IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

INTRODUCCIÓN AL ALMACENAMIENTO WEB – LOCALSTORAGE – OBJETOS JAVASCRIPT Y OBJETOS JSON

1 INTRODUCCIÓN:

La información es uno de los pilares de cualquier sistema informático y el elemento en torno al que giran el 99% de las aplicaciones. El objeto de una aplicación informática es reducir y automatizar las tareas, simplificando su gestión. La gran mayoría de las tareas involucra información y en ocasiones esta corresponde a volúmenes inmanejables por una persona.

Una aplicación, dentro de un navegador, en ocasiones requiera la posibilidad de almacenar datos de forma local, accesibles desde el navegador en todo momento. Esta tarea, originalmente se llevaba a cabo mediante el uso de pequeños ficheros almacenados en el disco duro del equipo del cliente, las denominadas **cookies**, cuyo uso derivó en software maligno. En los ficheros de **cookies** podíamos guardar información de la sesión de un usuario o bien ir más allá y recordar los gustos y preferencias de usuarios.

El volumen de datos actual es muy grande como para poder manejarlo con los pequeños archivos de **cookies**, de ahí la necesidad de utilizar en algunas ocasiones bases de datos o la caché del navegador para minimizar la información en transito en muchas ocasiones repetida y ya existente en el equipo local.

Otra de las ventajas de poder almacenar información en el cliente es la posibilidad que nos brinda de seguir trabajando incluso cuando no hay acceso al servidor, es decir, de forma off-line ya que tendríamos acceso a la información necesaria para seguir operando.

2 WEB STORAGE O ALMACENAMIENTO WEB JAVASCRIPT

Inicialmente el almacenamiento de datos usando Javascript era relativamente conflictivo: las variables desaparecían cuando se pasaba de una página web a otra. Hoy día existen herramientas que facilitan el trabajo a los programadores permitiendo la persistencia de la información sin necesidad de almacenarla en el servidor.

El almacenamiento web fue diseñado con el objetivo de mejorar y hacer más seguro el almacenamiento de información en el cliente y se realiza a través de pares clave/valor.

Existen dos tipos de almacenamiento proporcionados dentro del objeto window.

- **Por sesión**
- **Local**

Con la api Web Storage se utiliza el ordenador del usuario (su memoria o disco duro) para almacenar información útil para la navegación web. Esa información almacenada se recupera cuando es necesaria.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

- **Almacenamiento por sesión:** Este tipo de almacenamiento es utilizado para datos sin necesidad de una persistencia alta, y se utiliza a través del atributo **sessionStorage** del objeto window.

  Los datos son accesibles mientras dura la sesión de navegación. No son recuperables si:

  - Se cierra el navegador y se vuelve a abrir
  - Se abre una nueva pestaña de navegación independiente y se sigue navegando en esa pertaña
  - Se cierra la ventana de navegación que se estuviera utilizando y se abre otra.

- **Almacenamiento local:** El almacenamiento mediante esta técnica persiste más allá del cierre de sesión. El acceso al almacenamiento local se realiza a través del objeto **localStorage**, y la duración de la persistencia de la información va más allá del cierre de la pestaña/ventana del navegador.

3 COOKIES:

Una cookie HTTP, cookie web o cookie de navegador es una pequeña pieza de datos que un servidor envía a el navegador web del usuario. El navegador guarda estos datos y los envía de regreso junto con la nueva petición al mismo servidor. Las cookies se usan generalmente para decirle al servidor que dos peticiones tienen su origen en el mismo navegador web lo que permite, por ejemplo, mantener la sesión de un usuario abierta. Las cookies permiten recordar la información de estado en vista a que el protocolo HTTP es un protocolo sin estado.

