TEMA 3.5

Gestión de estado:

Cookies

HTTP: Protocolo sin estado

- HTTP es un protocolo **sin conexión**. Esto significa que el cliente y el servidor sólo se "ven" en el instante de un request o un response.
- HTTP no puede relacionar un request con otro anterior del mismo cliente. Al no haber conexión, no existe relación entre comunicaciones sucesivas.
- Por eso, se dice que HTTP es un protocolo **sin estado**.
- Entonces, ¿cómo conseguimos hacer aplicaciones tipo carrito de la compra?. ¿ Cómo sabe el servidor que la petición de añadir algo al carrito de la compra es mía y no de otro usuario?
- La **forma de mantener el "estado"**, es decir, de mantener esa información de relación entre diversos request, es artificial. Se utilizan dos herramientas:
  - **Cookies**
  - **Sesiones**

Cookies: Para qué se utilizan

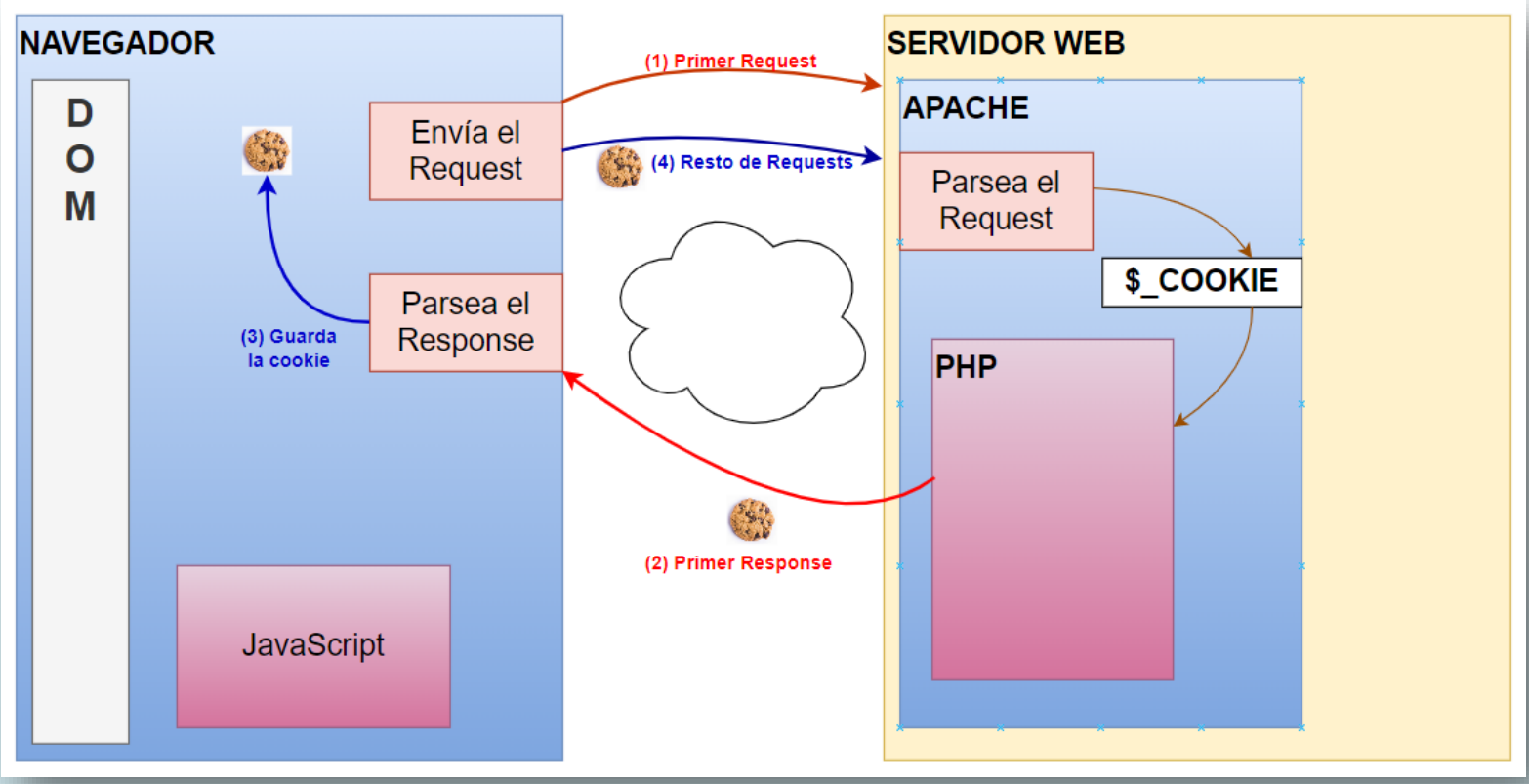
- Debido a las características de HTTP, cuando el servidor envía una página web al navegador, cierra la conexión y olvida todo respecto al usuario.
- Para resolver el problema de "cómo recordar la información del usuario", se inventaron las cookies.
- Cuando un usuario visita un sitio web, puede almacenarse su nombre en una cookie (por ejemplo, el login, o el email)
- La siguiente vez que el usuario visite la web, la cookie recuerda esa información.
- Las cookies suelen utilizarse para:
  - Recordar los inicios de sesión
  - Almacenar valores temporales del usuario
  - Almacenar el orden en que navega un usuario por nuestra web

