EBook Gratis

APRENDIZAJE xml

Free unaffiliated eBook created from
Stack Overflow contributors.

#xml

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: xml

It is an unofficial and free xml ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official xml.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con xml

Observaciones

XML es un lenguaje de marcado utilizado para almacenar datos jerárquicos en archivos de texto. También se conoce como datos semiestructurados, como JSON. XML es legible por máquina, pero también puede ser leído y producido por personas.

XML está formado por elementos, a veces denominados casualmente como una *sopa de etiquetas* , que pueden contener otros elementos y / o texto. Los elementos también pueden contener atributos.

XML se utiliza a menudo para el intercambio de datos entre plataformas, especialmente a través de Internet. También se usa cada vez más para almacenar datos semiestructurados en almacenes de datos NoSQL (bases de datos XML / almacenes de documentos). Además, tiene la flexibilidad de manejar datos orientados a documentos (texto con marcado), lo que lo hace muy popular en la industria editorial. XML también es ampliamente utilizado para archivos de configuración.

Una de las razones principales por las que XML se usa tanto es que está estandarizado, con muchos analizadores disponibles, incluido el código abierto. Esto hace que el costo de usar XML sea más bajo que la invención de la propia sintaxis nueva.

Puede encontrar más información sobre el origen y los objetivos de XML en la Recomendación oficial del W3C .

Hay dos versiones de XML, que se muestran en la siguiente tabla. Las ediciones de cada versión son solo revisiones de los documentos originales y no cambios de los estándares.

La primera versión de XML es 1.0 . XML 1.1 se cambió principalmente debido al cambio de versión de Unicode de 2.0 a 3.1 y especifica un conjunto de nuevas reglas para el uso e interpretación de los nuevos caracteres de Unicode.

Versiones

Versión	Fecha de lanzamiento
1.0	1998-02-10
1.1	2001-12-13

Examples

Instalación o configuración

XML es una sintaxis, lo que significa que un simple editor de texto es suficiente para comenzar.

Sin embargo, tener un editor específico de XML que le muestre cuándo y dónde su documento no está bien formado es casi indispensable para la productividad. Dichos editores también pueden permitirle validar documentos XML contra un esquema XML, o incluso generar esquemas XML a partir de documentos XML (y viceversa).

Algunos ejemplos de editores son oXygen, Atom, Eclipse y Altova XMLSpy. Una solución alternativa es utilizar un analizador XML de línea de comandos como Apache Xerces.

Los bloques de construcción básicos.

XML está hecho de bloques de construcción básicos, que son:

- elemento
- texto
- atributos
- comentarios
- instrucciones de procesamiento

Un elemento tiene corchetes angulares:

```
<element/>

<element>some content</element>
```

Un atributo aparece en una etiqueta de elemento de apertura:

```
<element
  attribute-name="attribute value"
  other-attribute='single-quoted value'>
  ...
</element>
```

El texto puede aparecer en cualquier lugar dentro o entre los elementos:

```
<element>some more <b>bold</b> text</element>
```

Los comentarios usan la siguiente sintaxis. Es importante saber que los comentarios XML, a diferencia de los lenguajes de programación, son parte del modelo y serán visibles para la aplicación sobre el analizador.

```
<!-- this is a comment -->
```

Las instrucciones de procesamiento permiten pasar mensajes a la aplicación consumidora (por ejemplo, cómo mostrar o una hoja de estilo, etc.). XML no restringe el formato de las instrucciones de procesamiento.

```
<?target-application these are some instructions?>
```

Más detalles sobre los bloques de construcción se pueden encontrar en este tema

Bien formado

Un documento XML es un archivo de texto que cumple con las reglas de buena formación de la especificación XML. Se dice que tal documento conforme está *bien formado* (no debe confundirse con *válido*). XML es muy estricto con una buena formación en comparación con otros lenguajes como HTML. Un archivo de texto que no está bien formado no se considera XML y no se puede usar al consumir aplicaciones.