Las cookies se utilizan principalmente con tres propósitos:

Gestión de Sesiones

- Inicios de sesión, carritos de compras, puntajes de juegos o cualquier otra cosa que el servidor deba recordar

Personalización

- Preferencias de usuario, temas y otras configuraciones

Rastreo

- Guardar y analizar el comportamiento del usuario

Las cookies se usaron una vez para el almacenamiento general del lado del cliente. Si bien esto era legítimo cuando eran la única forma de almacenar datos en el cliente, hoy en día se recomienda preferir las API de almacenamiento modernas.

Las cookies se envían con cada solicitud, por lo que pueden empeorar el rendimiento (especialmente para las conexiones de datos móviles), por ejemplo, mantener la sesión de un usuario abierta. Las cookies permiten recordar la información de estado en vista a que el protocolo HTTP es un protocolo sin estado.

Una cookie es una cadena de caracteres con la estructura nombre=valor que el navegador puede almacenar localmente en el ordenador del usuario a petición del servidor o del documento y que reenvía al servidor con cada nueva petición. Pueden ser leídas por el usuario y no deben guardar

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

información comprometida. Este tipo de información siempre tiene que guardarse en la sesión del servidor.

Las cookies solo permiten guardar cadenas de texto. Se encuentran asociadas a un único dominio y, por tanto, solamente serán enviadas aquellas que tengan relación sobre el dominio sobre el que se realizan las peticiones y no sobre aquellos donde no existe relación alguna.

Las **cookies.Js** se definen por una serie de parámetros, los cuáles se implementan a través de código Js.

Entre ellos está el **«nombre-valor»**, el cual contiene la información a almacenar, una fecha de expiración que determina la fecha de validez de la cookie o el dominio y la ruta a la cual se envía.

3.1 ¿Para qué se utilizan las cookies?

Entre otras cosas pueden recopilar información de nuestros gustos. Pongamos que estamos visitando un comercio electrónico para comprar un teléfono móvil. Pueden recopilar datos sobre determinados dispositivos que hemos buscado para ofrecernos publicidad orientada. También hay cookies analíticas, que sirven para elaborar estadísticas de los visitantes. Esto es algo que ayuda a los responsables de esa página.

"Al aceptar las cookies el usuario autoriza a los sitios web a rastrear información sobre sí mismo, como su actividad de navegación, su historial de búsqueda y sus datos de acceso

En la actualidad las páginas están obligadas a mostrar ese mensaje de advertencia, donde nos indican que pueden utilizar cookies.

3.2 ¿Cómo crear una cookie?

Bastaría con incluir la siguiente línea de código en un script:

```
document.cookie=”usuario=Dimas”;
```

3.3 ¿Cómo leer las cookies?

En JavaScript, se puede acceder a las cookies **a través de la propiedad cookie del objeto document** .

Esta propiedad permite acceder a todas las cookies de una página web, pero el acceso no es directo, ya que la propiedad cookie devuelve una única cadena que contiene todas las cookies de la página.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

4 ¿QUÉ ES LOCALSTORAGE?

Con el LocalStorage o HTML5 Web Storage podemos guardar información en nuestro navegador web a modo de sesión y que esa información persista y esté disponible durante la navegación entre las diferentes páginas de nuestro sitio o aplicación web.

El LocalStorage suele usarse mucho en aplicaciones web desarrolladas completamente con JavaScript, con tecnologías como Angular, aunque también puede aplicarse a cualquier web en la cual necesitemos compartir datos entre secciones.

En las aplicaciones web monolíticas desarrolladas con un lenguaje de backend como PHP o cualquier otro, el uso del LocalStorage es sustituido por las Sesiones del propio lenguaje de backend ya que el frontend y el backend están completamente integrados y mezclados.

Veamos como usar el LocalStorage en JavaScript.