Cookies en PHP: $_COOKIE

- Una **cookie**, o galleta, es un trozo de información, con un nombre y un valor.
- La cookie es *generada en el servidor*. Se envía al cliente en el response.
- Cuando un **navegador** recibe una cookie en un response, la almacena de la siguiente forma:
  - La cookie es un par **clave-valor**: El nombre de la cookie es la clave, y tendrá un valor. El navegador guarda ambos datos.
  - La cookie está **asociada** al dominio del servidor que la generó. El navegador guarda, una lista de servidores que enviaron cookies, y las cookies de cada uno.
  - La cookie tiene una **caducidad** asociada
  - El **navegador**, **siempre** que vaya a enviar un request a un servidor del cual guarde cookies, le enviará las cookies en el request
- El **servidor** parsea las posibles cookies que vengan en un request, y las guarda en la superglobal **$_COOKIE**, que es un array asociativo de tipo clave=>valor:
  - La clave es el nombre de una cookie
  - El valor será el valor de esa cookie

Cookies: Proceso de envío y almacenamiento

¡¡ Las cookies se guardan en el navegador !!



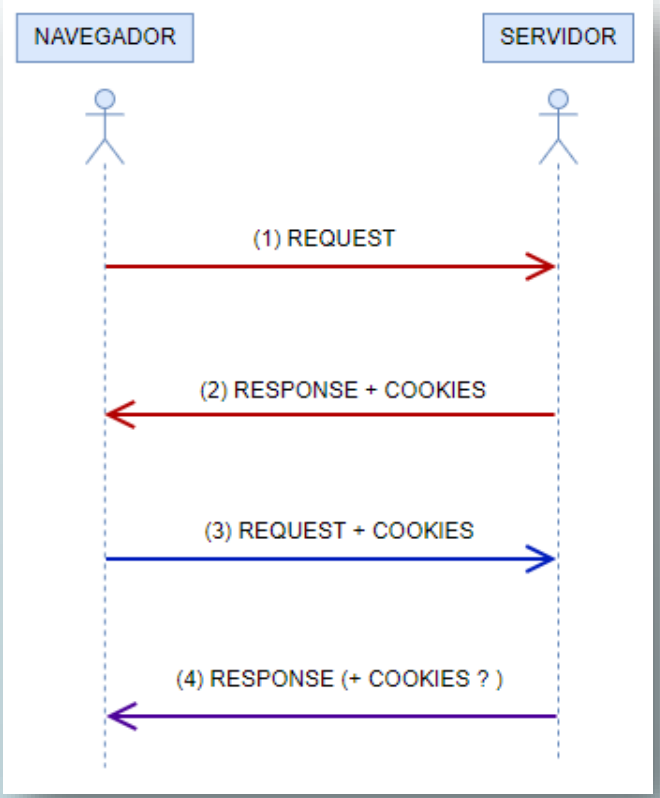
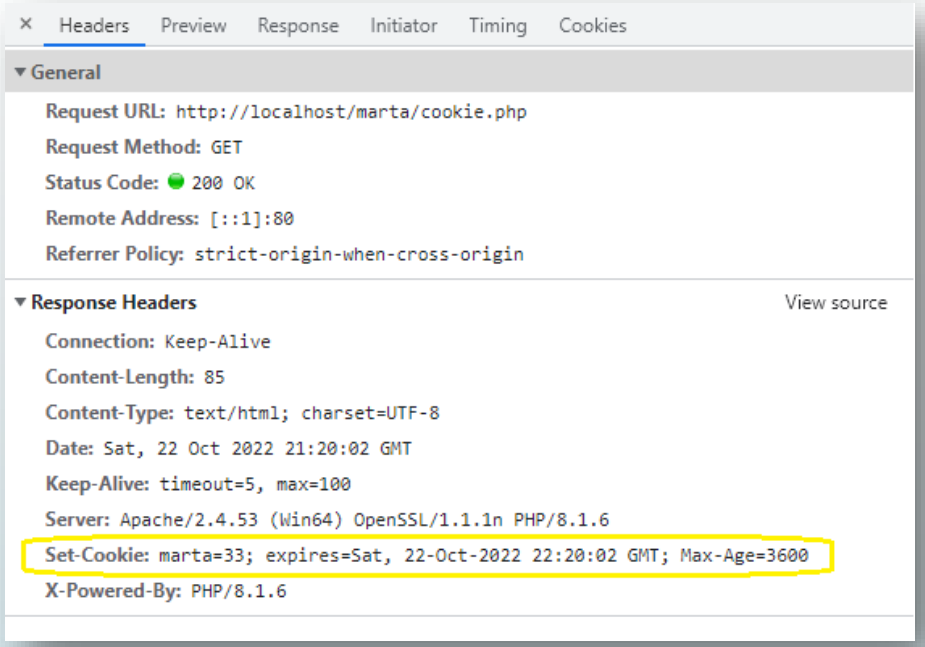


