MEMORIA ACADÉMICA XML Y JSON

Valentín Cortés Amadín

Contenidos

Historia.....3
Normas de uso.....4
No está permitido omitir la etiqueta de cierre al crear la sintaxis XML.....4
Las etiquetas XML son sensibles a mayúsculas y minúsculas.....5
Todos los elementos XML deben estar debidamente anidados.....5
Todos los documentos XML deben tener un elemento raíz.....6
Los valores de atributo siempre deben estar entre comillas.....6
Aplicaciones de XML en el ámbito empresarial.....8
El análisis es esencial.....9
Sistemas de información Heterogéneos.....10
Gestión de catálogos en Internet.....11
Dispositivos móviles.....11
Servicios Web.....12
Historia.....13
Normas de uso.....14
Reglas de sintaxis JSON:.....16
Aplicaciones de JSON en el ámbito empresarial.....17

Introducción a XML

Historia de XML

El lenguaje XML (eXtensible Markup Language) tiene sus orígenes en el año 1996, cuando el Word Wide Web Consortium (W3C) comienza a trabajar en un nuevo lenguaje de marcas optimizado para Internet que combinase la simplicidad de HTML con la capacidad de expresión que poseía SGML (Standard Generalized Markup Language), el lenguaje predecesor creado por IBM a comienzos de los años 70.

El 1 de febrero de 1998 nace la primera versión de XML, bautizada como XML 1.0, con el objetivo de poder compartir información entre dos ordenadores mediante un esquema correctamente definido y cerrado que diese como resultado documentos coherentes y legibles que no generen dudas de su estructura.

Más adelante, la W3C intenta aplicar su estándar XML a la definición de HTML, dando como resultado el nacimiento de XHTML 1.0 (eXtensible HyperText Markup Language). Esta primera versión recogía la definición de un lenguaje HTML correctamente estructurado, pero se enfrentaba a un problema: la web estaba inundada de documentos HTML que carecían de toda estructura, y para los navegadores de la época era prácticamente imposible aplicar XHTML 1.0.

Finalmente W3C desiste de implantar XHTML 1.0 y se genera una escisión de esta que acaba definiendo lo que conocemos hoy en día como el estándar HTML5.

Normas de uso

Existen unas reglas de sintaxis establecidas para el lenguaje XML:

- Todos los elementos XML deben tener una etiqueta de cierre.
- Las etiquetas XML son sensibles a mayúsculas y minúsculas.
- Todos los elementos XML deben estar debidamente anidados.
- Todos los documentos XML deben tener un elemento raíz.
- Los valores de atributo siempre deben estar entre comillas.
- Todos los elementos XML deben tener una etiqueta de cierre.

No está permitido omitir la etiqueta de cierre al crear la sintaxis XML

Los elementos XML deben tener una etiqueta de cierre. A continuación se muestran unos ejemplos de uso:

Incorrecto:

```
<body>See Spot run.
<body>See Spot catch the ball.
```

Correcto:

```
<body>See Spot run.</body>
<body>See Spot catch the ball.</body>
```

Las etiquetas XML son sensibles a mayúsculas y minúsculas

Cuando crea documentos XML, la etiqueta < Body > es diferente de la etiqueta <body>.

Incorrecto:

```
<Body>See Spot run.</body>
```

Correcto:

```
<body>See Spot run.</body>
```

Todos los elementos XML deben estar debidamente anidados

Un anidamiento incorrecto de las etiquetas no tiene sentido para XML.
Incorrecto:

```
<b><i>This text is bold and italic.</b></i>
```

Correcto:

```
<b><i>This text is bold and italic.</i></b>
```

Todos los documentos XML deben tener un elemento raíz

Todos los documentos XML deben contener un par de etiquetas individuales para definir un elemento raíz. Todos los demás elementos deben estar dentro de este elemento raíz. Todos los elementos pueden tener subelementos (elementos hijo). Los subelementos deben estar anidados correctamente dentro de su elemento padre.

Ejemplo:

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

Los valores de atributo siempre deben estar entre comillas

No está permitido omitir las comillas alrededor de los valores de atributos. Los elementos XML pueden tener atributos en pares nombre/valor; sin embargo, el valor de atributo siempre debe estar entre comillas.

