

# Lenguaje de marcas XML

## ¿Qué es XML?

XML (Extensible Markup Language) es un lenguaje de marcas que se utiliza para estructurar y almacenar datos de manera legible tanto por humanos como por máquinas. Es un lenguaje de marcas versátil que se utiliza para estructurar y representar datos de manera jerárquica y legible por humanos. Su extensibilidad y capacidad para describir datos complejos lo convierten en una herramienta valiosa en una variedad de campos y aplicaciones. Aquí están las características clave de XML:

1. Legibilidad: XML utiliza una sintaxis basada en etiquetas que es fácilmente comprensible para los humanos. Las etiquetas, atributos y contenido están diseñados para ser intuitivos y descriptivos.
2. Extensibilidad: Como su nombre indica, XML es "extensible", lo que significa que permite a los usuarios definir sus propias etiquetas y estructuras de datos según sus necesidades. Esto hace que XML sea adecuado para una amplia variedad de aplicaciones.
3. Jerarquía y anidamiento: Los datos en XML se organizan en una jerarquía de elementos que pueden estar anidados dentro de otros elementos. Esto permite representar relaciones complejas entre datos.
4. Uso de etiquetas: Los elementos en XML se definen utilizando etiquetas que constan de una etiqueta de apertura (**<nombre>**) y una etiqueta de cierre (**</nombre>**). El contenido del elemento se coloca entre estas etiquetas.
5. Atributos: Además de las etiquetas, los elementos pueden contener atributos que proporcionan información adicional sobre el elemento. Los atributos se definen dentro de la etiqueta de apertura.
6. Documentos bien formados: Un documento XML debe seguir ciertas reglas para considerarse "bien formado". Estas reglas incluyen tener un elemento raíz que envuelve todos los demás elementos, etiquetas debidamente anidadas y atributos en formato válido.
7. Documentos válidos (opcional): Además de ser bien formado, un documento XML también puede ser "válido" si cumple con una estructura y un conjunto de reglas definidas en un esquema o DTD (Definición de Tipo de Documento).
8. Independiente de la plataforma: XML es independiente de la plataforma, lo que significa que los documentos XML pueden ser creados y procesados en diferentes sistemas y plataformas de manera consistente.
9. Ampliamente utilizado: XML se ha convertido en un estándar de facto para representar y transferir datos estructurados en una amplia variedad de aplicaciones, como la web, la comunicación entre sistemas, el intercambio de datos, la configuración de aplicaciones y más.

## Sintaxis básica:

XML se define por una serie de reglas que permiten a los usuarios crear sus propias etiquetas y definir la estructura de sus documentos de datos.

- XML utiliza etiquetas (tags) para marcar elementos y atributos.
- Un elemento XML se define con una etiqueta de apertura (<nombre>), seguida de su contenido y una etiqueta de cierre (</nombre>). Los elementos pueden ser anidados.
- Los atributos se definen dentro de la etiqueta de apertura y se separan de la etiqueta con espacios en blanco.
- Un documento XML debe tener un elemento raíz que englobe a todos los demás elementos.
- XML no define tipos de datos nativos, por lo que puede contener cualquier tipo de datos, incluyendo texto, números, fechas, etc. Ejemplos de uso:

- Texto:

```
xml Copy code  
  
<nombre>Juan Pérez</nombre>
```

- Números:

```
xml Copy code  
  
<edad>30</edad>
```

- Fechas:

```
xml Copy code  
  
<fechaNacimiento>1993-05-15</fechaNacimiento>
```

- Atributos:

```
xml Copy code  
  
<persona id="1" genero="masculino">Juan Pérez</persona>
```

- Elementos anidados:

```
xml Copy code  
  
<alumno>  
  <nombre>Maria</nombre>  
  <apellido>González</apellido>  
</alumno>
```

- Lista de valores (usando elementos repetidos):

```
xml Copy code

<números>
  <valor>1</valor>
  <valor>2</valor>
  <valor>3</valor>
</números>
```

- Anidamiento de elementos y atributos:

```
xml Copy code

<libro id="1">
  <titulo>La Odisea</titulo>
  <autor>Homer</autor>
</libro>
```

- Comentarios:

- XML permite comentarios dentro del documento, los cuales se incluyen entre <!-- y -->.

```
xml Copy code

<!-- Esto es un comentario -->
```

- CDATA:

- XML permite incluir contenido sin procesar (por ejemplo, código HTML o XML) utilizando la sección <![CDATA[ y ]]>.

```
xml Copy code

<![CDATA[<html><p>Ejemplo de HTML dentro de CDATA</p></html>]]>
```

- Namespace:

- XML admite espacios de nombres (namespaces) para evitar conflictos de nombre de elementos y atributos cuando se integran múltiples vocabularios XML en un mismo documento.