1. **Primer paso: Comprobar si el navegador es compatible**

   Podemos ver si el navegador web tiene disponible la funcionalidad del localstorage así:

```
if (typeof(Storage) !== "undefined") {
    // LocalStorage disponible
} else {
    // LocalStorage no soportado en este navegador
}
```

2. **Segundo paso: Guardar datos en el navegado**r

   Para almacenar datos y guardar nuevos elementos o indices en el LocalStorage usaremos la siguiente instrucción:

```
// Guardar
localStorage.setItem("titulo", "Curso de Angular avanzado - Víctor Robles");
```

   De esta manera damos de alta un nuevo elemento en el almacenamiento del browser.

3. **Recuperar datos**

   Para conseguir los datos que tenemos guardados en un indice de nuestro almacenamiento local del navegador usaremos:

```
// Conseguir elemento

localStorage.getItem("titulo");
```

   Esto nos devolverá el valor que hemos guardado anteriormente.

4. **Guardar objetos JSON en el LocalStorage**

   Una manera de almacenar objetos es utilizando JSON. Como la representación de los objetos en JSON puede realizarse a través de texto, podemos almacenar estas cadenas de texto y recuperarlas posteriormente para convertilas en objetos.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Para guardar un objeto primero debemos convertirlo en un string json ya que el localstorage no permite guardar objetos de JavaScript como tal.

Tendríamos que hacer algo así:

localStorage.setItem("usuario", JSON.stringify(mi_objeto));

Guardamos el elemento usuario cuyo valor es un objeto convertido a string.

5. **Sacar objetos del LocalStorage**

   Para recuperar un objeto primero debemos convertirlo en un objeto de JavaScript json en lugar del string json que hay guardado por defecto.

   Haríamos esto:

   JSON.parse(localStorage.getItem("usuario"));

6. **Borrar o vaciar el LocalStorage**

   Para eliminar un elemento del localStorage haremos:

   localStorage.removeItem("titulo");

   Para eliminar todas las variables guardadas en el localStorage haremos:

   localStorage.clear();

7. **Eventos de almacenamiento:** Una de las funcionalidades más interesante de Web Storage es que incluye una serie de eventos que nos indican cuándo se ha producido un cambio en los datos almacenados. Esto quiere decir que los eventos para sessionStorage son lanzados en los iframe dentro de la misma página, o en las ventanas abiertas con window.open().

   Para localStorage, todas las ventanas abiertas con el mismo origen (protocolo + host + puerto) reciben los eventos.

   Cuando se lanzan los eventos, éstos contienen toda la información asociada con el cambio de datos:

```
StorageEvent {
   readonly DOMString key;
   readonly any oldValue;
   readonly any newValue;
   readonly DOMString url;
   readonly Storage storageArea;
};
```

   storageArea hace referencia al objeto sessionStorage o localStorage.
   Estos eventos se lanzan dentro del objeto window:

```
function handleStorage(event) {
   event = event || window.event; // support IE8
   if (event.newValue === null) { // it was removed
      // Do somthing
```

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es


```
    } else {
        // Do somthing else
    }
}
window.addEventListener('storage', handleStorage, false);
window.attachEvent('storage', handleStorage);
```

5 DIFERENCIAS ENTRE COOKIES Y STORAGE

Los objetos storage juegan un papel en alguna medida similar a las cookies, pero por otro lado hay diferencias importantes.

(a) Las cookies están disponibles tanto en el servidor como en el navegador del usuario, los objetos storage sólo están disponibles en el navegador del usuario.

(b) Las cookies se concibieron como pequeños paquetes de identificación, con una capacidad limitada (unos 4 kb). Los objetos storage se han concebido para almacenar datos a mayor escala.

6 OBJETOS JAVASCRIPT

En JavaScript, prácticamente todo es un objeto, las fechas, las expresiones regulares, los arrays, las funciones, etc. Salvo valores y tipos primitivos como false, true, null, etc.., los demás componentes del lenguaje son objetos en JavaScript.