Aquí hay algunas reglas que se aplican a los documentos XML:

1. XML utiliza una sintaxis muy autodescriptiva. Un prólogo define la versión XML y la codificación de caracteres:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   ```

2. Debe haber exactamente un elemento de nivel superior.

   Sin embargo, los comentarios, las instrucciones de procesamiento, así como la declaración XML inicial, también se permiten en el nivel superior. El texto y los atributos no lo son.

   ```
   <?xml version="1.0"?>
   <!-- some comments -->
   <?app a processing instruction?>
   <root/>
   <!-- some more comments -->
   ```

3. Los elementos pueden anidar, pero deben estar "correctamente anidados":

   ```
   <name>
     <first-name>John</first-name>
     <last-name>Doe</last-name>
   </name>
   ```

   Las etiquetas de inicio y fin de un elemento incrustado deben estar dentro de las etiquetas de inicio y fin de su elemento contenedor. Una superposición de elementos es ilegal. En particular, este no es un XML bien formado: `<foo><bar></foo></bar>`

4. Los atributos solo pueden aparecer en las etiquetas de elementos de apertura o en las etiquetas de elementos vacíos, no en las etiquetas de elementos de cierre. Si la sintaxis de atributo aparece entre los elementos, no tiene ningún significado y se analiza como texto.

   ```
   <person first-name="John" last-name="Doe"/>
   ```

   Esto no está bien formado: ~~`<person></person first-name="John"/>`~~

5. Los comentarios, las instrucciones de procesamiento, el texto y otros elementos pueden aparecer en cualquier lugar dentro de un elemento (es decir, entre su etiqueta de apertura y

de cierre) pero no dentro de las etiquetas.

```
<element>
    This is some <b>bold</b> text.
    <!-- the b tag has no particular meaning in XML -->
</element>
```

Este ejemplo no está bien formado: ~~`<element <-- comment --> />`~~

6. El carácter `<` puede no aparecer en el texto o en los valores de los atributos.
7. El `"` carácter puede no aparecer en los valores de atributo que se citan con `"` . El `'` carácter puede no aparecer en los valores de atributo que se citan con `'` .
8. La secuencia de caracteres `--` no puede aparecer en un comentario.
9. Los caracteres `<` y `&` literales deben ser escapados por sus respectivas entidades `<` y `&` .

Hola Mundo

```
<?xml version="1.0"?>
<?speech-generator voice="Siri"?>
<root xmlns:vocabulary="http://www.example.com/vocabulary">
  <!-- These are the standard greetings -->
  <vocabulary:greetings>
    <vocabulary:greeting xml:lang="en-US" type="informal">
      Hi!
    </vocabulary:greeting>
    <vocabulary:greeting xml:lang="en-US" type="intermediate">
      Hello!
    </vocabulary:greeting>
    <vocabulary:greeting xml:lang="en-US" type="formal">
      Good morning to <b>you</b>!
    </vocabulary:greeting>
  </vocabulary:greetings>
</root>
```

Espacios de nombres

Los nombres de elementos y atributos viven en los espacios de nombres que son URI. Los espacios de nombres están vinculados a los prefijos que se usan en los nombres reales de elementos y atributos, que se denominan QNames.

Este documento vincula un espacio de nombres al prefijo de `prefix` y define un espacio de nombres predeterminado, enlazado con la ausencia de prefijo.

```
<?xml version="1.0"?>
<document
    xmlns="http://www.example.com/default-namespace"
    xmlns:prefix="http://www.example.com/another-namespace">
  <prefix:element/>
</document>
```

Más detalles sobre los espacios de nombres se pueden encontrar en este tema

Lea Empezando con xml en línea: https://riptutorial.com/es/xml/topic/882/empezando-con-xml

Capítulo 2: Bloques de construcción

Examples

Elementos

Los elementos que vienen con corchetes angulares son el bloque de construcción más prominente de XML.

