parámetro, se cambia el contenido de dicha variable, pero será necesario hacerlo antes del comienzo de la sesión en cada script para prolongar su efecto.

- **string session_save_path([string path])**: devuelve el camino al directorio donde se ubican los ficheros asociados a la sesión.
  - es modificable, introduciendo una nueva ruta.
  - para prolongar su efecto, es necesario incluir esta función en cada script que haga uso de las sesiones.

¿Cómo iniciar y prolongar una sesión?

En **todas** las páginas que se desee que formen parte de la aplicación web, se debe ejecutar la acción de comenzar (o continuar) una sesión. Para ello, deberán utilizarse algunas de las funciones vistas. Pero ¿cómo consigue continuar una sesión?

Para cada una de las páginas, el intérprete de PHP comprueba si existe la sesión. En caso afirmativo, se retoma, sino, se crea una nueva (generando un nuevo identificador).

Existen **3** maneras de iniciar/prolongar una sesión:

- invocando a la función **session_start()**

```php
<?php

session_start();
echo "He inicializado la sesión";

?>
```

- a través de de la función **session_register()** (creación de variables).

```php
<?php

// Creamos la variable de sesión 'contador'

session_register('contador');

// La página enlaza consigo misma y muestra el identificador de sesión (variable interna $SID)
// y el valor del contador actualizado
```

```
echo '<a href="'.$PHP_SELF.'?'.$SID.'">Contador vale: '.++$contador.'</a>';

?>
```

- activando la directiva **session.auto_start** en el "php.ini".

Observación: Si no existiera, recordemos nuevamente que hay que crear una carpeta “tmp” e indicar en php.ini el path hasta ella en *session.save_path.*

Para poder continuar una sesión iniciada, cada vez que comienza una sesión se consulta si el script ha recibido un identificador de sesión o no. En caso afirmativo, se continua con la sesión que coincide con dicho identificador. En el otro caso , se crea una nueva sesión. Por lo tanto, es necesario comunicar el identificador de sesión a las distintas páginas que forman la aplicación web. Para ello, tenemos dos alternativas:

- Si usamos *cookies* (activando la directiva *session.use_cookies* del fichero php.ini) y además el navegador del cliente las acepta, no tenemos que realizar ninguna acción adicional.
- Si no usamos *cookies* o no las acepta el cliente, deberemos enlazar todas la páginas pasando el identificador en todos los enlaces a otras páginas y en todos los formularios que sean procesados en la aplicación.

En los enlaces a otras páginas dentro del query_string, deberemos incluir la variable que contiene el identificador, tenemos dos posibilidades:

1. <A href="pagina.php?<? echo SID ?>">enlace </A>
2. <A href="pagina.php?<? echo sesion_name(), '=', sesion_id() ?>">enlace </A>

En los formularios será necesario incluir un campo oculto cuyo nombre será sesion_name() y su valor sesion_id().

<input type="hidden" name="<? echo sesion_name() ?>" value ="<? echo sesion_id() ?>">

Ejemplo de funcionamiento

En el siguiente ejemplo vamos a exponer la dinámica de trabajo con sesiones a partir de un esquema de navegación de páginas determinado. El ejemplo consta de 3 páginas: la primera de ellas, cuyo código mostramos a continuación, es un formulario cuya recepción de datos se tramita en la misma página, generando las pertinentes variables de sesión.