Aunque se desaconseja (porque ralentizan el código y no añaden ninguna ventaja), se pueden declarar tipos primitivos como objetos. A continuación, se muestran las variables string, number y boolean de tipo objeto.

```
var a = new String();

var b = new Number();

var c= new Boolean();
```

Los objetos JavaScript son contenedores de valores con nombre denominados propiedades o métodos.

En JavaScriptt, pueden crearse objetos de dos formas, tal y como se indica a continuación. Aunque los dos códigos tiene la misma funcionalidad, el primero es más rápido, por lo tanto, es aconsejable utilizar la primera forma.,

- **1ª Forma:**

```
var person = {
        firstName: "John",
        lastName : "Doe",
        id : 5566
        };
```

- **2ª Forma:**

```
var person = new Object();
```

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es


```
person.firstName: "John",
person.lastName : "Doe",
person.id : 5566
```

También se pueden declarar los valores como pares nombre: valor (nombre y valor separados por dos puntos).

```
var car = {type:"Fiat", model:"500", color:"white"};
```

Los objetos también pueden tener métodos .

- Los métodos son acciones que se pueden realizar en objetos.
- Los métodos se almacenan en propiedades como definiciones de funciones .

```
var person = {
        firstName: "John",
        lastName : "Doe",
        id : 5566,
        fullName : function() {
        return this.firstName + " " + this.lastName;
        }
};
name = person.fullName();
```

Si accede a un método sin los paréntesis (), devolverá la definición de la función :

6.1 Recorrer la información de un objeto

En ocasiones, es necesario recorrer los campos de un objeto para poder procesarlos.

```
var car = {type:"Fiat", model:"500", color:"white"};
```

```
for (campo in car){
        alert (campo);
        alert(usuario[campo]); }
```

Ej2:

```
var usuario = {nombre:{first:'Pedro',last:'Salas'6},edad:30, esAdmin:true};
```

Las dos líneas siguientes mostrarían la misma información:

```
alert(usuario.nombre.first);
alert(usuario['nombre']['first']);
```

6.2 Constructores en JavaScript

El siguiente código muestra un constructor para la clase coche:

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es


```
var coche(modelo, color, kms, combustible) {
        this.modelo= modelo;
        this.color=color;
        this.kms=kms;
        this.combustible=combustible;
        }
};
var elmio=new coche("Mercedes", "Negro", 120000,"diésel");
var eltuyo= new coche("BMW", "blanco", 210000, "gasolina");
```

Si realizamos la siguiente sentencia

JavaScript no creará una copia del objeto, las variables de miBMV y el tuyo apuntarán al mismo objeto. Cualquier modificación de un atributo en una variable repercutirá en la otra.

```
var miBMV=eltuyo;
```

JavaScript puede añadir nuevos atributos , a pesar de que el objeto haya sido creado previamente. La siguiente línea funcionará sin problema alguno:

```
elmio.matricula="4321 JPH"
```

A continuación se muestra un objeto con algunos métodos setter y getter.

```
var coche(modelo, color, kms, combustible) {
        this.modelo= modelo;
        this.color=color;
        this.kms=kms;
        this.combustible=combustible;
        this.setmodelo=function(nuevomodelo){
                this.modelo=nuevomodelo;}

        this.getmodelo=function(){
                return this.modelo;
        }
};
```

6.2.1 Método estricto de crear una clase

El método **`constructor`** es un método especial para crear e inicializar un objeto creado con una `clase`. Solo puede haber un método especial con el nombre "constructor" en una clase. Si esta contiene mas de una ocurrencia del método `constructor`, se arrojará un *Error*

Un **constructor** puede usar la *palabra reservada* `super` para llamar al **constructor** de una *superclase*

Si no especifica un método constructor, se utiliza un constructor predeterminado.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es


```
class Square extends Polygon {
  constructor(length) {
    // Aquí, llama al constructor de la clase padre con sus longitudes
    // contemplando la anchura y la altura del Polígono
    super(length, length);
    // Nota: En las clases derivadas, super() se debe llamar primero
    // Se puede utilizar "this". Dejando esto causará un error de
    //referencia.
    this.name = 'Square';
  }
  get area() {
    return this.height * this.width;
  }
  set area(value) {
    this.area = value;
  }
}
```