Los elementos pueden estar vacíos, en cuyo caso están hechos de una etiqueta vacía (observe la barra diagonal final):

```
<an-empty-element/>
```

O pueden tener contenido, en cuyo caso tienen una etiqueta de apertura (sin barra diagonal) y una etiqueta de cierre (barra de inicio):

```
<a-non-empty-element>Content</a-non-empty-element>
```

Los elementos pueden anidar (pero solo entre las etiquetas de apertura y cierre):

```
<parent-element>
  <child-element/>
  <another-child-element>
    Some more content.
  </another-child-element>
</parent-element>
```

Los nombres de los elementos se llaman QNames (nombres calificados). Todos los elementos anteriores no están en ningún espacio de nombres, pero los nombres de los elementos también se pueden definir en espacios de nombres usando prefijos como los siguientes:

```
<my-namespace:parent-element xmlns:my-namespace="http://www.example.com/">
  <my-namespace:child-element/>
  <my-namespace:another-child-element>
    Some more content.
  </my-namespace:another-child-element>
</my-namespace:parent-element>
```

Los espacios de nombres y los nombres de elementos se describen con mayor detalle en esta sección de la documentación .

Atributos

Los atributos son pares nombre-valor asociados con un elemento.

Están representados por valores en comillas simples o dobles dentro de la etiqueta del elemento

de apertura, o la etiqueta del elemento vacío si es un elemento vacío.

```
<document>
  <anElement foo="bar" abc='xyz'><!-- some content --></anElement>
  <anotherElement a="1"/>
</document>
```

Los atributos no están ordenados (a diferencia de los elementos). Los siguientes dos elementos tienen los mismos conjuntos de atributos:

```
<foo alpha="1" beta="2"/>

<foo beta="2" alpha="1"/>
```

Los atributos no se pueden repetir en el mismo elemento (a diferencia de los elementos). El siguiente documento no está bien formado: ~~`<foo a="x" a="y"/>`~~ porque el atributo `a` aparece dos veces en el mismo elemento.

El siguiente documento está bien formado. Los valores pueden ser idénticos, es el nombre del atributo que no se puede repetir.

```
<foo a="x" b="x"/>
```

Los atributos no se pueden anidar (a diferencia de los elementos).

Texto

El texto está formado por todos los caracteres fuera de cualquier marca (etiquetas de elementos de apertura, etiquetas de elementos de cierre, etc.).

```
<?xml version="1.0"?>
<document>
  This is some text and <b>this is some more text</b>.
</document>
```

La terminología XML precisa para texto es *datos de caracteres* . La especificación XML en realidad usa la palabra *texto* para todo el documento XML, o una entidad analizada, porque define XML en el nivel sintáctico. Sin embargo, algunos modelos de datos como el XDM (XQuery y XPath Data Model), que representan documentos XML como árboles, se refieren a los datos de caracteres como *nodos de texto* , de modo que el *texto a* menudo se entiende como un sinónimo de datos de caracteres en la práctica.

Los datos de caracteres pueden no contener un carácter `<` esto se interpretaría como el primer carácter de una etiqueta de elemento de apertura, ni tampoco puede contener la secuencia de caracteres `]]>` . Los caracteres apropiados deben escaparse con una referencia de entidad en su lugar.

```
<?xml version="1.0"?>
<document>
```

```
  It is fine to escape the &lt; character, as well as ]]&gt;.
</document>
```

Por conveniencia, también se puede escapar una gran parte del texto con una sección CDATA (pero la secuencia `]]>` todavía no está permitida por razones obvias):

```
<?xml version="1.0"?>
<document>
  <![CDATA[
    In a CDATA section, it is fine to write < or even & and entity references
    such as &amp; are not resolved.
  ]]>
</document>
```

Comentarios

Los comentarios en XML se ven así:

```
<!-- This is a comment -->
```

Pueden aparecer en el contenido del elemento o de nivel superior:

```
<?xml version="1.0"?>
<!-- a comment at the top-level -->
<document>
  <!-- a comment inside the document -->
</document>
```

Los comentarios no pueden aparecer dentro de las etiquetas o dentro de los atributos:

~~<element <!-- comment with --- inside --> />~~

o

~~<element attr="<!-- comment with --- inside -->"/>~~

No están bien formados.

La secuencia de caracteres `--` no puede aparecer en medio de un comentario. Esto no es XML bien formado:

~~<!-- comment with --- inside -->~~

Los comentarios en XML, a diferencia de otros lenguajes como C ++, son **parte del modelo de datos** : se analizan, reenvían y son visibles para la aplicación consumidora.