XML se utiliza en una variedad de aplicaciones, como intercambio de datos, configuración de aplicaciones, representación de documentos estructurados y en la definición de lenguajes de marcado específicos. Su capacidad de anidar

elementos y atributos permite representar datos de manera muy flexible y estructurada.

# Lenguaje de marcas JSON

## ¿Qué es JSON?

JSON (JavaScript object notation -notación de objetos de JavaScript), Es un formato de texto totalmente independiente del lenguaje de programación, que se utiliza para el intercambio de datos. No es un lenguaje de programación sino un archivo que contiene datos estructurados, que se utiliza para transferir información entre sistemas.

Es ampliamente utilizado en la comunicación entre sistemas web, en el almacenamiento de datos estructurados y en la configuración de aplicaciones debido a su simplicidad y facilidad de lectura y escritura tanto para humanos como para máquinas. La capacidad de anidar objetos y arrays permite representar datos complejos de manera estructurada.

## ¿Cómo funciona?

Su funcionamiento se basa en la estructuración de una colección de pares con nombre y valor que contienen:

Una clave: que corresponde al identificador del contenido.

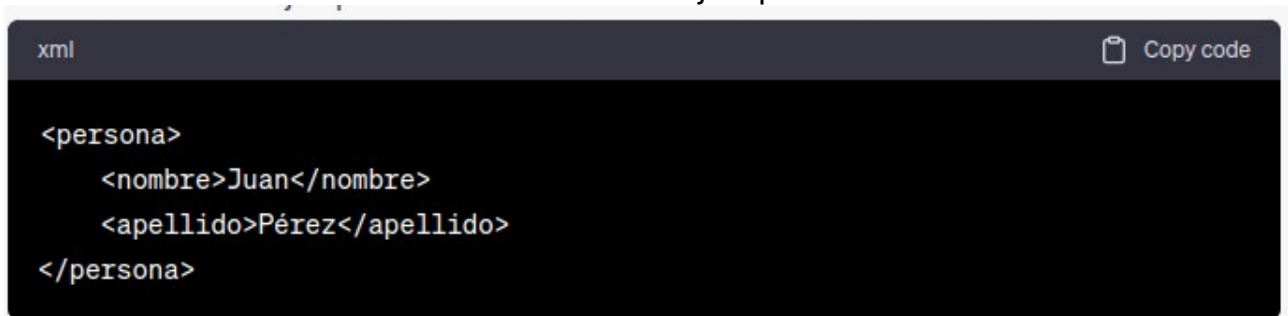
Un valor: que representa el contenido correspondiente.

## ¿Cuáles son sus diferencias con XML?

XML (Extensible Markup Language) y JSON (JavaScript Object Notation) son dos formatos de lenguaje de marcas utilizados para representar datos de manera estructurada, pero tienen diferencias significativas en cuanto a su sintaxis y uso. Aquí hay algunas de las diferencias clave entre XML y JSON:

### 1. Sintaxis:

- XML utiliza etiquetas (tags) para marcar los elementos y atributos, y estos deben estar anidados correctamente. Por ejemplo:

A screenshot of a code editor with a dark background. The title bar at the top left says 'xml' and the top right has a 'Copy code' button. The code is written in a light color and shows an XML structure for a person.

```
<persona>
  <nombre>Juan</nombre>
  <apellido>Pérez</apellido>
</persona>
```

- JSON utiliza pares clave-valor y se basa en una estructura de objetos y arrays. Por ejemplo:

```
json Copy code

{
  "nombre": "Juan",
  "apellido": "Pérez"
}
```

2. Legibilidad: tiende a ser más legible para los humanos debido a su sintaxis simplificada y su estructura concisa aunque puede resultar más difícil de leer debido a las etiquetas de apertura y cierre y la necesidad de anidamiento adecuado.