Si no especifica un método constructor, se utiliza un constructor predeterminado. Para las clases base, el constructor por defecto es ***super***

6.3 MIEMBROS DE UNA CLASE

Una clase tiene diferentes características que la forman, que generalmente se denominan miembros, y que normalmente son de dos tipos: propiedades y métodos. Vamos a ir explicándolas detalladamente. Pero primero, una tabla general para verlas en conjunto, con sus tipos:

Elemento	Descripción	Más información
Propiedad	Variable que existe dentro de una clase. Puede ser pública o privada.	Ver propiedades
`Propiedad pública`	Propiedad a la que se puede acceder desde fuera de la clase.	
`Propiedad privada` ES2020	Propiedad a la que no se puede acceder desde fuera de la clase.	
`Propiedad computada`	Función para acceder a una propiedad con modificaciones (getter/setter).	
Método	Función que existe dentro de una clase. Puede ser pública o privada.	Ver métodos
`Método público`	Método que se puede ejecutar desde dentro y fuera de la clase.	
`Método privado` ES2020	Método que sólo se puede ejecutar desde dentro de la clase.	
`Constructor`	Método especial que se ejecuta automáticamente cuando se crea una instancia.	
`Método estático`	Método que se ejecuta directamente desde la clase, no desde la instancia.	
`Inicializador estático` ES2022	Bloque de código que se ejecuta al definir la clase, sin necesidad de instancia.	

Todas estas características se dividen en dos grupos:

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

- **Las propiedades:** variables dentro de clases
- **Los métodos:** funciones dentro de clases

La palabra clave **this** hace referencia al objeto instanciado.

```
function hello() {
        return this;
}
hello();                    // Window
const object = { hello }    // Metemos la función dentro del objeto
object.hello() === object;  // true
```

En este caso, podemos ver que si ejecutamos la función **hello()** en un contexto global, nos devuelve el padre, es decir, el objeto **Window**. Sin embargo, si metemos la función **hello()** dentro de un objeto , al ejecutar object.hello() nos devuelve el padre, es decir, el propio objeto object.

6.3.1 Métodos estáticos:

La palabra clave static define un método estático para una clase. Los métodos estáticos son llamados sin instanciar su clase y no pueden ser llamados mediante una instancia de clase. Los métodos estáticos son a menudo usados para crear funciones de utilidad para una aplicación.

6.4 CLASES EN FICHEROS EXTERNOS

Generalmente para tener el código lo más organizado posible, las clases se suelen almacenar en ficheros individuales, de forma que cada clase que creamos debería estar en un fichero con su mismo nombre.

```
// Animal.js
export class Animal
 {  /* Contenido de la clase */
}
```

Luego si queremos crear objetos basados en esta clase, lo habitual suele ser importar el fichero de la clase en cuestión y crear el objeto a partir de la clase.

```
// index.js
import { Animal } from "./Animal.js";
const pato = new Animal();
```

Si nuestra aplicación se complica, deberíamos crear carpetas para organizar mejor.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

7 JSON

JSON (JavaScript Object Notation) es un formato de texto muy sencillo utilizado para intercambio de datos. Es comúnmente utilizado para transmitir datos en aplicaciones web (por ejemplo: enviar algunos datos desde el servidor al cliente, así estos datos pueden ser mostrados en páginas web, o viceversa). Es una notación subconjunto de la utilizada en JavaScript, aunque hoy en día se utiliza* desde cualquier lenguaje y plataforma, sobre todo como alternativa a XML. Sus características básicas son:

- Los datos se almacenan en pares clave/valor, separados por el carácter dos puntos :
- Las claves y los valores de tipo cadena van encerradas entre comillas dobles
- Los datos se separan por comas ,
- Los arrays se almacenan entre corchetes [ ]
- Los objetos se almacenan entre llaves { }

Un objeto es un conjunto desordenado de pares nombre/valor. Un objeto comienza con {llave de apertura y termine con } llave de cierre. Cada nombre es seguido por :dos puntos y los pares nombre/valor están separados por ,coma.