Incorrecto:

```
<?xml version= “1.0” encoding=“ISO-8859-1”?>
<note date=05/05/05>
<to>Dick</to>
<from>Jane</from>
</note>
```

Correcto:

```
<?xml version= “1.0” encoding=“ISO-8859-1”?>
<note date=”05/05/05”>
<to>Dick</to>
<from>Jane</from>
</note>
```

En el documento incorrecto, el atributo de fecha del elemento note no está entre comillas.

Aplicaciones de XML en el ámbito empresarial

Ya analizamos al comienzo de la memoria que XML es un lenguaje de marcas que permite definir de un modo muy sencillo la estructura jerárquica a la que pertenece un dato, así como HTML permite definir la forma en que se muestra un dato en nuestro navegador.

Hasta este punto, el principal inconveniente al integrar aplicaciones desarrolladas en diferentes lenguajes o plataformas radicaba en que cada método de transmisión de datos estaba ligado a la aplicación que lo generaba, lo que dificultaba en gran medida la comunicación fluida entre ellas. Este desafío persistía incluso al intentar integrar aplicaciones desarrolladas en un mismo lenguaje en una misma plataforma. En el mejor escenario posible, una vez establecida la comunicación de datos, surgía una dificultad real al intentar organizarlos jerárquicamente según el modelo original.

XML facilita de manera muy simple la estructuración de la información, lo que garantiza que el receptor comprenda fácilmente las relaciones entre los datos dentro de la misma estructura enviada. Además, XML permite especificar el tipo de datos recibidos mediante XML Schema, establecer su presentación visual a través de XSL e incluso definir cómo deben ser devueltos mediante SOAP.

XML tiene diversas aplicaciones. Inicialmente utilizado para la transmisión de datos, se complementa con XML Schema para definir el tipo de datos transmitidos, así como para especificar si se permiten valores nulos, repetidos, decimales o si existe una integridad referencial con otros datos en el mismo documento transmitido.

XML es la base de SOAP, un protocolo estándar que facilita el envío bidireccional de paquetes de información para la integración de aplicaciones remotas. Esto permite la transmisión de datos por referencia e incluso en una transacción.

Con XML y XSL, es posible modelar la información visualmente para su presentación, lo que resulta en presentaciones dinámicas, especialmente orientadas al consumidor final (B2C).

XML ofrece una manera sencilla, legible y comprensible de parametrizar aplicaciones, accesible tanto para sistemas informáticos como para personas.

Prácticamente el 99% de las aplicaciones de escritorio modernas admiten la lectura, escritura, importación y exportación en formato XML para persistir la información de manera coherente. Con cada nueva versión, XML se integra más en el back-office de sistemas de escritorio, gestión, web, etc.

Toda esta funcionalidad se logra mediante un modelo descriptivo en formato de texto, basado en estándares de la industria definidos por el W3C (Consorcio World Wide Web). Estos estándares garantizan que la información pueda ser transmitida por Internet sin obstáculos, como los firewalls, y que su interpretación sea universal, independientemente de las plataformas o lenguajes de desarrollo utilizados.

En la actualidad, una solución tecnológica no debería limitar la integración o comunicación con nuevas aplicaciones, módulos, funcionalidades o dispositivos. Por lo tanto, es fundamental considerar la transmisión e integración de información utilizando XML al establecer los alcances de una solución.

Por todas estas razones, una solución tecnológica no puede considerarse completa sin un análisis adecuado de la utilización de la infraestructura XML.

El análisis es esencial

Como con cualquier otra herramienta, arquitectura, metodología, etc., relacionada con proyectos, ya sean informáticos u de otra índole, el primer paso crucial es identificar una necesidad en un momento dado y luego analizarla en profundidad.

Durante este análisis, donde se definen las necesidades que originan la solución, es fundamental evaluar las posibilidades presentes y futuras de utilizar XML. Sin embargo, su aplicación puede variar significativamente según el contexto en el que se pretenda implementar la solución.

A continuación, se presentan algunos ejemplos prácticos que se han encontrado en los proyectos desarrollados en diferentes empresas, los cuales ilustran claramente las oportunidades reales de XML en la integración de aplicaciones de gestión empresarial:

Sistemas de información Heterogéneos

Un ejemplo clásico y muy relevante es el uso de XML en la integración de sistemas de información heterogéneos.

