Almacenamiento Local y JSON

Introducción

- Recordemos que las variables, y todos los datos que podemos procesar desde JavaScript en un documento estático HTML, desaparecerán de memoria una vez cerrada la página, o el navegador.
- De esta forma funcionan las variables almacenadas en memoria ram de nuestro computador. Una vez finalizada la ejecución de un programa, las mismas dejan de existir.
- Si queremos persistir información, debemos guardarla en una base de datos, o en un archivo.

Almacenamiento Local

- En el desarrollo web con JavaScript, podemos hacer uso de una pequeña base de datos de nuestros navegadores, la cual se llama Local Storage.

- En el Local Storage, podremos persistir información de manera local en el navegador, la cual no se perderá si recargamos o cerramos la página, o incluso el navegador.

LocalStorage y SessionStorage

- **LocalStorage** y **SessionStorage** son propiedades que acceden al objeto Storage
- Sirven para almacenar datos de manera local.
- Con localStorage podemos almacenar la información de forma indefinida o hasta que se decida limpiar los datos del navegador.
- En cambio, sessionStorage almacena información mientras la pestaña donde se esté utilizando siga abierta, una vez cerrada, la información se elimina.

Guardando Datos

- Utilizamos el método setItem()

```
> localStorage.setItem("nombre", "Joaquin");
< undefined
> sessionStorage.setItem("carrera", "Programador");
< undefined
>
```

- El almacenamiento funciona de la forma clave – valor
- Tanto la clave como valor, deben ser del tipo String.

Obteniendo y eliminando datos

- Para obtener datos del localStorage, o del sessionStorage, utilizaremos el método getItem()

```
> sessionStorage.getItem("carrera");
"Programador"
> localStorage.getItem("nombre");
"Joaquin"
>
```

- Para eliminar, usamos el método removeItem()

```
> localStorage.removeItem("nombre");
undefined
> localStorage.getItem("nombre");
null
```

Vaciar el Storage

- Para limpiar todo el localStorage o el sessionStorage, podemos hacerlo con el método clear()

```
> localStorage.clear();
< undefined
> localStorage.getItem("nombre");
< null
> sessionStorage.clear()
< undefined
> sessionStorage.getItem("carrera");
< null
>
```

¿Y si queremos almacenar datos que no sean cadenas?

- Para almacenar arrays, objetos, numeros, o algun dato que no sea una cadena, lo que debemos hacer es convertir dicho elemento a string, o guardar una representación del dato en forma de string.

- Si queremos guardar un Array, o un objeto, debemos convertirlo a cadena.
- Esto lo podemos hacer con algo llamado **JSON**

JSON

- JavaScript Object Notation.
- Es un estándar para el intercambio de información, que usa la sintaxis de objetos de JavaScript.
- Antiguamente se utilizaba algo llamado XML.
- Las claves deben ser strings
- Los valores pueden ser numeros, objetos, cadenas, booleanos, arrays, o nulos (null)

Ejemplo de JSON

```json
[
    {
        "id_empleado": 1,
        "nombre": "Juan Perez",
        "areas_trabajo": [
            {
                "nombre": "Desarrollo Web Frontend",
                "tecnologias": "Angular, HTML/CSS/JS"
            }
        ],
        "recibe_aumento": true
    },
    {
        "id_empleado": 2,
        "nombre": "Maria Núñez",
        "areas_trabajo": [
            {
                "nombre": "Desarrollo Movil iPhone",
                "tecnologias": "Objective-C, Ionic"
            }
        ],
        "recibe_aumento": false
    }
]
```

Algunos usos del JSON

- Son muy usados para intercambiar información entre distintos sistemas, o entre un cliente y un servidor.

- Las bases de datos no relacionales se basan en documentos del tipo JSON

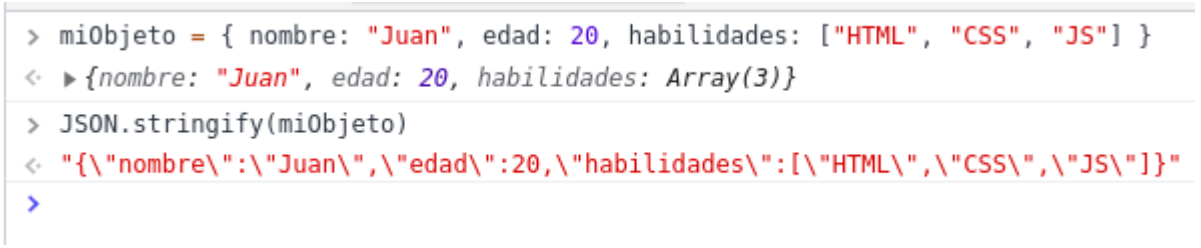
student_id	age	score
1	12	77
2	12	68
3	11	75


```
[
 {
  "student_id":1,
  "age":12,
  "score":77
 },
 {
  "student_id":2,
  "age":12,
  "score":68
 },
 {
  "student_id":3,
  "age":11,
  "score":75
 }
]
```

Convertir un objeto o array a cadena

- Desde JavaScript, podemos hacer úso del objeto JSON, para parsear arrays de objetos y cadenas.

- JSON.stringify(objeto) // nos retorna el objeto en forma de texto



Parsear una cadena a JSON

- Usaremos el método JSON.parse()

```
"{\"nombre\":\"Juan\",\"edad\":20,\"habilidades\":[\"HTML\",\"CSS\",\"JS\"]}"
> let objetoCadena = JSON.stringify(miObjeto)
undefined
> JSON.parse(objetoCadena)
▶ {nombre: "Juan", edad: 20, habilidades: Array(3)}
>
```

Guardando objetos o JSON en Storage

```
> datos = { nombre: "Juan", edad: 20, habilidades: ["HTML", "CSS", "JS"] };
< ▶ {nombre: "Juan", edad: 20, habilidades: Array(3)}
> localStorage.setItem("mis_datos", JSON.stringify(datos))
< undefined
>
```

- Para obtener:

```
> jsonMisDatos = JSON.parse(localStorage.getItem("mis_datos"))
< ▶ {nombre: "Juan", edad: 20, habilidades: Array(3)}
>
```

Accediendo al Storage desde Chrome

- Podemos hacerlo en la pestaña de Application, en herramientas del desarrollador.