Instrucciones de procesamiento

Una instrucción de procesamiento se utiliza para transmitir directamente cierta información o instrucciones a la aplicación a través del analizador.

```
<?my-application some instructions ?>
```

El token después del signo de interrogación inicial (aquí `my-application`) se denomina destino e identifica la aplicación a la que se dirige la instrucción. Lo que sigue no se especifica más y depende de la aplicación interpretarla. Las referencias de entidades y personajes no son reconocidas.

Puede aparecer en el nivel superior o en el contenido del elemento.

Lea Bloques de construcción en línea: https://riptutorial.com/es/xml/topic/1590/bloques-de-construccion

Capítulo 3: Catálogos XML

Introducción

Un catálogo XML está formado por entradas de uno o más archivos de entrada de catálogo. Un archivo de entrada de catálogo es un archivo XML cuyo elemento de documento es catálogo y cuyo contenido sigue el DTD de catálogo XML definido por OASIS en http://www.oasis-open.org/committees/entity/spec.html . La mayoría de los elementos son entradas de catálogo, cada una de las cuales sirve para asignar un identificador o URL a otra ubicación.

Examples

Entrada de catálogo para resolver ubicación de DTD

1

2 3

```
<public
   publicId="-//OASIS//DTD DocBook XML V4.5//EN"  4
   uri="docbook45/docbookx.dtd"/>

<system
   systemId="http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd"  5
   uri="docbook45/docbookx.dtd"/>

<system
   systemId="docbook4.5.dtd"    6
   uri="docbook45/docbookx.dtd"/>
```

Lea Catálogos XML en línea: https://riptutorial.com/es/xml/topic/10875/catalogos-xml

Capítulo 4: DTD

Introducción

La Declaración de Tipo de Documento XML comúnmente conocida como DTD es una manera de describir con precisión el lenguaje XML. Las DTD comprueban la validez, la estructura y el vocabulario de un documento XML con las reglas gramaticales del lenguaje XML apropiado. Una DTD define la estructura y los elementos legales y atributos de un documento XML.

Examples

Declaración de Tipo de Documento

Un documento XML puede contener una DTD. DTD significa *Declaración de Tipo de Documento* . Una DTD comienza con `<!DOCTYPE root-element-name >` donde `doc-element-name` debe coincidir con el nombre del llamado elemento de documento (el elemento en el nivel superior).

```
<?xml version="1.0"?>
<!DOCTYPE document>
<document>
  <!-- the rest of the document -->
</document>
```

Entidades

Un DTD puede contener declaraciones de entidad.

```
<?xml version="1.0"?>
<!DOCTYPE document [
  <!ENTITY my-entity "This is the replacement text">
]>
<document>
  <!-- the rest of the document -->
</document>
```

Las entidades se describen en detalle en este tema .

Documento XML con una DTD interna.

Una DTD se conoce como una DTD interna si se declaran elementos dentro de los archivos XML. Para referenciarlo como DTD interno, el atributo independiente en la declaración XML debe establecerse en sí.

Un XML que describe una nota que contiene propiedades hacia, desde y mensaje junto con DTD interno se verá así:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
```

```
<!DOCTYPE note [
<!ELEMENT note (to,from,message>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT message (#PCDATA)>
]>
<note>
<to>Mr.X</to>
<from>Mr.Y</from>
<message>Stack Overflow is awesome </message>
</note>
```

Documento XML con una DTD externa

En la DTD externa los elementos se declaran fuera del archivo XML. Se accede a ellos especificando los atributos del sistema que pueden ser el archivo .dtd legal o una URL válida. Para referenciarlo como DTD externo, el atributo independiente en la declaración XML debe establecerse como no.

A continuación se muestra un XML que describe una nota que contiene propiedades para, desde y el mensaje.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Mr.X</to>
  <from>Mr.Y</from>
  <message>Stack Overflow is awesome</message>
</note>
```

DTD externa para el XML anterior, *note.dtd* se proporciona a continuación

```
<!DOCTYPE note [
<!ELEMENT note (to,from,message>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT message (#PCDATA)>
]>
```

Lea DTD en línea: https://riptutorial.com/es/xml/topic/3897/dtd

Capítulo 5: Entidades

Observaciones

Desde una perspectiva de almacenamiento, un documento XML está hecho de entidades. Una de las entidades es la entidad de documento, que es el principal documento XML en sí.