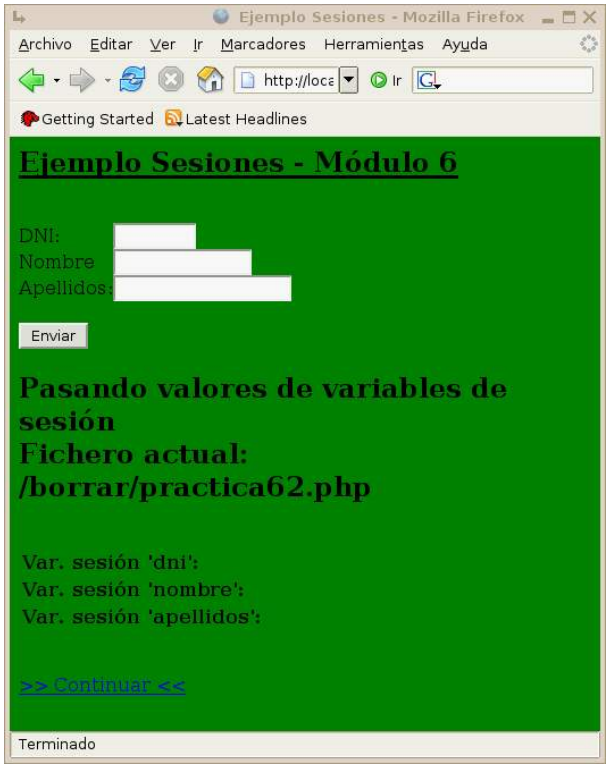
Copia el siguiente código y guárdalo como *practica62.php.*

```
<?php

// Con la función session_start(), iniciamos la sesión
session_start();

?>

<!----------------------------------------------------------
* Módulo: 6 Práctica: 2 Fichero: practica62.php
```

```
* Descripción: Formulario entrada de datos + creación
               variables sesión.
* Pre condi.:
* Post cond.:
------------------------------------------------------------>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Ejemplo Sesiones</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>

<h2><u>Ejemplo Sesiones - M&oacute;dulo 6</u> </h2>
<br>

<form name="form1" method="post" action="<? echo $PHP_SELF; ?>">
<table width="200" border="0" cellspacing="0" cellpadding="0">
<tr>
<td width="110">DNI: </td>
<td><input name="dni" type="text" id="codi5" size="8" maxlength="8"></td>
</tr>
<tr>
<td>Nombre</td>
<td><input name="nombre" type="text" id="any4" size="15" maxlength="15"></td>
</tr>
<tr>
<td>Apellidos:</td>
<td><input name="apellidos" type="text" id="etapa4" size="20" maxlength="20"></td>
</tr>
</table>
<br>
<input type="submit" name="enviar" value="Enviar">
</form>

</body>
</html>

<?php

echo "<script language='javascript'> document.body.style.backgroundColor='green'; </script>";

// Recogemos los valores enviados por el
// formulario con la ayuda del array _POST

$dni = $_POST['dni'];
$nombre = $_POST['nombre'];
$apellidos = $_POST['apellidos'];

// Mostramos los datos recibidos

echo "<h2>Pasando valores de variables de sesión<br>\n";
echo "Fichero actual: ".$_SERVER['PHP_SELF']."</h2><br>\n";

echo "<table>";
echo "<tr><td> <b>Var. sesión 'dni':</b> </td><td>".$dni."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'nombre':</b> </td><td>".$nombre."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'apellidos':</b> </td><td>".$apellidos."</td></tr>\n";
echo "</table>";
```

```
// Generamos la variables de sesión

session_register("dni");
session_register("nombre");
session_register("apellidos");

// Preparamos el enlace a la página siguiente pasándole
// mediante el 'query_string' el nombre de la
// variable de sesión más el identificador de sesión

echo "<br><br><a href=\"practica62_b.php?".session_name()."=".session_id();

echo "\">>> Continuar <<</a>";

echo "<br><br><a href=\"practica62_b.php?".SID;

echo "\">>> Continuar <<</a>";

?>
```

A continuación, pasamos a ver la página que utilizará las variables de sesión creadas anteriormente. Esta página se encarga de visualizar los datos de sesión.