Generalmente un archivo JSON ocupa menos espacio que su análogo XML, y es mucho más sencillo de analizar (en XML necesitamos un "parser" (analizador) XML en cambio para JSON basta con una sencilla función).

- JSON es sólo un formato de datos — contiene sólo propiedades, no métodos.
- JSON requiere usar comillas dobles para las cadenas y los nombres de propiedades. Las comillas simples no son válidas.
- Una coma o dos puntos mal ubicados pueden producir que un archivo JSON no funcione.Se debe ser cuidadoso para validar cualquier dato que se utilizar (aunque los JSON generados por computador tienen menos probabilidades de tener errores, mientras el programa generador trabaje adecuadamente). Es posible validar JSON utilizando una aplicación como JSONLint.
- JSON Puede tomar la forma de cualquier tipo de datos que sea válido para ser incluido en un JSON, no sólo arrays u objetos. Así, por ejemplo, una cadena o un número único podrían ser objetos JSON válidos.
- A diferencia del código JavaScript en que las propiedades del objeto pueden no estar entre comillas, en JSON, sólo las cadenas entre comillas pueden ser utilizadas como propiedades.

Un archivo JSON mínimo debe tener la siguiente sintaxis:

```
{
}
```

Esto simplemente es un objeto vacío. Un archivo JSON, puede contener varios tipos de datos:

```
{
```

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es


```
"name": "Manz",
"life": 99,
"dead": false,
"props": ["invisibility", "coding", "happymood"],
"senses": {
"vision": 50,
"audition": 75,
"taste": 40,
"smell": 50,
"touch": 80
}
}
```

Como se puede ver, en JSON todos los textos deben estar entrecomillados con «comillas dobles», y solo se pueden utilizar tipos de datos como , , , , o null. Un valor null, simplemente, también sería un JSON válido.

OJO: JSON no permite utilizar tipos de datos como , , o valores undefined. Tampoco es válido incluir comentarios en un JSON.

(a) **¿Cómo utilizar JSON?:** Si analizamos bien la sintaxis de un JSON, nos daremos cuenta que es muy similar a algo a lo que ya deberíamos estar acostumbrados:

```
const o = {
name: "Manz",
life: 99,
};
```

Simplemente añadiendo const o = al principio, nos daremos cuentade que se trata de un objeto de Javascript y que no debería ser muy sencillo pasar de JSON a Javascript y viceversa.

En Javascript tenemos una serie de métodos que nos facilitan esa tarea, pudiendo trabajar con que contengan JSON y objetos Javascript de forma indiferente:

Método	***Descripción***
JSON.parse(str)	Convierte el texto str (un JSON válido) a un objeto y lo devuelve
JSON.stringify(obj)	Convierte un objeto Javascript obj a su representación JSON y la devuelve

(b) **Convertir JSON a Objeto:** La acción de convertir JSON a objeto Javascript se le suele denominar parsear. Es una acción que analiza un elemento que contiene un JSON válido y devuelve un objeto Javascript con dicha información correctamente estructurada. Para ello, utilizaremos el método JSON.parse():

```
const str = '{ "name": "Manz", "life": 99 }';
const obj = JSON.parse(str);
obj.name; // 'Manz'
obj.life; // 99
```

IES de Teis
Avda. de Galicia, 101
36216 – Vigo
Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Como se puede ver, obj es un objeto generado a partir del JSON recogido en la variable str y podemos consultar sus propiedades y trabajar con ellas sin problemas.