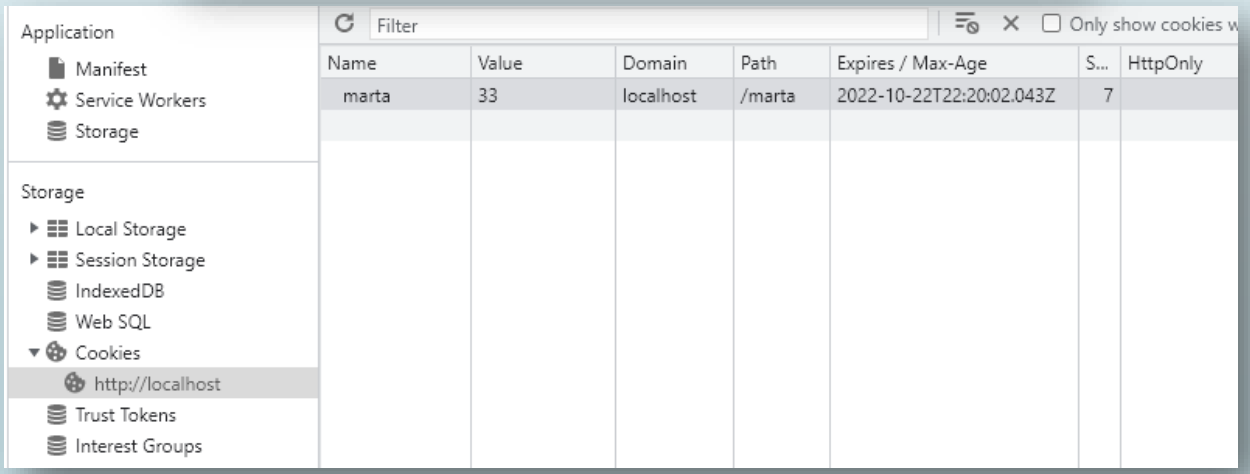
Diagrama de secuencia

(1) El navegador hace un primer request a la página del servidor web
(2) La página envía una cookie en el response
(3) El navegador parsea el response, ve la cookie, y la guarda con todas sus características
(4) El navegador, siempre que envíe un request a ese mismo servidor, adjuntará la cookie guardada

Visualización de cookies en el navegador:



- La imagen de la izquierda muestra la **pestaña de red** del inspector de Chrome.
- En la cabecera del **response**, podemos ver un ejemplo de recepción de cookie (set-cookie)
- La **cookie** se llama "marta", tiene valor "33", y caduca el sábado 22 de octubre de 2022 a las 22:20 hora de Greenwich (meridiano de Greenwich, hora de Londres). Su vida máxima es de 3600 segundos (1 hora)
- La cookie **será guardada por el navegador**, asociada al servidor que la envió.
- Arriba podemos ver que la Request se envió al servidor **localhost**.



