TEMPORIZADORES

El objeto window posee varios métodos relacionados con temporizadores. El manejo del tiempo es un elemento fundamental de la programación. Es fundamental en la programación de juegos, en la creación de animaciones, en publicidad y en otras muchas áreas.

Todas estas capacidades tienen que ver con dos métodos que permiten ejecutar un determinado código (normalmente por medio de una función callback) cuando pase cierto tiempo.

SETTIMEOUT

El método **setTimeout** de **window** tiene dos parámetros: el primero es la función que se ejecutará cuando se cumpla el tiempo, el segundo es un valor numérico que indica el tiempo a cumplir en milisegundos. Ejemplo:

```
setTimeout(()=>alert("Hola"),5000);
```

El primer parámetro del código anterior es la función flecha ()=>alert("Hola") esa función simplemente lanza el mensaje "Hola" en un cuadro de diálogo. El segundo parámetro indica que se esperará 5 segundos antes de ejecutar el código de la función.

El método setTimeout devuelve un número que identifica al temporizador en sí. Es decir, a cada temporizador lanzado con setTimeout se le asigna un número o identificador. Ese número se puede usar para cancelar el temporizador mediante **clearTimeout.**

Ejemplo:

```
var templ=setTimeout(()=>alert("Hola"),5000);
clearTimeOut(templ);
```

Tal cual está escrito el código anterior, jamás aparecerá el mensaje **Hola** porque se cancela el temporizador inmediatamente (no habrán pasado los 5 segundos).

SETINTERVAL

El método **setInterval** es muy similar al anterior. Tiene los mismos parámetros y devuelve también un identificador de temporizador que puede ser almacenado en una variable para poder cancelar el temporizador con un método llamado **clearInterval.**

La diferencia es que, en este caso, el código asignado al temporizador se invoca cada vez que pase el tiempo indicado. Es decir, setTimeout invoca al código una sola vez y setInterval lo invoca constantemente:

```
var temp2=setInterval(()=>alert("Hola"),5000);
```

El código es casi idéntico al que vimos en el apartado anterior. Pero ahora el cuadro de mensaje diciendo ***Hola*** aparecerá cada 5 segundos y no solo una vez como antes. Jamás dejará de aparecer a no ser que lo cancelemos:

```
var cont=0;
var temp2=setInterval(function(){
   alert("Hola");
   cont++;
   if(cont>=10) clearInterval(temp2);
},5000);
```

Este código muestra la palabra ***Hola*** en un mensaje cada 5 segundos. Lo hará 10 veces y luego ya no lo hará, ya que cuando el contador valga diez, se habrá eliminado el temporizador mediante clearInterval.

COOKIES

Las cookies no son más que una serie de archivos de texto que las aplicaciones web graban en el ordenador del usuario que contienen datos que la aplicación puede volver a recuperar. Dado que **http** es un protocolo sin estado, cada petición http es independiente de la anterior. Las cookies se envían automáticamente en la cabecera de cada petición permitiendo a las aplicaciones recordar aspectos de peticiones anteriores gracias a los datos que han grabado en las cookies.

Cada cookie puede grabar como mucho 4KB de datos y se permiten 20 cookies por dominio. Suelen ser suficiente estas restricciones con casi cualquier tipo de aplicación.

Todos los usuarios europeos de Internet han oído hablar de las cookies, ya que en la Unión Europea hay una directiva relativa a su uso que obliga a las aplicaciones web a pedir permiso al usuario antes de poder grabar cookies. Ese aviso tiene que incluir una explicación (o un enlace a la explicación) que aclare la utilización y necesidad que la aplicación requiere de las cookies.

En realidad, esta directiva distingue entre las cookies técnicamente necesarias y las no necesarias. Las primeras son imprescindibles para el correcto funcionamiento de la aplicación. Un ejemplo de este tipo de cookies son las que graban los datos de inicio de sesión del usuario que es clave para que ese usuario acceda a su carrito de la compra o a su espacio personal. Las cookies técnicamente necesarias, se permite que se graben directamente y que luego se pida permiso al usuario. Si el usuario deniega el permiso habría que borrarlas y, siendo imprescindibles, la aplicación dejará de trabajar con normalidad, pudiendo incluso redirigir al usuario a una página en blanco.

Las cookies que, según la UE, se consideran técnicamente no necesarias son las cookies de seguimiento y de análisis de la acción del usuario. Estas no se pueden utilizar antes de pedir permiso al usuario.

