Introducción

Con el almacenamiento web, las aplicaciones web pueden almacenar datos localmente dentro del navegador del usuario.

Antes de HTML5, los datos de la aplicación tenían que ser almacenados en cookies, incluidos en cada petición al servidor, lo que causaba peores rendimientos a la pagina web.

Antes de usar esta funcionalidad debemos comprobar que nuestro navegador es compatible con ella con el siguiente código, aunque la gran mayoría de los navegadores actuales son compatibles:

API					
Web Storage	4.0	8.0	3.5	4.0	11.5

```
if (typeof(Storage) !== "undefined") {
  // El navegador es compatible con la tecnología.
} else {
  // El navegador no es compatible con la tecnología.
}
```

Los datos están almacenados en pares de clave/valor.

Para almacenar un dato en el navegador es tan simple como acceder a ***localStorage o sessionStorage*** como un objeto o llamar a alguno de los métodos que posee dicho objeto.

Se puede usar las siguientes nomenclaturas:

- Para asignar:

  localStorage.setItem("lastname", "Smith");

  localStorage.lastname = "Smith";

- Para acceder:

  localStorage.getItem("lastname");

  localStorage.lastname;

- Para iterar:

  localStorage.length;

  Se le pasa un numero, en caso de que exista algo con ese indice lo devuelve, en otro caso devuelve null.

  localStorage.key(0);

· Para eliminar:

localStorage.removeItem("lastname");

delete localStorage.lastname;

Este borra todos los datos almacenados con esta tecnología.

localStorage.clear();

Como nota añadir que los valores almacenados con esta tecnología se guardan como una cadena (String) que deberá ser convertida a otro formato en caso de ser necesario. Por ejemplo podríamos guardar un objeto usando **`JSON.stringify(mi_objeto)`** y así poder guardar objetos en dichas variables.

El almacenamiento en el navegador en el navegador tiene 2 tipos de almacenamiento que pasaré a comparar entre si y luego con las cookies:

LocalStorage vs SessionStorage

Los datos almacenados por ***localStorage*** no expiran, es decir, se conservan incluso después de cerrar el navegador, pudiendo ser consultada el siguiente día, mes o año.

Sin embargo, los datos almacenados por ***sessionStorage*** expiran cuando se cierra el navegador o la pestaña actual.

Cookies vs WebStorage

Las cookies poseen fecha de caducidad, lo cual el localStorage no posee como tal.

Las peticiones realizadas usando web storage no envían la información almacenada, sin embargo en el caso de que la parte de servidor necesite información almacenada en el cliente, habrá que enviársela mediante algún hidden input o alguna otra mecánica. A las cookies les pasa lo contrario, todas las cookies de un dominio se envían en cada petición, realizando un gasto innecesario de recursos del servidor haciendo la pagina inestable en algunos casos.

Las cookies tienen mayor soporte debido a que es lo que se usaba siempre.

El tamaño máximo que se puede almacenar en las cookies es de 4KB, el del web storage es de 5120KB, proporcionando mucho mas tamaño para el almacenado el web storage.

Según he leído, las cookies son mas vulnerables a la inyección SQL, y el web storage es mas vulnerable al XSS.

Las cookies son fácilmente accesibles por ambas partes, tanto FrontEnd como BackEnd, sin embargo el LocalStorage es unicamente de FrontEnd, teniendo que enviar la información a través de algún hidden input o alguna otra mecánica.

SessionStorage vs SessionVariable

Ambas son variables de sesión, es decir, que al cerrar el navegador ambas expiran.

SessionStorage es de la parte FrontEnd y las sessionVariables son de BackEnd.

En mi opinión, ambas tienen su nicho, cuando se desea guardar información que no se desea ver comprometida, lo mejor será guardarla en el BackEnd, en cualquier otro caso se pueden usar el sessionStorage.