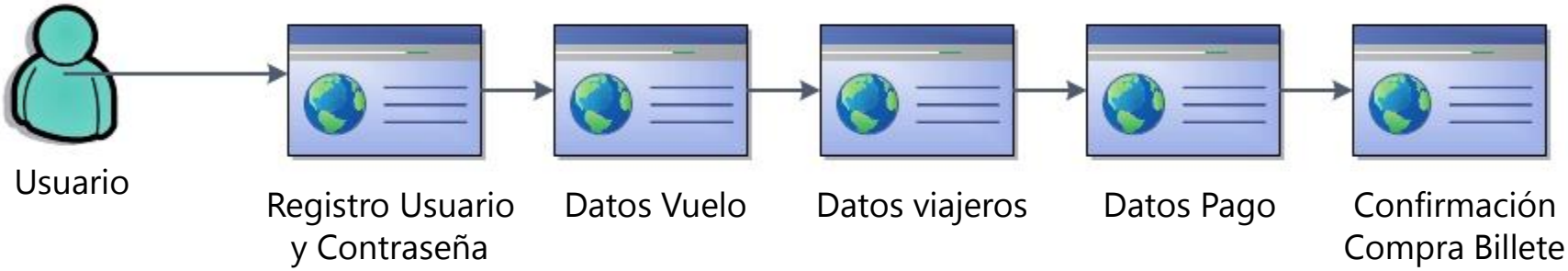
Desarrollo Web en Entorno Servidor

7. Cookies y sesiones.

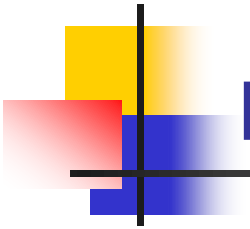
1. Las Cookies.
2. Sesiones.
3. Variables de sesión.

Control de Estado en Aplicaciones Web



- *App Web: Se navega por varias páginas web.*
- *¿Cómo guardar los datos que ingresa el usuario en cada una de las páginas?*
- *¿Cómo pasar la información para crear la salida final?*

- Objetivo: Garantizar conservación de los datos de un cliente que está realizando una operación en la Web.



Las Cookies. Definición

- Una cookie es un fragmento de información que un navegador web almacena en el disco duro del visitante a una página web. La información se almacena a petición del servidor web, ya sea directamente desde la propia página web con JavaScript o desde el servidor web mediante las cabeceras HTTP, que pueden ser generadas desde un lenguaje de web scripting como PHP. La información almacenada en una cookie puede ser recuperada por el servidor web en posteriores visitas a la misma página web.

- Las cookies resuelven un grave problema del protocolo HTTP: al ser un protocolo de comunicación "sin estado" (stateless), no es capaz de mantener información persistente entre diferentes peticiones. Gracias a las cookies se puede compartir información entre distintas páginas de un sitio web o incluso en la misma página web pero en diferentes instantes de tiempo

Las Cookies.

Las cookies se envían al cliente mediante encabezados HTTP. Como cualquier otro encabezado, las cookies se deben enviar antes que cualquier salida que genere la página (antes que <html>, <head> o un simple espacio en blanco). Si se intenta llamar a setcookie() después de haber realizado un echo, se producirá un error

Los usos más frecuentes de las *cookies* son:

- **Llevar el control de usuarios**: cuando un usuario introduce su nombre de usuario y contraseña, se almacena una *cookie* para que no tenga que estar introduciéndolas para cada página del servidor. Sin embargo una cookie no identifica a una persona, sino a una combinación de computador y navegador.
- **Ofrecer opciones de diseño (colores, fondos, etc) o de contenidos al visitante.**
- **Conseguir información sobre los hábitos de navegación del usuario, e intentos de spyware, por parte de agencias de publicidad y otros**. Esto puede causar problemas de privacidad y es una de las razones por la que las *cookies* tienen sus detractores.

¿Cómo enviar Cookies?

- Se usa la función:

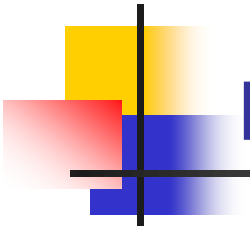
  setcookie(nom, val, exp)

  donde:

  - **nom** es una cadena que contiene el *nombre de la variable* que recoge el valor de la *cookie*.
  - **val** es el valor que *se asignará a la variable anterior*. Puede ser tanto numérico como de tipo cadena.
  - **exp** indica cuál ha de ser la *fecha de caducidad* de la *cookie*.

    Suele escribirse: **time()** (hora actual) **más** un número que representa los *segundos* que han de transcurrir hasta que la *cookie* expire.