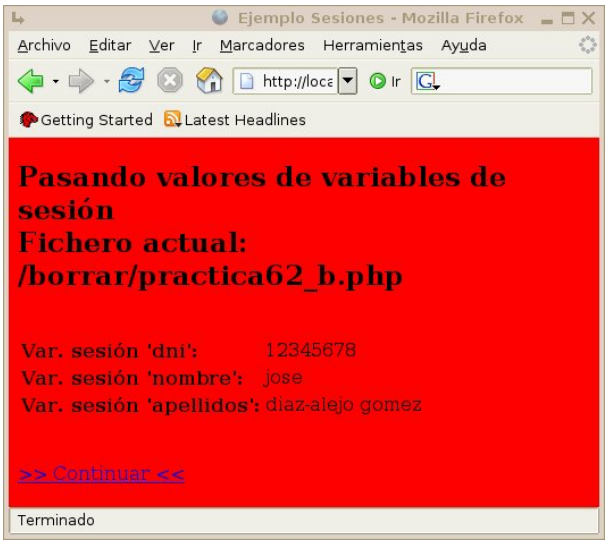


Observa que en las páginas siguientes utilizamos el array _**SESSION** para mostrar el valor de las variables de sesión (es tan solo una opción alternativa a haber utilizado el nombre de cada variable directamente).

Copia el siguiente código y guárdalo como *practica62_b.php*

```
<?
  session_start();
  // continuamos con la sesión
```

```
?>
<html>
<head>
<title>Ejemplo Sesiones</title>
</head>
<body>

<?php
/*-------------------------------------------------------------------
* Módulo: 6 Práctica: 2 Fichero: practica62_b.php
* Descripción: Paso de variables de sesión (2)
* Pre condi.: Datos de entrada generados en 'practica62.php'
* Post cond.:
-------------------------------------------------------------------*/

// Código Javascript para cambio de color de fondo

echo "<script language='javascript'> document.body.style.backgroundColor='red'; </script>";

echo "<h2>Pasando valores de variables de sesi&oacute;n<br>\n";
echo "Fichero actual: ".$_SERVER['PHP_SELF']."</h2><br>\n";

// Si la primera variable de sesión está definida...

if (session_is_registered("dni")){

// Mostramos los valores de sesión que han llegado
echo "<table>";
echo "<tr><td> <b>Var. sesión 'dni':</b> </td><td>".$_SESSION['dni']."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'nombre':</b> </td><td>".$_SESSION['nombre']."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'apellidos':</b> </td><td>".$_SESSION
['apellidos']."</td></tr>\n";
echo "</table>";
}else{

// Si la primera variable de sesión no está definida...

echo "Variables de sesión inexistentes \n";
}

echo "<br><br><a href=\"practica62_c.php?".session_name()."=".session_id();
echo "\">>> Continuar <<<</a>";

?>

</body>
</html>
```

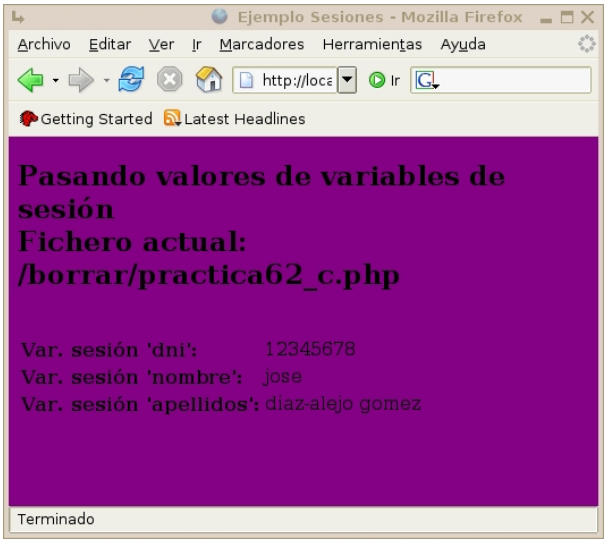


Por último , tenemos una página implicada en el esquema de navegación actual y que utilizará de nuevo las variables de sesión creadas anteriormente (fíjate que el código de esta página es prácticamente igual el de la anterior, por lo que puede ser útil recurrir al clásico "Copiar+Pegar")

