0️⃣

Introducción a JSON

1. ¿Qué es JSON?

- **Significado:** JSON significa ***J**ava**S**cript **O**bject **N**otation*.

▼ **Uso:**

- Popular para la comunicación entre navegadores web y servidores. A pesar de derivarse de JavaScript, es independiente del lenguaje.
- En **Sistemas de gestión de bases de datos NoSQL como MongoDb y CouchDb**.
- En aplicaciones de sitios web de **redes sociales como Twitter, Facebook, LinkedIn y Flickr**

    - Incluso con la **API de Google Maps.**

- **Objetivo:** Estructura datos en formato de pares clave/valor, lo que permite su transmisión de manera sencilla.
- **Aplicaciones:** Usado en AJAX para intercambiar datos y en sistemas NoSQL como MongoDB o CouchDB.

2. Características

- **Independiente del lenguaje:** JSON puede ser generado y leído en cualquier lenguaje de programación.
- **Legible por humanos:** Su sintaxis simple facilita la lectura y escritura por desarrolladores.
- **Ligero:** Más simple que formatos como XML, lo que lo hace más rápido en la transmisión y procesamiento de datos.

3. Reglas Sintácticas

- **Estructura:** Los datos están organizados en pares clave/valor, separados por comas.
- **Llaves ({})** contienen objetos, mientras que **corchetes ([])** encierran arrays.
- **Tipos de datos admitidos:** cadena, número, booleano, array, objeto y null.

Ejemplo:

```
{
  "Libros": [
    {
      "Nombre": "Árboles",
      "Curso": "Introducción a los árboles",
      "Contenido": ["Árbol Binario", "BST", "Árbol Genérico"]
    },
    {
      "Nombre": "Grafos",
      "Temas": ["BFS", "DFS", "Orden Topológico"]
```

```
    }
  ]
}
```

4. Ventajas de JSON

- **Ligereza:** Su sintaxis es simple, lo que permite respuestas rápidas.
- **Amplia compatibilidad:** Funciona bien en todos los navegadores y sistemas operativos.
- **Análisis eficiente en el servidor:** En comparación con otros formatos, JSON permite un análisis rápido en el servidor.

5. Desventajas de JSON

- **Gestión de errores limitada:** Un pequeño error en el formato puede romper toda la estructura.
- **Seguridad:** Puede ser vulnerable a ataques si no se maneja adecuadamente, especialmente en navegadores no autorizados.
- **Herramientas de desarrollo:** El soporte de herramientas puede ser limitado en algunos casos.

6. Tipos de datos JSON

- `String` **(Cadena):** Valores entre comillas dobles.
- `Number` **(Número):** Decimales en base 10, sin soporte para NaN o notaciones especiales.
- `Boolean` **(Booleano):** Valores `true` o `false`.
- `Null` **(Nulo):** Representa valores nulos.
- `Object` **(Objeto):** Colección de pares clave/valor entre llaves `{}`.
- `Array` **(Matriz):** Lista ordenada de valores, entre corchetes `[]`.
- `Null` **(Nulo):** Un valor vacío, utilizando la palabra clave `null`.

Ejemplo:

```
{
  "Poeta": {
    "nombre": "Sylvia Plath",
    "edad": 32,
    "géneroLiterario": "Poesía"
  }
}
```

7. Ejemplo completo

```
{
  "Poetas": [
    {
      "nombrePoeta": "Sylvia Plath",
      "obraDestacada": "Ariel",
      "géneroLiterario": "Poesía"
    },
    {
      "nombrePoeta": "Emily Dickinson",
      "obraDestacada": "The Collected Poems",
      "géneroLiterario": "Poesía"
    },
    {
      "nombrePoeta": "Walt Whitman",
      "obraDestacada": "Leaves of Grass",
      "géneroLiterario": "Poesía"
    }
  ]
}
```

📖 JSON es la opción preferida para muchos desarrolladores debido a su simplicidad, velocidad y facilidad de integración en diversas aplicaciones y lenguajes de programación. Aunque tiene algunas limitaciones en seguridad y manejo de errores, sigue siendo una herramienta esencial para el intercambio de datos.