- *La instrucción para el envío de cookies debe insertarse al principio de la página y antes de cualquier etiqueta HTML o línea en blanco.*
- *Es importante señalar que después de crear una cookie ésta no está disponible hasta la siguiente petición del cliente.*



Nota Importante

- Si ejecutas el ejemplo por primera vez observarás que solo aparecerá el texto *Esto es la cookie:* sin **ningún** valor y un error. Sin embargo, si actualizas el navegador o ejecutas más tarde el ejemplo siguiente aparecerá *Mi regalito* como valor de la variable.

La explicación es la siguiente: las instrucciones PHP se ejecutan en el servidor *antes* de enviar la página al cliente. Eso significa que, al ejecutar por primera vez, se inserta la orden de escritura y se comprueba el valor de la variable, que aun no ha sido creada y por ello aparece en blanco. Será en la actualización –ya se habría producido un envío al navegador y ya se habría escrito la cookie– cuando si se leerá el valor *anterior*.

Siempre que tratemos de visualizar el valor de una cookie estaremos viendo el valor asignado en la petición anterior

Ejemplos(1)

Ejemplo 1. (ejemplo1_cookies.php)

```
<?php
# setcookie escribe el contenido de la cookie
# en el ordenador del cliente
setcookie("cookie1","Mi regalito",time()+3600);
# escribe el valor leido en la cookie
echo "Esta es la cookie:",$_COOKIE['cookie1'];
?>
```

Ejemplo 2. (ejemplo2_cookies.php)

```
<?php
$value = 'Juan';
setcookie("nombre", $value, time()+3600);  /* expira en una hora */
echo "&iexcl;".'Hola ' .$_COOKIE["nombre"] . '!';;
?>
```

Ejemplos(2)

Un contador como aplicación práctica. (ejemplo3_cookies.php)

```
<? php
if (isset($_COOKIE['visitante']))
      $numero=$_COOKIE['visitante'];
else
      $numero=0;
$numero+=1;
setcookie("visitante",$numero,time()+86400);
if($numero==1){print "Es la primera vez que visitas esta página";}
if($numero>1){print "Es la $numero ª vez  que visitas esta página";}
?>
```

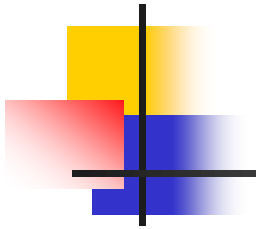
Evitando el error de la 1ª ejecución: (ejemplo4_cookies.php)

```
<?php
if (isset ($_COOKIE['entra'] ) )
  {
  if ($_COOKIE['entra']=="yes")
      {print "Gracias por regresar a mi sitio web de nuevo. <p>";
       setcookie ("entra", "otra vez", time () + 604800);}
  else
      print "Gracias por tu fidelidad.<p>";
  }

      else
      {
                print "Gracias por Probarnos por primera vez.<p>";
                setcookie ("entra", "yes", time () + 604800);

      }
?>
```

Ver las Cookies en el navegador

- En Chrome:
  - Configuración
    - Privacidad y Seguridad
      - Cookies y otros datos de sitios web
        - Ver Todas las cookies y los datos de sitios web

- En Mozilla Firefox:
  - Opciones
    - Privacidad & Seguridad
      - Cookies y datos del sitio
        - Administrar datos



Sesiones

- Sesión: Mecanismo que guarda información específica de un usuario que está usando una aplicación web.

  Un usuario = una sesión.

  La sesión está vigente durante las páginas web que navega el usuario.

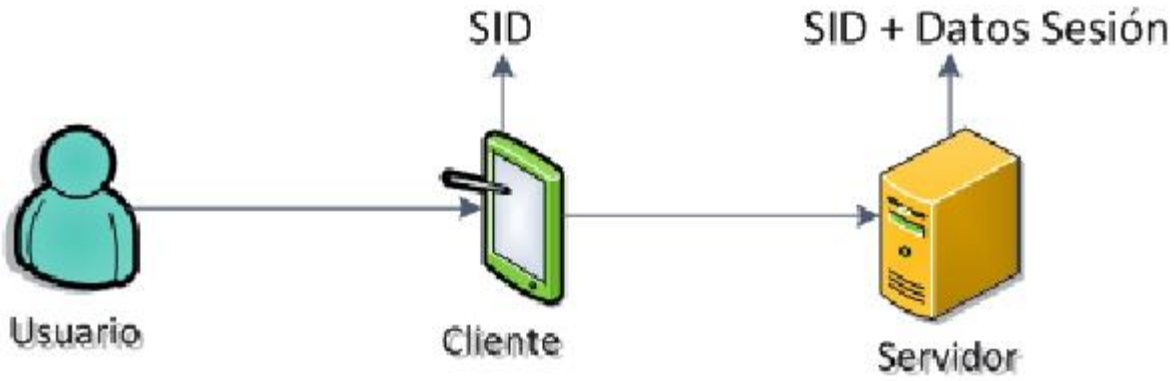
  Cada sesión se identifica con un SID.