- La imagen de la izquierda muestra la **pestaña de aplicación** del inspector de Chrome.
- En el apartado **Cookies**, hemos seleccionado el servidor **localhost** (izquierda)
- A la derecha vemos las cookies que ha guardado el navegador asociadas al servidorseleccionado (localhost)
- En este caso podemos ver la cookie llamada "marta", con valor "33" y fecha de caducidad 22 del 10 del 2022 a las 10:22 hora de Greenwich.

Navegadores: Limitación de almacenamiento

- Los navegadores tienen un límite al número de cookies que pueden guardar por cada dominio. Este límite varía con las versiones del navegador.
- Algunas cookies están diseñadas para desaparecer cuando se cierra el navegador.
- La siguiente tabla es un ejemplo de límites:

Browser Limitations.

Browser	Cookie count limit per domain	Total size of cookies
Chrome	180	4096
Firefox	150	4097
Opera	60	4096
Safari	600	4093

3 feb 2022

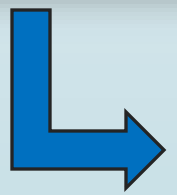
Cómo generar cookies con PHP: Ejemplo

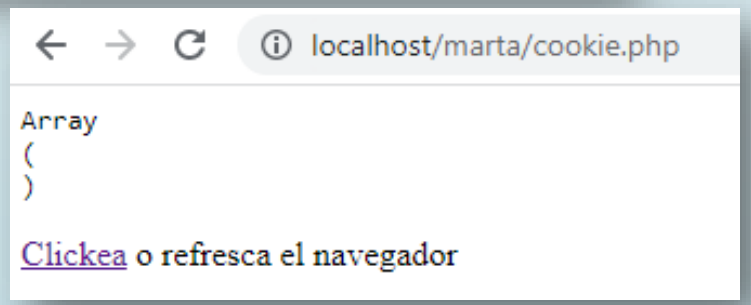
- El siguiente código genera una cookie llamada marta, con valor 33, y un límite de vigencia de 1 hora (3600 minutos)

Ejercicios clase > cookie.php

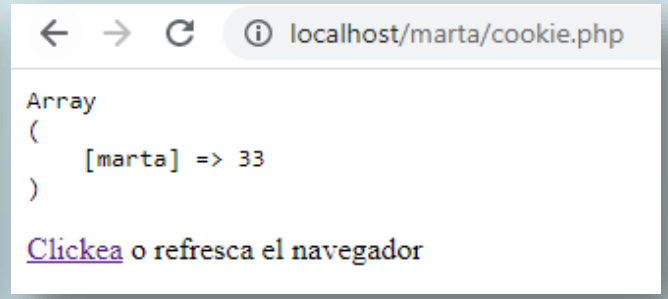
```php
<?php
if ( ! isset($_COOKIE['marta']) ) {
    setcookie('marta', '33', time()+3600);
}
?>

<!-- VISTA -->
<pre>
<?php print_r($_COOKIE); ?>
</pre>
<p><a href="cookie.php">Clickea</a> o refresca el navegador</p>
```

(1) La primera vez que hago un request a localhost/marta/cookie.php, $_COOKIE está vacío. En ese caso generamos una cookie con la función setcookie
(2) Las siguientes veces, el navegador mandará la cookie en el request, y el servidor genera un elemento para ella en $_COOKIE
El print_r muestra $_COOKIE





(1)



(2)

Función setcookie

- La función **setcookie** define una cookie para ser enviada junto al resto del encabezado del response de HTTP.
- Como otros encabezados, la cookie debe ser enviada **antes** que cualquier otra salida del script, incluso antes que la etiqueta <html>, incluso antes que cualquier espacio en blanco!!
- Por tanto, si queremos generar cookies, tenemos que hacer la llamada a setcookie **antes de generar cualquier salida**.
- Si no ponemos tiempo de caducidad al hacer setcookie, caducará cuando se cierre el navegador
- Si queremos borrar una cookie desde el servidor, hacemos un setcookie con tiempo de expiración en pasado, por ejemplo time() - 100

https://www.php.net/manual/es/function.setcookie.php

Ejercicio 1:

Cookies

1. Escribe un programa "tunombre.php" que genere una cookie de la siguiente forma:
   - El nombre de la cookie será el tuyo
   - El valor de la cookie será el mes de tu cumpleaños
   - La duración de la cookie será de tres horas
2. El programa mostrará en el navegador el contenido del arrays $_COOKIE en forma de tabla
3. Comprueba en el navegador que se ha guardado la cookie