Cópialo y guárdalo como *practica62_c.php*

```
<?// continuamos con la sesión

session_start();

?>
<html>
<head>
<title>Ejemplo Sesiones</title>
</head>
<body>

<?php
/*-------------------------------------------------------------
* Módulo: 6 Práctica: 2 Fichero: practica62_c.php
* Descripción: Paso de variables de sesión (3)
* Pre condi.: Datos de sesión sumistrados por 'practica62_b.php'
* Post cond.:
-------------------------------------------------------------*/

// continuamos con la sesión

session_start();

// Código Javascript para cambio de color de fondo

echo "<script language='javascript'> document.body.style.backgroundColor='purple'; </script>";

echo "<h2>Pasando valores de variables de sesión<br>\n";
echo "Fichero actual: ".$_SERVER['PHP_SELF']."</h2><br>\n";

// Si la primera variable de sesión está definida...

if (session_is_registered("dni")){

// Mostramos los valores de sesión que han llegado
```

```
echo "<table>";
echo "<tr><td> <b>Var. sesión 'dni':</b> </td><td>".$_SESSION['dni']."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'nombre':</b> </td><td>".$_SESSION['nombre']."</td></tr>\n";
echo "<tr><td> <b>Var. sesión 'apellidos':</b> </td><td>".$_SESSION
['apellidos']."</td></tr>\n";
echo "</table>";
}else{

// Si la primera variable de sesión no está definida...

echo "Variables de sesión inexistentes \n";
}

?>

</body>
</html>
```



Ejemplo de funcionamiento II

En este segundo ejemplo vamos a ver las directrices para crear un sistema de autenticación de usuario para un sitio web y restringir el acceso a las zonas que se consideren privadas.

El sistema consta de tres páginas, una para realizar la entrada introduciendo usuario y contraseña, una segunda para realizar la comprobación de los datos y una tercera que controlará durante la navegación. Además de las páginas privadas y una última para cerrar la sesión de una forma segura.

La primera página a la que llamaremos **login.html** es un formulario que solicita el usuario y la contraseña, una vez se pulse el botón de envio, se enviarán los datos por método POST a otra página llamada **comprobacion.php**.

En la página comprobacion.php se realizará una conexión con la base de datos y se comprobarán en la tabla usuarios que existe un usuario con ese nombre y clave. Una vez realizada la comprobación de que hay resultados se inicia una nueva sesión con **session_start ()**; y a continuación con **sesión_register()** se le pasa el valor **autentificado** que será el nombre de dicha sesión, y que es una variable global.

Copia el siguiente código y guárdalo como *comprobacion.php.*

```php
<?php

// Conexión con la base de datos

$conec= mysql_connect("servidor","usuario","password");

mysql_select_db("base_datos",$conec);

$sql = "SELECT * FROM usuarios WHERE nom=';

$sql .=$_POST['usuario'];

$sql .="and clave ='";

$sql .=$_POST['clave'];

$sql .= "'"

// echo $sql  //  (para comprobar la sql).

$resultado = mysql_query($sql,$conec);

if (mysql_num_rows($resultado)!=0)

{

  session_start();

  session_register("autentificado");

  $_SESSION['autentificado'] = "OK";

  header ("Location: zona.php");

}

else {

  header ("Location: login.html");

}

mysql_free_result($resultado);

mysql_close($conec);

?>
```

Si el usuario y contraseña existen se redirigirá con **header** a la página privada donde se quiere enviar al usuario. Por lo contrario, si ese usuario no existiera en el sistema, simplemente se le enviará a la página de entrada.

En todas la páginas privadas de la web, deberían contener un fichero que compruebe si hay una sesión activa para que pueda ser visializada. Para ello crearemos una página .php que se llamará **seguridad.php**.