En la actualidad, el mercado está saturado de aplicaciones específicas y/o verticales, junto con la presencia de aplicaciones generales y/o horizontales, lo que conduce a la necesidad frecuente de integrar aplicaciones desarrolladas en diferentes plataformas, modelos de datos y lenguajes.

Por lo tanto, cuando surge este tipo de problemas, típicamente nos encontramos con tres opciones:

- Mantener las aplicaciones que funcionan correctamente e integrarlas utilizando XML.
- Cambiar todos los sistemas para lograr una integración "de fábrica".
- No integrar y mantener las aplicaciones de forma independiente.

Es lógico que la decisión tomada deba basarse en consideraciones tecnológicas y en relación coste-beneficio.

Gestión de catálogos en Internet

XML también presenta oportunidades significativas para la gestión de catálogos electrónicos a través de Internet.

En comparación con otros lenguajes, el uso de XML permite que la gestión del contenido se limite a cargarlo en una base de datos, sin necesidad de crear manualmente cada página del catálogo.

Los catálogos se pueden administrar utilizando XML como medio de transporte de los datos de artículos, familias, categorías, descripciones, etc. Los formatos de visualización están determinados por XSL y su lenguaje XPath, que permite crear dinámicamente los contenidos de un catálogo.

Dispositivos móviles

En la actualidad, la movilidad del personal de una empresa es fundamental en muchos casos para su funcionamiento. Una de las principales complicaciones en estos escenarios suele ser proporcionar a los usuarios de dispositivos móviles información inmediata, oportuna y actualizada procedente del centro de datos.

Además, es importante que los usuarios tengan la capacidad de modificar esta información y actualizarla en el centro de datos sin necesidad de desplazarse físicamente, conectarse a la red y realizar actualizaciones.

La tecnología móvil nos permite utilizar dispositivos como PDAs, portátiles, teléfonos móviles, etc., que pueden comunicarse con un servidor intercambiando información mediante XML, WML y servicios web. Esto permite optimizar la dinámica de la empresa al contar con información fiable y actualizada en todo momento.

Servicios Web

Los Servicios Web están emergiendo como una tecnología que tendrá un impacto significativo en el futuro cercano. Esta forma innovadora de transmisión de datos permite la comunicación bidireccional entre aplicaciones utilizando protocolos estándar basados en XML, como SOAP (Simple Object Access Protocol), y especificaciones como UDDI (Universal Description, Discovery and Integration) y WSDL (Web Service Definition Language), aunque estos últimos aún no son estándares.

Estas tecnologías encapsulan el XML en paquetes de transmisión o mensajes (SOAP), facilitan la ubicación en Internet de los Servicios Web existentes, de manera similar a las Páginas Amarillas de Web Services (UDDI), y permiten que la aplicación que utiliza el Servicio Web comprenda las interfaces de comunicación de este último (WSDL).

Este conjunto de definiciones permite que las aplicaciones distribuidas se basen en tecnología abierta y estándares, a diferencia de los protocolos propietarios como DCOM o CORBA.

Como resultado, las aplicaciones de escritorio o web pueden comunicarse con otras aplicaciones remotas para obtener o gestionar datos, como si se tratara de una aplicación local, sin importar la plataforma en la que se encuentren, siempre que se respeten los estándares mencionados.

Introducción a JSON

Historia de JSON

JSON (JavaScript Object Notation) es un formato de intercambio de datos ligero que se emplea para la comunicación entre sitios web y servidores. Se basa en un subconjunto del lenguaje de programación JavaScript y es ampliamente utilizado en aplicaciones web para transmitir datos.

La historia de JSON se remonta a principios de la década de 2000, cuando surgió como una alternativa a XML (eXtensible Markup Language) para el intercambio de datos entre sistemas. Antes de JSON, XML era el formato predominante, pero se percibía como engorroso y difícil de analizar. En contraste, JSON fue diseñado para ser simple de leer y escribir, lo que lo convirtió en una opción más atractiva para los desarrolladores.

Una de las principales razones de la popularidad de JSON en el intercambio de datos es su simplicidad. Al estar basado en un subconjunto de JavaScript, resulta familiar para la mayoría de los desarrolladores. Además, al ser un formato de texto, puede ser fácilmente comprendido tanto por humanos como por máquinas.