Las entidades se pueden clasificar como tales (clasificadas tentativamente por orden de uso descendente):

- **entidad de documento** : este es el archivo XML principal.
- **Entidades generales internas** : esta es la más común además de la entidad de documento, y la que más usuarios de XML conocen. A menudo, la palabra **entidad** se usa casualmente para ellos. Permiten especificar algunos accesos directos para textos de reemplazo más largos en el contenido del documento. Están declarados en la DTD.
- **el subconjunto DTD externo** : otro archivo en el que se subcontrata parte de la DTD.
- **entidades de parámetros** : accesos directos, para su uso en la DTD.
- **Entidades generales analizadas externas** : son fragmentos XML almacenados en otros archivos.
- **entidades no analizadas** : pueden ser cualquier archivo en el que XML no impone restricciones, incluidas imágenes, sonidos, etc.

En muchos casos, un documento XML consiste únicamente en la entidad del documento.

Examples

Entidades generales predefinidas

XML pre-define cinco entidades generales que se pueden usar sin declararlas:

```
& " ' < >
```

Están asociados con los nombres `amp` , `quot` , `apos` , `lt` y `gt` .

```
<?xml version="1.0"?>
<entities>
  &amp; is an ampersand.
  &quot; is a quote.
  &apos; is an apostrophe.
  &lt; is a lower-than sign.
  &gt; is a greater-than sign.
</entities>
```

Entidades generales (internas) definidas por el usuario

Es posible definir las propias entidades generales. La declaración se produce en el subconjunto

DTD, con un nombre y el texto de reemplazo asociado.

Luego se puede usar en el documento usando la sintaxis de referencia de la entidad `&...;` , ya sea en texto, o en valores de atributo.

```
<?xml version="1.0"?>
<!DOCTYPE my-document [
  <!ENTITY my-entity "This is my entity">
]>
<my-document>
    The entity was declared as follows: &my-entity;
    <element attribute="Entity: &my-entity;"/>
</my-document>
```

Entidades analizadas externas

Los fragmentos XML, también conocidos con el nombre de *entidades analizadas externas* , pueden almacenarse en archivos separados.

Los fragmentos XML, a diferencia de los documentos XML, son menos restrictivos, ya que pueden aparecer varios elementos de nivel superior, así como nodos de texto. Al igual que un documento XML, una entidad analizada externa puede comenzar con una declaración XML, pero esta declaración no se considera parte de su texto de reemplazo.

Este es un ejemplo de entidad analizada externa:

```
<?xml version="1.0" encoding="UTF-8"?>
This is some text
<element/>
<element/>
```

Una entidad analizada externa se puede declarar en un documento XML, en la DTD, y se puede usar con una referencia de entidad, que tiene la misma sintaxis que para las entidades internas generales:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
<!ENTITY fragment SYSTEM "fragment.xml">
]>
<root>
    &fragment;
</root>
```

Con la referencia de la entidad resuelta, este documento es equivalente a:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
<!ENTITY fragment SYSTEM "fragment.xml">
]>
<root>
  This is some text
  <element/>
```

```
  <element/>
</root>
```

Cada etiqueta de elemento de apertura en una entidad analizada externa debe tener una etiqueta de final correspondiente: no está permitido distribuir elementos individuales sobre varias entidades, ni difundir el marcado.

Se requiere un analizador de validación para resolver la referencia de la entidad e incluir su texto de reemplazo en el documento como se indicó anteriormente. Un analizador no validador puede omitir esto y, en cambio, decirle a la aplicación consumidora que hay una referencia no resuelta a una entidad analizada externa.

Lea Entidades en línea: https://riptutorial.com/es/xml/topic/2302/entidades

Capítulo 6: Escapando

Observaciones

Los caracteres pueden escaparse en XML usando referencias de entidades y referencias de caracteres, o secciones CDATA.