LECTURA Y GRABACIÓN DE COOKIES

El objeto **document** posee la propiedad **cookie** que es la que controla las cookies. A esta propiedad se asigna un texto que es el que graba en sí la cookie. La sintaxis básica es la siguiente:

document.cookie=”nombreCookie=valor”;

Ejemplo:

document.cookie=”usuario=Juan”; // se ha grabado una cookie con el nombre usuario y valor Juan

Cada nueva cookie puede utilizar el mismo formato, tendría otro nombre y el valor que le queramos dar. Si queremos modificar el valor de una cookie ya creada basta con indicar el mismo nombre e indicar el nuevo valor. Ejemplo:

document.cookie=”usuario=Ana”;

Para leer el contenido basta con ver el contenido de la propiedad cookie:

console.log(document.cookie);

Para realizar la lectura de cada cookie por separado se puede realizar usando el método split de los string el cual puede dividir las cookies usando el separador entre cada una (que es el punto y coma seguido de un espacio en blanco). Cada elemento del array resultante lo volveremos a dividir mediante el mismo método para separar el nombre y valor.

Ejemplo:

```
let arrayCookie=document.cookie.split("; ");
for(let cookie of arrayCookie){
  let [nombre,valor]=cookie.split("=");
  console.log(`La cookie "${nombre}" tiene el valor "${valor}" `);
}
```

FECHAS DE EXPIRACIÓN

Por defecto, las cookies se eliminan cuando el usuario finaliza su sesión. El fin de sesión ocurre cuando se cierra el navegador. Pero hay cookies que se requiere que sean persistentes; es decir, que sus valores aguanten entre sesión y sesión. Para ello hay que indicar una fecha de expiración futura. Esta fecha es el momento máximo en el que esa cookie sobrevive. Los navegadores eliminan las cookies que han caducado cada vez que el usuario inicia sesión en el dominio que grabó la cookie.

La fecha de expiración se debe indicar al modificar el valor de una cookie colocando un punto y coma tras el valor e indicando la palabra expires seguida de la fecha de expiración en formato UTC. La forma de grabar la fecha de espiración en JavaScript es obtener la fecha actual, después añadimos el tiempo de duración de la cookie. Añadir tiempo a las fechas mediante objetos de tipo Date nos obliga a hacerlo en milisegundos. Finalmente, la fecha ya incrementada se pasa a formato UTC.

Ejemplo:

```
//Crea un objeto de tipo fecha con la fecha actual
let hoy=new Date();
//crea una variable tomando los milisegundos de hoy
//y sumando los milisegundos de la semana que viene
let caducidadMs=hoy.getTime() + 1000 * 60 * 60 * 24 * 7;
//Convierte los milisegundos a un objeto Date que representa la fecha
//de dentro de una semana
let caducidad=new Date(caducidadMs);
//Se coloca la fecha de expiración en formato UTC, la cookie "usuario"
//caducará dentro de una semana
document.cookie = `usuario=Juan;expires=${caducidad.toUTCString()} `;
```

RUTA Y DOMINIO DE LAS COOKIES

Por defecto una cookie solo puede ser leída por aplicaciones del dominio que creó la cookie y que estén en el mismo directorio en el que se creó la cookie. Eso significa, por ejemplo que si una cookie se graba en un directorio como www.jose.net/dwec, los documentos de la raíz, fuera de la carpeta de dwec, no pueden leer esas cookies.

La propiedad **path** de las cookies permite indicar la ruta raíz desde la que cualquier aplicación puede leer esa cookie:

```
document.cookie="usuario=Jose;path=/";
```

Con la *propiedad **path***, hemos indicado que la raíz desde la que cualquier aplicación puede leer la cookie es "/" que es el nombre del directorio raíz del dominio. Es decir, en este caso,

cualquier documento del dominio puede leer esa cookie, ya que se incluyen los subdirectorios de la ruta.

Con el dominio ocurre lo mismo. Si hemos grabado la cookie en un subdominio y queremos que otro subdominio lea esa cookie, por defecto no podrá. Por ello, deberemos indicar un dominio más general. Por ejemplo:
document.cookie="usuario=Jose;path=/;domain=jose.net";

Al indicar como dominio: jose.net, aunque hayamos grabado la cookie en un documento del subdominio www.jose.net, un documento de subdominio.jose.net también las podrá leer.

BORRAR COOKIES
La forma de que una aplicación elimine una de sus cookies es indicar como fecha de expiración una fecha del pasado. Ejemplo:
Document.cookie=”usuario=Jose;expires=Sat, 01 Jan 2000 00:00:01 GMT”;

Al colocar como fecha el 1 de enero del año 2000, la cookie quedará caducada y el navegador la borrará.