JSON sigue las reglas de sintaxis de JavaScript pero es más estricto en su formato. Aunque es un subconjunto de JavaScript, puede ser utilizado con cualquier lenguaje de programación. Su facilidad para convertirse en objetos JavaScript y viceversa lo hace una opción conveniente para transmitir datos en aplicaciones web, donde los datos frecuentemente se manipulan en código JavaScript.

Además de su uso en comunicación entre sistemas, JSON también es utilizado como formato para almacenar datos en archivos. Es comúnmente empleado para guardar datos de configuración, ajustes de aplicaciones y otros tipos de datos que requieren un formato legible por máquinas.

Normas de uso

Un JSON puede tomar dos formas principales:

1. Una colección de pares nombre-valor: En esta forma, un JSON consiste en una serie de pares clave-valor, donde cada clave (nombre) debe ser una cadena de texto entre comillas dobles. El valor puede ser una cadena de texto, un array, un número, un valor booleano o nulo. Además, el valor en sí mismo puede ser otro objeto JSON. Esta estructura es similar a la de un objeto en JavaScript o a un diccionario en otros lenguajes de programación.

2. Una colección ordenada: JSON también se puede utilizar para almacenar una colección ordenada de objetos o valores. En este caso, la colección se asemeja a una matriz de elementos en los lenguajes de programación. Los elementos pueden ser primitivos (cadenas, números, booleanos, nulos) o objetos JSON. Aunque menos común que la primera forma, esta estructura sigue siendo útil en situaciones donde se requiere un orden específico para los elementos.

El siguiente ejemplo muestra una representación JSON de un objeto de persona:

```json
{
    "firstName": "John",
    "lastName": "Snow",
    "age": 25,
    "children": [],
    "spouse": null,
    "address": {
        "street": "7504 Taylor Drive",
        "city": "New York City",
        "state": "New York",
        "postalCode": "11238"
    },
```

```
    "phoneNumbers": [
        {
            "type": "mobile",
            "number": "212 555-3346"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

En el ejemplo anterior:

- Los primeros dos pares de nombre y valor asignan una cadena de texto a otra cadena de texto.
- El tercer par de nombre-valor mapea una cadena de texto "age" con el número 25.
- El cuarto par de nombre-valor mapea una cadena de texto "children" con una matriz vacía ([]).
- El quinto par de nombre-valor mapea una cadena de texto "spouse" con un valor nulo.
- El sexto par de nombre-valor mapea una cadena de texto "address" con otro objeto JSON.
- El séptimo par de nombre-valor mapea una cadena de texto "phoneNumbers" con una matriz de objetos JSON.

Reglas de sintaxis JSON:

- Un JSON object está rodeado por llaves {}.
- Los pares nombre-valor se agrupan por un colon (:) y separados por un coma (,).
- Un array comienza con un corchete izquierdo y termina con un soporte derecho [].
- Las comas finales y los ceros iniciales en un número están prohibidos.
- Los formatos octal y hexadecimal no están permitidos.
- Cada llave dentro del JSON debe ser única y debe estar entre comillas dobles.
- los boolean type coinciden sólo con dos valores especiales: true y false y los valores NULL están representados por el null literal (sin comillas).

Aplicaciones de JSON en el ámbito empresarial

Muchas empresas líderes, incluyendo Google, Facebook y Twitter, utilizan JSON, entre otras. Esto se debe principalmente a sus numerosas ventajas, siendo su simplicidad la más destacada. Además, se ha demostrado que el software que utiliza JSON tiende a tener un rendimiento superior, ya que la información se procesa de manera más rápida. A esto se suma su gran versatilidad y accesibilidad, tanto para humanos como para máquinas.

El uso de JSON en una empresa mejora la capacidad de respuesta del sitio web, ya que facilita el manejo de datos, lo que se traduce en una experiencia de usuario más fluida y sencilla. Una carga más rápida y una navegación más fluida contribuyen a una mejor experiencia de usuario, lo que a su vez puede mejorar la reputación del sitio en los motores de búsqueda.

Además, JSON también es útil para intercambiar datos entre sitios web de manera rápida y sencilla, lo que resulta especialmente valioso en situaciones de problemas de dominio.

A pesar de todas estas ventajas, todavía hay profesionales que prefieren el formato XML. Aunque XML también tiene sus ventajas, no es tan simple y ligero como JSON. Sin embargo, sigue siendo una opción sólida, aunque su complejidad puede hacerlo menos comprensible para algunos usuarios.