XML pre-define cinco entidades:

Entidad nombrada	Texto de reemplazo
amperio	Y
cotizar	"
apos	'
es	<
gt	>

Las aplicaciones consumidoras no sabrán si cada personaje se ha escapado o no, y cómo.

Examples

Ampersand

El carácter `&` aparece primero en las referencias de entidad y debe escaparse en el contenido del elemento o en el contenido del atributo.

```
<?xml version="1.0"?>
<document attribute="An ampersand is escaped as &amp;">
  An ampersand can also be escaped as &amp; in element content.
</document>
```

Signo más bajo que

El `<` carácter aparece primero en las etiquetas de entidad y debe escaparse en el contenido del elemento o en el contenido del atributo.

```
<?xml version="1.0"?>
<document attribute="A lower-than sign is escaped as &lt;">
  2 + 2 &lt; 5
</document>
```

Signo mayor que

La secuencia de caracteres `]]>` no está permitida en el contenido del elemento. La forma más fácil de escapar es escapar `>` como `>` .

```
<?xml version="1.0"?>
<document>
  The sequence ]]&gt; cannot appear in element content.
</document>
```

Apóstrofes y citas

Los valores de los atributos pueden aparecer entre comillas simples o dobles. El personaje apropiado debe ser escapado.

```
<?xml version="1.0"?>
<document
  quot-attribute="This is a &quot;double quote&quot; and this one is 'simple'"
  apos-attribute='This is a &apos;simple quote&apos; and this one is "double"'>
</document>
```

Secciones de CDATA

Las partes más largas del texto que contienen caracteres especiales pueden escaparse con una sección CDATA. Las secciones CDATA solo pueden aparecer en el contenido del elemento.

```
<?xml version="1.0"?>
<document>
  This is a CDATA section : <![CDATA[ plenty of special characters like & < > " ; ]]>
</document>
```

Una sección CDATA no puede contener la secuencia `]]>` porque la termina.

Referencias de caracteres

Los caracteres pueden escaparse utilizando referencias de caracteres, en contenido de elementos o valores de atributos. Su punto de código Unicode se puede especificar en decimal o hexadecimal.

```
<?xml version="1.0"?>
<document>
  The line feed character can be escaped with a decimal (&#10;) or hex (&#xA;)
  representation of its Unicode codepoint (10).
</document>
```

XML restringe los caracteres que pueden aparecer en un documento, incluso escapados. En particular, los únicos caracteres de control permitidos son el salto de línea (10), el retorno de carro (13) o la pestaña horizontal (9).

Lea Escapando en línea: https://riptutorial.com/es/xml/topic/3685/escapando

Capítulo 7: Espacios de nombres

Observaciones

Los nombres de elementos y atributos en XML se denominan QNames (nombres calificados).

Un QName está hecho de:

- un espacio de nombres (un URI)
- un prefijo (un NCName, NC porque no contiene dos puntos)
- un nombre local (un NCName)

Solo el espacio de nombres y el nombre local son relevantes para comparar dos QNames. El prefijo es solo un proxy para el espacio de nombres.

El espacio de nombres y el prefijo son opcionales, pero el espacio de nombres siempre está presente si el prefijo está presente (esto se garantiza en el nivel sintáctico, por lo que no se puede hacer mal).

La representación léxica de un QName es `prefix:local-name` . El espacio de nombres se enlaza por separado utilizando los atributos `xmlns:...` especiales (recordatorio: los atributos que comienzan con *xml* están reservados en XML).

Si el prefijo está vacío, no se utilizan dos puntos en la representación léxica del QName, que solo contiene el `local-name` . Los QNames con un prefijo vacío no tienen espacio de nombres (si no hay un espacio de nombres predeterminado dentro del alcance) o están en el espacio de nombres predeterminado.

Examples

Unir un prefijo a un espacio de nombres

Un espacio de nombres es un URI, pero para evitar la verbosidad, los prefijos se utilizan como un proxy.