Se recoge el valor y si es diferente a OK, entonces se considerará que esa entrada no es

válida pues no se ha iniciado la sesión y se le envía a la página de entrada.

```
<?
  session_start();
  // continuamos con la sesión
  if ($_SESSION["autentificado"] != "OK")
    {
     //no existe
     header("Location: login.html");
     exit();
     }
?>
```

Por último, la página de salida, a la que llamaremos **salir.php** en la que se destruye la sesión.

```
<?// continuamos con la sesión

session_start();

session_destroy();

?>

<html>

<head><title>SALIR</title></head>

<body>

Gracias por su visita <br><br>

< a href="login.html">Entrar de nuevo en la zona privada</a>

</body>

</html>
```

Todas las páginas que componen la zona privada deben incluir el archivo **seguridad.php**.

include (seguridad.php)

y deben terminar con una opción salir que enlace con la página **salir.php** para finalizar de forma segura la sesión iniciada.

En el caso de que el usuario no exista, se debería dar un mensaje de error, pero esto se pedirá en el ejercicio de este módulo.

Directivas de sesión en el fichero "php.ini"

En **php.ini** existe una sección para con una serie de directivas relativas a la gestión de las sesiones que nos permiten hacer ciertos ajustes sobre su comportamiento.

Las siguientes serían las más relevantes:

- **session.save_handler**: define el gestor de sesiones:
  - activado por defecto a *files***.**
  - posibilidad de definir otro gestión --> *user*
- **session.save_path**: especifica el parámetro que recibe el gestor para el almacenamiento y recuperación de los datos de sesión
  .
- **session.use_cookies**: fijada a 1 significa que el uso de cookies estará disponible.
- **session.use_cookies_ lifetime**: tiempo de validez de la cookie que contiene el identificador de sesión. Si está fijada a 0, la cookie se mantiene hasta que se cierre el navegador.
- **session.name**:indica el nombre de la variable que contiene el identificador de sesión. Se utiliza como nombre o como parámetro enviado por la página que invocó a la actual.
- **session.auto_star**t: fijada a uno inicializa la sesión automáticamente en cualquier script PHP ejecutado en el servidor.
- ...

Funciones para la gestión de sesiones

- **bool session_start()**: crea una nueva sesión diferenciando 2 casos:
  - si es la 1ª solicitud de sesión:
    - se genera un identificador aleatorio.
    - se crea un fichero en el servidor cuyo nombre es el identificador precedido del prefijo **sess_.**
    - se envía una cookie al cliente con el identificador.

    .
  - en sucesivas solicitudes, se utiliza la información ya dispuesta.

- **bool session_destroy()**: elimina los datos de sesion.
  - se destruye el fichero en el servidor.
  - se conserva la cookie para futuras sesiones.

- **bool session_register(string nombre [, string nombre])**: crea una serie de variables globales registradas como variables de sesión.
  - se guardan en el fichero de sesión correspondiente.
  - comunes y operativas en todos los scripts implicados en la sesión.

- **bool session_unregister(string nombre [, string nombre])**: las variables de sesión pasan a ser variables normales y se limpia el contenido del fichero.

- **bool session_is_registered(string nombre)**: se comprueba la existencia de una variable de sesión asociada a “nombre”.

- **session_unset()**: se “vacía” el contenido de las variables de sesión.

- **string session_name([string nombre])**: tiene 2 comportamientos:
  - si se invoca sin parámetro devuelve el nombre de la variable interna que contiene el identificador de sesión (por defecto se asigna el valor de la directiva *session.name* de "php.ini").
  - si existe parámetro, se cambia el contenido de dicha variable, pero será necesario hacerlo antes del comienzo de la sesión en cada script para prolongar su efecto.

- **string session_save_path([string path])**: devuelve el camino al directorio donde se ubican los ficheros asociados a la sesión.

- es modificable, introduciendo una nueva ruta.
- para prolongar su efecto, es necesario incluir esta función en cada script que haga uso de las sesiones.