

## 1. Portada

# JSON

JavaScript Object Notation

(Notación de Objetos de JavaScript).

## 2. Ejemplo

Curriculum.json

```
{
  "nombre": "María García",
  "edad": 28,
  "hobbies": ["lectura", "senderismo", "fotografía"],
  "casada": false,
  "educacion": {
    "grado": "Ingeniería Informática",
    "universidad": "Universidad Politécnica de Cataluña",
    "añoGraduacion": 2018
  },
  "idiomas": [
    {
      "nombre": "Español",
      "nivel": "nativo"
    },
    {
      "nombre": "Inglés",
      "nivel": "avanzado"
    }
  ]
}
```

## 3. Qué es JSON

JSON es un formato de archivo utilizado para:

### a) Almacenar información:

Ejemplos:

- Archivo de configuración.
- Información de contactos de una aplicación.

### b) Transmitir información

Ejemplos:

- Enviar localidades de una provincia.
- Compartir datos de alumnos entre dos colegios.

## 4. Sintaxis de JSON.

JSON tiene una sintaxis muy sencilla y por tanto eficiente.

- Observa que no contiene Prólogo o instrucciones de procesamiento.

- **OBJETO:**

- En JSON, todo el contenido está dentro de algo llamado objeto, que define entre llaves { }. Por eso un archivo JSON comienza por una llave abierta y termina con una llave cerrada.

- **CLAVE-VALOR**

- La información del interior del objeto se organiza como propiedades separadas por comas
  - Ejemplo de propiedad: **"edad": 28,**
- Cada Propiedad consta de un par Clave-Valor separados por dos puntos:
  - Clave: **"edad"** Es el nombre de la propiedad
  - Valor: **28** Es el nombre de la propiedad

- **VALOR:**

- **Cadenas de texto:** están entre comillas dobles (ejemplo: **"María García"**).
- **Números:** no llevan comillas (ejemplo: **28**).
- **Booleanos:** se escriben como true o false, sin comillas (ejemplo: **false**).
- **Arrays (listas):** son una secuencia de valores entre corchetes [ ].

Ejemplo: **["lectura", "senderismo", "fotografía"],**

- **Objetos:** son valores compuestos por otros pares clave-valor, encerrados entre { }. Ejemplo:

```
{  
  "grado": "Ingeniería Informática",  
  "universidad": "Universidad Politécnica de Cataluña",  
  "añoGraduacion": 2018  
}
```

- **Arrays de Objetos:** Puedes tener arrays que contengan objetos. Cada objeto dentro del array puede tener sus propios pares clave-valor:

```
[  
  {  
    "nombre": "Español",  
    "nivel": "nativo"  
  },  
  {  
    "nombre": "Inglés",  
    "nivel": "avanzado"  
  }  
]
```

## 5. JSON NO es un Lenguaje de Marcas

- **Propósito diferente:** JSON está diseñado específicamente para el intercambio y almacenamiento de datos estructurados, mientras que los lenguajes de marcas como XML están diseñados también para crear documentos de texto con formato con significados semánticos.
- **Ausencia de etiquetas:** Los lenguajes de marcas típicamente utilizan etiquetas para delimitar y describir elementos, mientras que JSON utiliza una estructura de pares clave-valor y arrays.
- **Estructura más simple:** JSON tiene una estructura más simple basada en objetos y arrays, mientras que los lenguajes de marcas suelen tener una estructura más compleja con elementos anidados, atributos y espacios de nombres.
- **Ausencia de metadatos:** JSON no tiene capacidad inherente para incluir metadatos o información adicional sobre la estructura de los datos, algo que sí es común en lenguajes de marcas (XML: prólogo, atributos, las propias etiquetas...)
- **Limitaciones en tipos de datos:** JSON soporta un conjunto limitado de tipos de datos primitivos, mientras que los lenguajes de marcas pueden ser más flexibles en este aspecto porque de hecho no especifican tipos de datos.

## 6. Qué programas se utilizan para ver y editar JSON.

JSON es un tipo de archivo de texto plano. Es decir, que utiliza caracteres que pueden ser leídos por los humanos e imprimidos:

- Letras: a, B, c, ... Z
- Dígitos: 0, 1, 2... 9
- Caracteres comunes en la escritura: - ( { \* \_ / \$ i ¿...

Por ese motivo, para crear o editar un archivo JSON podemos emplear algo tan simple como el block de notas.

Sin embargo, el block de notas es poco útil. Hay muchas aplicaciones que no permiten trabajar con más facilidad y confort con archivos JSON, ya que ofrecen múltiples funciones muy útiles como:

- **Visualización mejorada**
  - Resaltado de sintaxis: Colorea diferentes elementos del JSON (claves, valores, corchetes, etc.) para mejorar la legibilidad
  - Formateo automático: Organiza y sangra el código automáticamente para una mejor estructura visual
  - Vista de árbol: Permite visualizar la estructura jerárquica del JSON de forma más clara
- **Funcionalidades avanzadas**
  - Validación en tiempo real: Detecta y señala errores de sintaxis mientras se escribe
  - Autocompletado: Sugiere claves y valores basados en el contexto
  - Plegado de código: Permite colapsar y expandir secciones del JSON para facilitar la navegación
- **Herramientas de edición**
  - Búsqueda y reemplazo avanzados: Ofrece opciones más potentes para encontrar y modificar datos
  - Edición de múltiples líneas: Permite realizar cambios en varias líneas simultáneamente
  - Conversión de formatos: Facilita la conversión entre JSON y otros formatos como XML o YAML
- **Integración y extensibilidad**
  - Integración con control de versiones: Se integra con sistemas como Git para un mejor seguimiento de cambios
  - Extensiones y plugins: Permite añadir funcionalidades adicionales según las necesidades específicas
- **Rendimiento**
  - Manejo de archivos grandes: Procesa archivos JSON de gran tamaño de manera más eficiente que el bloc de notas

Cuando queremos indicar a un navegador u otro software de qué tipo es un archivo .json, se emplea "application/json".

## 6. EL ORIGEN DE JSON -> JavaScript.

De un modo simplificado, un objeto en Javascript es como una agrupación de datos.

Estos objetos se crean en memoria así, en JS:

```
// 1. Crear el objeto 'persona'  
const persona = {  
  nombre: "María García",  
  edad: 28,  
  hobbies: ["lectura", "senderismo", "fotografía"],  
  casada: false  
};
```

Después se puede acceder a sus miembros así:

```
// 2. Modificar la edad a 29  
persona.edad = 29;  
  
// 3. Mostrar la nueva edad  
console.log(persona.edad); // Imprimirá: 29
```

Si quisiéramos enviar ese objeto a otro ordenador, o guardarlo en un archivo, antes habría que convertirlo en una cadena JSON. A eso se le denomina serialización:

```
// 4. Convertir el objeto a una cadena JSON  
const personaJSON = JSON.stringify(persona, null, 2); // 'null, 2' es para formatear con sangría
```

Si quisiéramos guardar esa cadena JSON en un archivo:

```
// 5. Guardar la cadena JSON en un archivo llamado 'persona.json'  
fs.writeFile('persona.json', personaJSON);
```

Aunque se creó para JSON, actualmente es utilizado por muchos otros lenguajes y estándares para almacenar y transmitir información.

