TALENTO TECH VALLE **ACTIVIDAD: JSON Y APIS** **103 – 104** **CLASE 11**	

Guía Completa de JSON

¡Claro! JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos que es fácil de leer y escribir para los humanos y fácil de parsear y generar para las máquinas. Es ampliamente utilizado en aplicaciones web para enviar y recibir datos entre un servidor y un cliente.

Guía Completa de JSON

1. ¿Qué es JSON?

JSON es un formato de texto que sigue una sintaxis específica para representar datos estructurados. Se utiliza principalmente para transmitir datos en aplicaciones web.

2. Sintaxis de JSON

La sintaxis de JSON se basa en dos estructuras:

- **Colección de pares nombre/valor**: En varios lenguajes, esto se realiza como un objeto, registro, estructura, diccionario, tabla hash, lista con clave o matriz asociativa.
- **Lista ordenada de valores**: En la mayoría de los lenguajes, esto se realiza como una matriz, vector, lista o secuencia.

Reglas de sintaxis:

- Los datos están en pares nombre/valor.
- Los datos están separados por comas.
- Los objetos están delimitados por llaves `{}`.
- Los arreglos están delimitados por corchetes `[]`.
- Los nombres deben estar entre comillas dobles `"`.
- Los valores pueden ser una cadena entre comillas dobles, un número, un objeto, un arreglo, un valor booleano (`true` o `false`), o `null`.

TALENTO TECH VALLE **ACTIVIDAD: JSON Y APIS** **103 – 104** **CLASE 11**	

3. Ejemplos Básicos de JSON

Objeto JSON:

```
{
  "nombre": "Juan",
  "edad": 30,
  "casado": true,
  "hijos": ["Ana", "Luis"],
  "direccion": {
    "calle": "Calle Falsa 123",
    "ciudad": "Ciudad Ejemplo"
  }
}
```

Arreglo JSON:

```
[
  {
    "nombre": "Juan",
    "edad": 30
  },
  {
    "nombre": "Ana",
    "edad": 25
  }
]
```

4. Métodos JSON en JavaScript

- **`JSON.parse()`**: Convierte una cadena JSON en un objeto JavaScript.
- **`JSON.stringify()`**: Convierte un objeto JavaScript en una cadena JSON.

TALENTO TECH VALLE ACTIVIDAD: JSON Y APIS 103 – 104 CLASE 11	

Ejemplo de `JSON.parse()`:

```
const jsonString = '{"nombre": "Juan", "edad": 30}';
const objeto = JSON.parse(jsonString);
console.log(objeto.nombre); // Output: Juan
```

Ejemplo de `JSON.stringify()`:

```
const objeto = { nombre: "Juan", edad: 30 };
const jsonString = JSON.stringify(objeto);
console.log(jsonString); // Output: {"nombre":"Juan","edad":30}
```

5. ¿Para qué se utiliza JSON?

JSON se utiliza principalmente para:

- **Intercambio de datos**: Entre un servidor y un cliente en aplicaciones web.
- **Configuración**: Archivos de configuración en muchas aplicaciones.
- **Almacenamiento de datos**: En bases de datos NoSQL como MongoDB.

6. ¿Cuándo utilizar JSON?

- **Comunicación Cliente-Servidor**: Cuando necesitas enviar datos desde el cliente al servidor o viceversa.
- **APIs**: Muchas APIs modernas utilizan JSON para enviar y recibir datos.
- **Configuración**: Cuando necesitas un formato legible y fácil de modificar para archivos de configuración.

TALENTO TECH VALLE **ACTIVIDAD: JSON Y APIS** **103 – 104** **CLASE 11**	

Ejemplo:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>json</title>
</head>
<body>
<script>
    let datos =
                {
                "nombre": "juan",
                "apellido": "Giraldo",
                "edad": 26,
                "direccio": [
                    {
                        "Ciduad": "Cali",
                        "Dir": "cra 86 # 26 -65",
                        "barrio": "Valle del lili"
                    }
                ],
                "telefonos": ["3145678321", "3160986542"]

            }


        console.log(datos)

        console.log(datos.nombre, datos.telefonos[0])
        datos.direccio.forEach(direccion => {
            console.log(`ciudad: ${direccion.Ciduad}`)

        });
</script>
</body>
</html>
```

TALENTO TECH VALLE **ACTIVIDAD: JSON Y APIS** **103 – 104** **CLASE 11**	

2. Fetch API

¿Qué es Fetch API?

- La **Fetch API** es una interfaz que permite hacer peticiones HTTP asíncronas, lo que es ideal para interactuar con servidores y obtener o enviar datos.
- Es moderna, flexible y retorna promesas, lo que facilita el manejo de respuestas asincrónicas.

Sintaxis Básica de Fetch API

```javascript
fetch('URL') // Hace la petición
  .then(response => response.json()) // Convierte la respuesta a JSON
  .then(data => console.log(data)) // Muestra los datos recibidos
  .catch(error => console.error('Error:', error)); // Manejo de errores
```

Ejemplo Práctico de Fetch API

Imaginemos que estamos obteniendo datos de una API ficticia que devuelve información sobre usuarios.

```javascript
fetch('https://jsonplaceholder.typicode.com/users')
  .then(response => response.json()) // Convertimos la respuesta a JSON
  .then(users => {
    users.forEach(user => {
      console.log(`Nombre: ${user.name}, Email: ${user.email}`);
    });
  })
  .catch(error => console.error('Error al obtener datos:', error));
```

Explicación del código:

- `fetch('URL')`: Realiza una solicitud GET a la URL proporcionada.
- `response.json()`: Convierte la respuesta en formato JSON.

- `.then()`: Se ejecuta cuando la promesa se resuelve con éxito (respuesta exitosa del servidor).
- `.catch()`: Se ejecuta si ocurre un error en la solicitud.

TALENTO TECH VALLE **ACTIVIDAD: JSON Y APIS** **103 – 104** **CLASE 11**	

Actividades para Resolver

Actividad 1: Visualiza los Datos de una API

Enunciado:

1. Realiza una petición a la API pública de `https://jsonplaceholder.typicode.com/posts` para obtener una lista de publicaciones.
2. Muestra el título de cada publicación en la página HTML en un formato ordenado.
3. Maneja posibles errores en la solicitud y muestra un mensaje adecuado si algo falla.

Pistas:

- Usa el método `response.json()` para obtener los datos.
- Crea un contenedor en el HTML para mostrar los títulos de las publicaciones.

Actividad 2: Filtrado de Datos

Enunciado:

1. Obtén los datos de la API `https://jsonplaceholder.typicode.com/users`.
2. Filtra los usuarios cuyo nombre contenga la palabra “Leanne”.
3. Muestra en la página un mensaje indicando cuántos usuarios coinciden con el filtro y muestra sus nombres y correos electrónicos.

Pistas:

- Usa `Array.prototype.filter()` para filtrar los usuarios.
- Asegúrate de que los resultados se muestren solo si se encuentran coincidencias.

Actividad 3: Manipulando Datos y Enviando Datos

Enunciado:

1. Crea un formulario HTML que permita ingresar un nuevo usuario con su nombre y correo electrónico.

2. Cuando el formulario se envíe, usa la Fetch API para enviar estos datos a la API de `https://jsonplaceholder.typicode.com/users` (aunque la API no guardará los datos, simula el envío con una respuesta exitosa).
3. Muestra un mensaje de éxito en la página si los datos se enviaron correctamente.

Pistas:

- Usa el método `POST` en la solicitud de Fetch.
- Los datos deben enviarse en formato JSON, usando el método `JSON.stringify()`.