(c) **Convertir Objeto a JSON:** La acción inversa, convertir un objeto Javascript a JSON también se puede realizar fácilmente haciendo uso del método JSON.stringify(). Este método dificil de pronunciar viene a ser algo así como «convertir a texto», y lo podemos utilizar para transformar un objeto de Javascript a JSON rápidamente:

```
const obj = {
name: "Manz",
life: 99,
saludar: function () {
return "Hola!";
},
};
const str = JSON.stringify(obj);
str; // '{"name":"Manz","life":99}'
```

Observa que, como habíamos dicho, las funciones no están soportadas por JSON, por lo que si intentamos convertir un objeto que contiene métodos o funciones, JSON.stringify() no fallará, pero simplemente devolverá un omitiendo las propiedades que contengan funciones.

(d) **Leyendo JSON externo:** Teniendo en cuenta todo lo visto hasta ahora, JSON es un formato ideal para guardar en pequeños archivos de texto que se puedan leer desde Javascript, pasar a objetos y trabajar con ellos.

8 FTECH

La API Fetch proporciona una interfaz JavaScript para acceder y manipular partes del canal HTTP, como peticiones y respuestas. También provee un método global fetch() que proporciona una forma fácil y lógica de obtener recursos de forma asíncrona por la red.

Este tipo de funcionalidad se conseguía previamente haciendo uso de XMLHttpRequest. Fetch proporciona una mejor alternativa que puede ser empleada fácilmente por otras tecnologías

Una petición básica de fetch es realmente simple de realizar.

```
fetch('http://example.com/movies.json')
.then(function(response) {
return response.json();
})
.then(function(myJson) {
console.log(myJson);
});
```

Aquí estamos recuperando un archivo JSON a través de red e imprimiendola en la consola. El uso de fetch() más simple toma un argumento (la ruta del recurso que quieres obtener) y devuelve un objeto Promise conteniendo la respuesta, un objeto Response.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Una petición promise fetch() será rechazada con TypeError cuando se encuentre un error de red, aunque esto normalmente significa problemas de permisos o similares — por ejemplo, un 404 no constituye un error de red.

- **El método THEN:** El método then() retorna un Promise. Recibe dos argumentos: funciones callback para los casos de éxito y fallo de Promise

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Es muy probable que más de una vez has aceptado unas cuantas cookies informáticas sin tener del todo claro qué son o para qué sirven exactamente.

El mensaje cuando navegamos por internet suele ser el siguiente:

"Las cookies nos permiten ofrecer nuestros servicios. Al utilizar nuestros servicios, aceptas el uso que hacemos de las cookies".

Y a continuación, dos opciones: "Aceptar" o "Más información".

4 cosas que quizás no sabías sobre las cookies

Cómo averiguar todo lo que Google sabe de ti

La mayoría de las veces -en muchos casos porque necesitamos ver o algo en la red y no tenemos mucho tiempo- simplemente, las aceptamos.

Pero, ¿qué son exactamente las cookies y por qué siempre nos preguntan por ellas? ¿Qué aceptamos cuando decimos que sí a las cookies?

Programas-espía

Al contrario de lo que algunos piensan, las cookies no son spam, ni gusanos informáticos, ni ningún otro tipo de virus extraño.

Son unos archivos informáticos diminutos enviados por los sitios web que se almacenan en nuestro navegador y que obtienen datos sobre nosotros.

TIPOS DE COOKIES SEGÚN SU FINALIDAD

* Técnicas: controlan el tráfico, identifican sesiones, almacenan contenidos...

* De personalización: idioma, tipo de navegador, configuración regional.

* De análisis: siguen el comportamiento de los usuarios para medir actividad del sitio.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

* Publicitarias: permiten la gestión de espacios publicitarios que el editor incluyó en web.

* De publicidad comportamental: crean un perfil específico del usuario.