  Modalidades:

  - Paso de datos a través de URL: Métodos POST o GET.
  - Cookies.

Sesiones

En el caso de las sesiones normales, el SID es enviado al cliente mientras que en el servidor se guarda toda la info de sesión.



PHP cuenta con un catálogo de funciones para el manejo de sesiones y de variables de sesión.

Sesiones. Funciones de Sesión

Funciones de Sesión:

- **session_start()**: Crea una sesión o continúa con la actual. En el segundo caso el identificador de sesión debe ser *transferido* por medio de una variable GET o a través de una *cookie*.

- **session_name()**: Recoge el nombre de la sesión. Si no se asigna uno de forma explícita, utiliza como nombre de sesión el contenido de la directiva **session.name** del fichero **php.ini**.

- **session_name('nombre')**: Esta función permite asignar un **nuevo nombre** a la sesión actual.

  Debemos tener en cuenta que si **cambiamos de página** y queremos mantener el mismo **identificador** (conservar la sesión anterior) esta función debe ser escrita, con el mismo nombre, en la nueva página, y además, ha de ser *insertada* **antes** de *llamar* a la función **session_start()** para que se inicie la sesión.

Sesiones

Funciones de Sesión:

- **session_cache_limiter()**: El limitador de caché controla las *cabeceras* HTTP enviadas al cliente.

  Estas *cabeceras* determinan las reglas mediante las que se habilita la opción de que los *contenidos* de las páginas puedan ser guardados en la caché local del cliente o se impida tal almacenamiento.

  En este último modo eriría una nueva llamada al servidor, lo cual tiene nvenientes

  Entre las ventajas est mos viendo la versión actualizada de la pág odo. El inconveniente es que requiere una nue o de espera, mientras el servidor produce la r

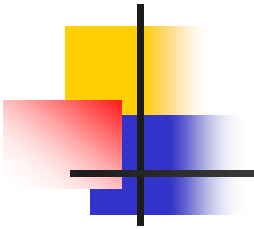
  Esta opción viene configurada *por defecto* en las directivas de configuración del **php.ini** como **nocache**.

  Para evitar –sea cual fuera la configuración de php.ini– el almacenamiento de las páginas en la caché del cliente hemos de utilizar:

  session_cache_limiter ('nocache,private')

El *orden* de escritura de estas instrucciones sería el siguiente:

```
<php?
session_cache_limiter();
session_name('nombre');
session_start();
.....
?>
```



Sesiones

Propagación de Sesiones:

- La verdadera utilidad de las sesiones es la posibilidad de que sean propagadas, es decir, que tanto el *identificador de sesión* como los *valores de las variables de sesión* -luego hablaremos de estas variables- puedan ir *pasando* de *una página a otra* sin necesidad de recurrir al uso de formularios.
- La forma habitual de *propagar* las sesiones es:
  - A través de **cookies**, pero como quiera que el usuario tiene la posibilidad de activar la opción *no aceptar cookies* y eso es algo que no podemos prever,
  - **PHP** dispone una opción alternativa que permite la *propagación* a través de la **URL**.

Sesiones

Caso de que el cliente tenga activada la opción aceptar cookies.

- Hacer la llamada a la nueva página siguiendo el método tradicional, es decir:

  <A href="pagxx.php">

- O desde un formulario

  ACTION="pagxx.php"

- Esa nueva página contendrá sin que lo preceda ninguna línea en blanco el *script* siguiente:

```
<?php
session_cache_limiter();
session_name('nombre');
session_start();
.....
?>
```

Caso de cookies deshabilitadas.

- tendremos que *pasar* el *identificador de sesión* junto con las llamadas a las páginas siguientes. Para ello se requiere la siguiente sintaxis:

```
<A href="pagxx.php?<?php echo session_name(). "=".session_id()?>
```

Ejemplos:
Fichero: ejemplo1_sesiones.php

```
<?php
session_start();
#pedimos que escriba el identificador único y el nombre de la sesión
echo session_id(),"<br>";
echo session_name(),"<br>";
?>
<A Href=" ejemplo1_sesiones.php">Volver a llamar a esta página</A>
```

Ejemplos:
Fichero: ejemplo2_sesiones.php

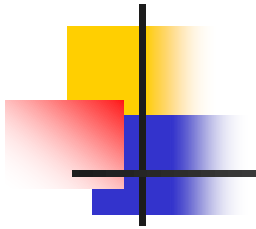
```
<?php
session_start();
#pedimos que escriba el identificador único y el nombre de la sesión
echo session_id(),"<br>";
echo session_name(),"<br>";
?>
<A Href="ejemplo2_sesiones.php?<?php echo session_name()."=".session_id()?>">
      Volver a llamar esta página</A>
```