Ejemplo de uso de cookies: Nº de visitas de usuario

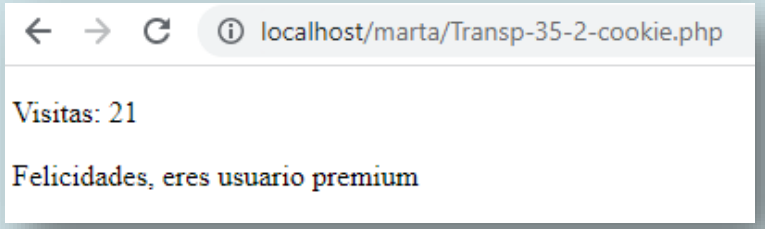
- Podemos utilizar una cookie para saber cuántas visitas hace un usuario a nuestra página
- En el ejemplo siguiente, la primera vez no hay cookies, por lo que $visitas será 0. Las siguientes veces habrá cookie, y $visitas será el valor de la cookie + 1
- En este ejemplo, siempre se va a enviar una cookie en el response. Como el nombre es el mismo, el navegador sustituirá el nuevo valor enviado

Ejercicios clase > Transp-35-2-cookie.php

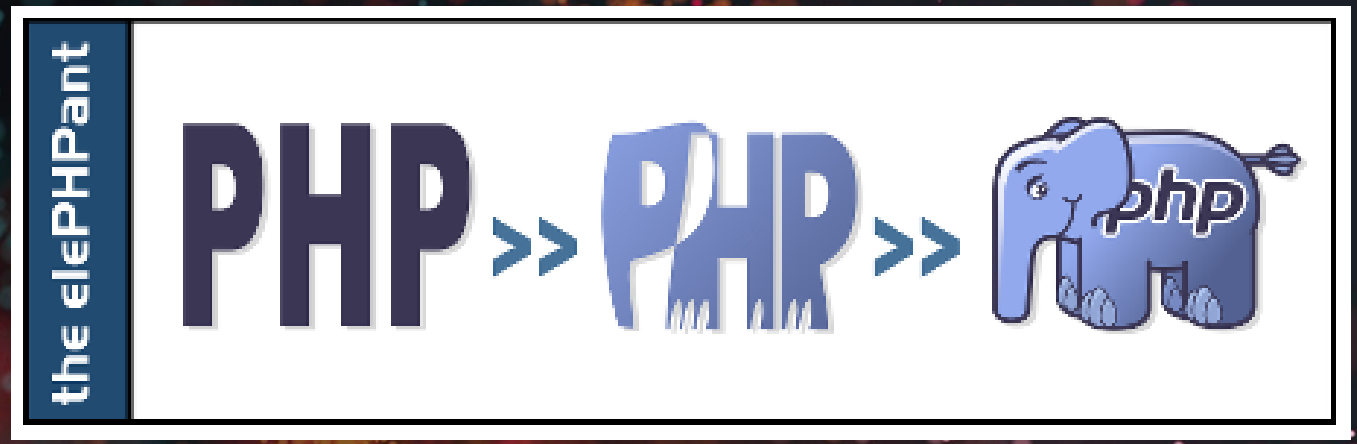
```php
<?php
  // Este programa genera una cookie llamada visitas, cuyo valor es el número de visitas
  // de un navegador a esta página.
  $visitas = 0;
  if (isset($_COOKIE['visitas'])) {
    $visitas = $_COOKIE['visitas']; // Recupero la cookie
    $visitas = $visitas+ 1; // Incremento en 1 el nº de visitas
  }
  setcookie("visitas", $visitas); // Vuelvo a generar la cookie, con el nuevo valor
  echo "El número de visitas es: $visitas";
?>
```

Ejercicio 2:

Cookies para contadores



1. Escribe un programa "cookie.php" que genere una cookie de la siguiente forma:
   - El nombre de la cookie será el tuyo
   - El valor de la cookie será 0
   - Caducará en una semana
2. Cada vez que se visite la página, el valor de la cookie se incrementará en "1"
3. Se mostrará en el navegador el número de visitas.
4. Si se alcanzan las 10 visitas, debajo del número de visitas aparecerá el mensaje: "Felicidades, eres usuario avanzado"
5. Si se alcanzan las 20 visitas, debajo del número de visitas aparecerá el mensaje: "Felicidades, eres usuario premium".
6. Si las visitas son menores de 10, aparecerá el mensaje: "Bienvenido, usuario"



https://www.php.net/docs.php