

JSON y XML

¿Qué es un XML?

El [Extensible Markup Language](#) es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos en un **formato que es tanto legible por humanos como por máquinas.**

Se utiliza para **estructurar, almacenar y transportar información de manera consistente y estandarizada.**

Ejemplo:

```
<libro>
  <titulo>XML Práctico</titulo>
  <autor>Jorge López</autor>
  <precio>129.99</precio>
</libro>
```



Fortalezas y debilidades del XML

Ventajas

- Tiene un formato **estructurado** que facilita la organización de los datos.
- Puede ser **validado fácilmente** mediante Schemas(XSD).
- Se pueden definir **estructuras complejas y reutilizables**.



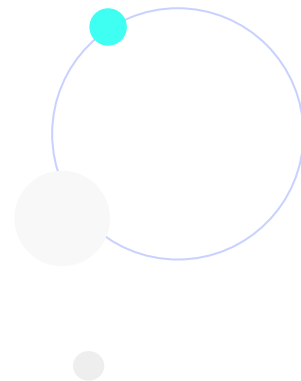
Desventajas

- Aunque es estructurado, puede ser más **complicado de entender** debido a su sintaxis y la **necesidad de manejar elementos adicionales** como los namespaces.
- Su formato estricto puede **dificultar la escritura y modificación** de los documentos.
- Lleva **más tiempo** procesarlo.
- Un error con los namespaces puede hacer que **todo el documento sea inválido**.



Ejemplo:

```
<persona>
  <nombre>Juan Pérez</nombre>
  <edad>30</edad>
  <direccion>
    <calle>Calle Falsa 123</calle>
    <ciudad>Madrid</ciudad>
    <codigoPostal>28013</codigoPostal>
  </direccion>
  <telefono>+34 600 123 456</telefono>
  <email>juan.perez@example.com</email>
</persona>
```



¿Qué es JSON?

JavaScript Object Notation (notación de objeto de JavaScript) es un **formato de texto sencillo para el intercambio de datos**.

Se trata de un subconjunto de la notación literal de objetos de JavaScript, aunque, debido a su **amplia adopción como alternativa a XML**, se considera un formato independiente del lenguaje.



Los tipos de datos disponibles con JSON son:

Números	Se permiten números negativos y, opcionalmente, pueden contener parte fraccional separada por puntos. Ejemplo: 123.456
Cadenas	Representan secuencias de cero o más caracteres . Se ubican entre dobles comillas y se permiten cadenas de escape. Ejemplo: "Hola"
Booleanos	Representan valores <i>booleanos</i> y pueden tener dos valores: true y false . null : Representa el valor nulo .
Array	Representa una lista ordenada de cero o más valores que pueden ser de cualquier tipo. Los valores se separan por comas y el vector se escribe entre corchetes . Ejemplo: ["juan", "pedro", "jacinto"]
Objetos	Son colecciones no ordenadas de pares de la forma <nombre>: <valor> separados por comas y puestas entre llaves. El nombre tiene que ser una cadena y entre ellas. El valor puede ser de cualquier tipo. Ejemplo: {"departamento":8,"nombredepto":"Ventas","director": "juan rodriguez", "empleados":[{"nombre":"Pedro", "apellido":"Fernandez"}, {"nombre":"Jacinto", "apellido":"Benavente"}]}

Fortalezas y debilidades de JSON

Ventajas

- **Es sumamente simple:** JSON usa una estructura basada en pares clave-valor. Este formato facilita su comprensión y edición.
- **Velocidad de procesamiento alta:** Su formato ligero y estructura optimizada permiten una rápida lectura y escritura. La mayoría de las bibliotecas están diseñadas para procesar JSON de manera eficiente.
- **Archivos de menor tamaño:** JSON produce archivos más compactos en comparación con XML. Esto es ideal para aplicaciones web y móviles.



Desventajas

- **Estructura enredosa:** JSON puede volverse complejo cuando los datos están profundamente anidados. Esto hace que sea difícil de interpretar a simple vista sin herramientas adecuadas.
- **Falta de tipos de datos avanzados:** JSON no soporta tipos de datos complejos como fechas o números precisos. Esta limitación puede requerir conversiones adicionales.



Ejemplo de JSON

```
{  
  "nombre": "Juan Pérez",  
  "edad": 30,  
  "direccion": {  
    "calle": "Calle Falsa 123",  
    "ciudad": "Madrid",  
    "codigoPostal": 28013  
  },  
  "telefono": "+34 600 123 456",  
  "email": "juan.perez@example.com"  
}
```



Diferencias entre XML y JSON

	XML	JSON
Legibilidad	Tiende a ser más verboso debido a las etiquetas de apertura y cierre.	Suele ser más compacto y legible , especialmente para estructuras de datos más simples.
Sintaxis	Utiliza etiquetas de apertura y cierre (<code><etiqueta>...</etiqueta></code>), lo que le da una estructura jerárquica y anidada .	Utiliza pares clave-valor en un formato de texto con llaves <code>{}</code> para definir objetos y corchetes <code>[]</code> para definir <i>arrays</i> .

(Continúa en el siguiente *slide*)

Continuación:

	XML	JSON
Interoperabilidad y uso	Es ampliamente utilizado en aplicaciones que requieren validación estricta de datos y documentos (por ejemplo, en servicios web SOAP, configuración de aplicaciones, y otros).	Es preferido en aplicaciones web modernas , especialmente en servicios web <i>RESTful</i> , debido a su simplicidad y eficiencia .
Uso de atributos	<p>Puede usar atributos dentro de las etiquetas.</p> <p>Por ejemplo:</p> <pre><libro titulo="Cien años de soledad" autor="Gabriel García Márquez" año="1967" /></pre> <p>En este caso, titulo, autor, y año son atributos dentro de la etiqueta <libro>.</p>	<p>No utiliza atributos, todo se representa como pares clave-valor.</p> <p>Por ejemplo:</p> <pre><libro> <titulo>Cien años de soledad</titulo> <autor>Gabriel García Márquez</autor> <año>1967</año> </libro></pre>