Ejemplos:
Fichero: ejemplo3_sesiones.php

```
<?php
session_name('leocadia');
session_start();
#pedimos que escriba el identificador único
echo session_id(),"<br>";
echo session_name(),"<br>";
?>
<A Href="ejemplo3_sesiones.php">Volver a llamar esta página</A>
```

Ejemplos:
Fichero: ejemplo4_sesiones.php

```
<?php
session_cache_limiter('nocache,private');
session_name('leocadia');
session_start();
#pedimos que escriba el identificador único
echo session_id(),"<br>";
echo session_name(),"<br>";
?>
<A Href="ejemplo4_sesiones.php?<?php echo session_name()."=".session_id()?>">
 Volver a llamar esta página</A>
```



Variables de sesión

Variables de sesión

- La verdadera utilidad de trabajar con *sesiones* estriba en la posibilidad de **propagar** junto con ellas los *valores* de las *variables de sesión*.
- Se trata de *ir añadiendo y propagando* variables con sus valores, y de la posibilidad de utilizarlas de la misma forma que se utilizarían las *variables externas* enviadas a través de un *formulario*.
- Igual que ocurría en caso de los formularios, también las variables de sesión pueden ser tratadas de forma distinta según como estén configurado PHP (**php.ini**) y cual sea la versión de PHP que utilicemos.

Variables de sesión

Manejo de Variables.

- **$_SESSION['var']**.

- **unset($_SESSION)**: La función **unset** destruye las variables contenidas en el paréntesis. En este caso, al contener el array $_SESSION destruiría todas las variables contenidas en él.

- **unset($_SESSION['var'])**: Es similar a la anterior. En este caso solo sería destruida la variable de sesión indicada en *var*.

- **isset($_SESSION['var'])**: devuelve un valor *booleano* (UNO ó NUL) según que exista o no exista la variable contenida en el paréntesis.

Control de Estado en Aplicaciones Web. Sesiones en PHP

- Iniciar una sesión y crear variables de sesión:

```
<?php
//Iniciamos la sesión. Si la sesión ya existe se toma su
   SID
session_start();
echo 'Bienvenido a la página #1';
// El formato para inicializar una variable de sesión es:
// $_SESSION['nombre_var_sesión'] = 'Valor de la
   variable';
// A continuación, iniciamos tres variables de sesión
$_SESSION['favcolor'] = 'verde';
$_SESSION['animal'] = 'gato';
$_SESSION['time'] = time();
?>
```

Control de Estado en Aplicaciones Web. Sesiones en PHP

- `$_SESSION` = Vector con los datos de la sesión.

- Leer valores de variables de sesión:

```
<? php
 $_SESSION["nombre_de_variable_de_sesion "];
?>
```

- Ver si una variable ya existe:

```
<?php
if (isset($_SESION[“nombre_de_variable_de_sesión”])
?>
```

- Destruir una sesión:

```
session_destroy();
```

Ver fichero :ejemplo5_sesiones.php

Ejemplos

ejemploA.php

```
<?php
# desactivamos la opcion de que las páginas puedan guardarse en la cache del navegador
    del cliente
session_cache_limiter('nocache,private');
# le asignamos un nombre a la sesión aunque lo habitual sería dejar el nombre por defecto
# que le asigna la configuración de php.ini
session_name('pruebas');
# iniciamos la sesion
session_start();
# creamos variables de sesion y les asignamos valores
$_SESSION['valor1']=25;
$_SESSION['valor2']="Ambrosio de Morales";
$_SESSION['variable3']="Una prueba más";
/* cerramos el script e insertamos un enlace a otra página  y propagamos la sesión
    incluyendo en la llamada el nombre de la session y su identificador En esta página no se
    visualizaría nada. Solo el enlace */
?>
<A Href="ejemploB.php"> Propagar la sesion</A>
```

Ejemplos

ejemploB.php

```
<?php
session_cache_limiter('nocache,private');
session_name('pruebas');
session_start();

foreach($_SESSION as $indice=>$valor){
      print("Variable: ".$indice." Valor: ".$valor."<br>");
}

$_SESSION['valor1']+=87;
$_SESSION['valor2'] .=" bonito nombre";

unset($_SESSION['variable3']);

?>
<A Href="ejemploC.php">Propagar la sesion</A>
```

Ejemplos

ejemploC.php

```
<?php
# identicos comentarios a los anteriores
session_cache_limiter('nocache,private');
session_name('pruebas');
session_start();
# este bucle nos confirmará que se han propagado
# los nuevos valores y que la tercera variable ha sido destruida
foreach($_SESSION as $indice=>$valor){
      print("Variable: ".$indice." Valor: ".$valor."<br>");
}
?>
```