Fuente: Agencia Española de Protección de Datos (AEPD)

Getty

Estos pequeños programas-espía consiguen información clave para la publicidad en internet, especialmente en lo que respecta a los avisos publicitarios personalizados.

La página en la que puedes ver cómo Facebook te analiza para enviarte anuncios personalizados

El trabajo de las cookies es "contarles" a las marcas y empresas cómo nos comportamos en internet para colocar anuncios de acuerdo con nuestros gustos e intereses.

Entre otras cosas, pueden recabar este tipo de información:

direcciones y contraseñas del correo electrónico

nuestro número de teléfono y dirección

nuestra dirección de IP

el sistema operativo de nuestra computadora

el navegador que utilizamos

páginas que hemos visitado anteriormente

Pueden ser propias o de terceros; temporales o permanentes.

Las propias se generan en la web que estamos visitando, y las de terceros pertenecen a una página externa, normalmente a los anunciantes.

Las temporales o "de sesión" sólo duran mientras tenemos una sesión abierta en el navegador. Cuando la cerramos, desaparecen.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Sin embargo, para hacer que las permanentes o "persistentes" dejen de recibir información sobre nosotros debemos borrarlas manualmente de nuestro buscador.

Según un reporte de la Unión Europea sobre protección de datos que analizó cerca de 500 páginas web, el 70% de las cookies son de terceros y rastrean nuestra actividad para ofrecernos publicidad personalizada.

Cómo las marcas te espían por internet para enviarte publicidad personalizada y cómo evitarlo

cookies

Pueden ser propias, de terceros, temporales o permanentes.

Otras sirven para personalizar el servicio que nos ofrece el sitio web, en función de nuestro navegador o la manera en la que usamos los datos.

Y otras son "técnicas" y sirven para controlar el tráfico, identificar el inicio de sesión del usuario, almacenar contenidos o permitir el uso de elementos de seguridad.

Aunque la categoría no es exclusiva: una sola cookie puede llegar a tener varias finalidades.

Ventajas y desventajas

La mayoría de las páginas web nos obligan a aceptarlas para poder seguir usando el servicio -por eso están obligadas a informarnos bien sobre ellas- aunque existen maneras de desactivarlas y bloquearlas.

Si no quieres que la información sobre ti queden almacenada en tu equipo, puedes eliminarlas en la sección "Herramientas" y después hacer clic en "Borrar los datos de navegación".

En el teléfono, encuentras esta opción en "Configuración", después en "Privacidad" y, finalmente, en "Borrar cookies".

Pero no siempre es necesario rechazarlas; la política de cookies tiene sus ventajas y desventajas.

IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
Fax: 886 12 04 85
e-mail: ies.teis@edu.xunta.es
http://www.iesteis.es

Por un lado, pueden ser de gran ayuda para mejorar nuestra experiencia en internet, creando un perfil de usuario y evitando que tengamos que rellenar formularios, contraseñas e interminables hojas de contacto una y otra vez.

navegador

Las cookies también acumulan información sobre las web que visitas.

"Las cookies son archivos creados por los sitios web que visitas y la caché de tu navegador, la cual ayuda a que las páginas se carguen más rápido. Te permiten navegar más fácilmente por la web", explica Google en su blog.

"Si vacías la caché y eliminas las cookies de tu navegador, se borrará la configuración de sitios web (como los nombres de usuario y las contraseñas) y es posible que algunos sitios funcionen más lentamente, dado que todas las imágenes deben cargarse de nuevo".

El problema es cuando hay un abuso y obtienen datos personales de los usuarios (sobre todo, cuando ocurre sin su consentimiento), algo que organismos como la Comisión Europea han criticado recientemente.

La compañía estadounidense de software informático Vertical Response dice que es conveniente eliminar la caché y las cookies de vez en cuando para "limpiar" el navegador.

"Tendrás que volver a escribir los nombres de usuario y contraseñas, pero tu privacidad estará más a salvo y tu navegador trabajará mejor".