3. Tamaño del archivo: generalmente resulta en archivos más pequeños en comparación con XML para la misma cantidad de datos debido a su sintaxis más compacta.

4. Tipos de datos: XML no define tipos de datos nativos, por lo que se deben utilizar atributos o convenciones para representar tipos de datos como fechas o números mientras que JSON admite tipos de datos nativos como números, cadenas, booleanos, arrays y objetos, lo que hace que sea más natural representar datos estructurados.

5. Uso: XML se ha utilizado ampliamente en la industria durante mucho tiempo, especialmente en aplicaciones como la representación de documentos (por ejemplo, HTML) y en tecnologías como SOAP (Simple Object Access Protocol) para la comunicación web y JSON se ha vuelto más popular en aplicaciones web modernas y en la transferencia de datos entre aplicaciones y servicios web, en parte debido a su simplicidad y facilidad de uso en JavaScript.

6. Validación: XML tiene una especificación de esquema XML (XML Schema) que permite definir la estructura y los tipos de datos permitidos en un documento XML, lo que facilita la validación, JSON no tiene un mecanismo de validación incorporado, pero se pueden utilizar herramientas de validación externas o esquemas JSON (como JSON Schema) para lograrlo.

La elección entre ellos depende de los requisitos específicos de la aplicación y las preferencias del desarrollador.

## Sintaxis de JSON

La sintaxis de JSON se resume en:

- JSON utiliza pares clave-valor, donde las claves (keys) son cadenas de texto y los valores (values) pueden ser de diferentes tipos de datos.
- Los pares clave-valor se separan por dos puntos (:), y los elementos se separan por comas (,).
- Los objetos JSON se encierran entre llaves {} y los arrays se encierran entre corchetes [].

Los tipos de valores que podemos encontrar acompañando al nombre de un dato son:

- Números (enteros y flotantes).
- Cadenas de texto (entre comillas simples o dobles).
- Booleanos (true o false).
- Nulos (null).
- Objetos JSON (pares clave-valor encerrados en llaves { }).
- Arrays JSON (elementos encerrados en corchetes [ ]).

Ejemplos de uso:

- Número:

```
json Copy code  
  
{  
  "edad": 30  
}
```

- Cadena de texto:

```
json Copy code  
  
{  
  "nombre": "Juan Pérez"  
}
```

- Booleano:

```
json Copy code  
  
{  
  "esEstudiante": true  
}
```

- Nulo:

```
json Copy code  
  
{  
  "comentario": null  
}
```

- Objeto JSON Los objetos son conjuntos de datos separados por comas que se identifican por ir entre llaves { }(anidamiento):

```
json Copy code  
  
{  
  "persona": {  
    "nombre": "María",  
    "apellido": "González"  
  }  
}
```

- Array JSON Un array es un conjunto de objetos, también separados por comas y que se identifica por ir entre corchetes [ ](lista de valores):

```
json Copy code  
  
{  
  "números": [1, 2, 3, 4, 5]  
}
```

- Anidamiento de objetos y arrays:

```
json Copy code  
  
{  
  "alumno": {  
    "nombre": "Juan",  
    "notas": [85, 92, 78]  
  }  
}
```

JSON no admite comentarios directamente en su sintaxis. Los comentarios no son parte de la especificación JSON y se deben omitir en los archivos JSON válidos.

Es importante separar los elementos con comas (,) en objetos JSON y en arrays JSON, excepto para el último elemento en la lista.

Las cadenas de texto siempre deben ir entre comillas dobles ("), y las claves también deben estar entre comillas dobles en un objeto JSON.



## **Software**

JSON Editor Online (<http://jsoneditoronline.org/>)

Herramienta online que nos permite crear de manera gráfica un archivo JSON mediante el uso de menús contextuales, así como de visualizarlo. También se encuentra como extensión de Chrome. Podéis leer más sobre él en este artículo.

Online JSON Viewer (<http://jsonviewer.stack.hu/>)

Herramienta online que ayuda a visualizar mejor un JSON, tanto en formato tabulado como en formato libre de espacios en blanco para minimizar su tamaño.