En el siguiente ejemplo, el prefijo `my-prefix` está vinculado al espacio de nombres `http://www.example.com/my-namespace` utilizando el atributo especial `xmlns:my-prefix` (`my-prefix` puede reemplazarse con cualquier otro prefijo) :

```
<?xml version="1.0"?>
<my-prefix:foo xmlns:my-prefix="http://www.example.com/my-namespace">
  <!-- the element my-prefix:foo
       lives in the namespace http://www.example.com/my-namespace -->
</my-prefix:foo>
```

Ausencia de espacio de nombres

En XML, los nombres de elementos y atributos viven en espacios de nombres.

Por defecto, no están en ningún espacio de nombres:

```
<?xml version="1.0"?>
<foo attr="value">
  <!-- the foo element is in no namespace, neither is the attr attribute -->
</foo>
```

Irrelevancia de los prefijos

Estos dos documentos son equivalentes semánticamente, ya que los espacios de nombres son importantes, no los prefijos.

```
<?xml version="1.0"?>
<myns:foo xmlns:myns="http://www.example.com/my-namespace">
</myns:foo>

<?xml version="1.0"?>
<ns:foo xmlns:ns="http://www.example.com/my-namespace">
</ns:foo>
```

Espacio de nombres predeterminado

El espacio de nombres predeterminado es el espacio de nombres correspondiente a la ausencia de cualquier prefijo. Se puede declarar con el atributo `xmlns` especial.

```
<?xml version="1.0"?>
<foo xmlns="http://www.example.com/my-namespace">
  <!-- the element foo is in the namespace
       http://www.example.com/my-namespace -->
</foo>
```

Si no se declara ningún espacio de nombres predeterminado, los nombres sin prefijo no están en ningún espacio de nombres.

Atributos de nombres sin prefijo

Los elementos y atributos se comportan de manera diferente con respecto a los espacios de nombres predeterminados. Esto es a menudo la fuente de confusión.

Un atributo cuyo nombre no tiene prefijo vive en ningún espacio de nombres, **también cuando un espacio de nombres predeterminado está dentro del alcance** .

```
<?xml version="1.0"?>
<foo attr="value" xmlns="http://www.example.com/my-namespace">
  <!-- The attribute attr is in no namespace, even though
       a default namespace is in scope. The element foo,
       however, is in the default namespace. -->
</foo>
```

Alcance de los enlaces de espacio de nombres

Un enlace de espacio de nombres (`xmlns` especiales o `xmlns:...` atributo) está en el alcance de todos los descendientes del elemento que lo incluye, incluido este elemento.

```
<?xml version="1.0"?>
<root>
  <my:element xmlns:my="http://www.example.com/ns1">
    <!-- here, the prefix my is bound to http://www.example.com/ns1 -->
  </my:element>
  <my:element xmlns:my="http://www.example.com/ns2">
    <!-- here, the prefix my is bound to http://www.example.com/ns2 -->
  </my:element>
</root>
```

El enlace se puede anular en un elemento anidado (aunque esto afecta la legibilidad):

```
<?xml version="1.0"?>
<my:element xmlns:my="http://www.example.com/ns1">
  <!-- here, the prefix my is bound to http://www.example.com/ns1 -->
  <my:first-child-element/>

  <my:child-element xmlns:my="http://www.example.com/ns2">
    <!-- here, the prefix my is bound to http://www.example.com/ns2,
         including for the element my:child-element -->
  </my:child-element>

  <!-- here, the prefix my is bound to http://www.example.com/ns1 -->
  <my:last-child-element/>

</my:element>
```

Es muy común declarar todos los enlaces de espacio de nombres en el elemento raíz, lo que mejora la legibilidad.

```
<?xml version="1.0"?>
<root
  xmlns="http://www.example.com/default-namespace"
  xmlns:ns1="http://www.example.com/ns1"
  xmlns:ns2="http://www.example.com/ns2">

  <ns1:element>
    <ns2:other-element/>
  </ns1:element>

</root>
```

Lea Espacios de nombres en línea: https://riptutorial.com/es/xml/topic/1593/espacios-de-nombres

Capítulo 8: Esquema XML

Introducción

El esquema XML se conoce comúnmente como definición de esquema XML (XSD). Se utiliza para describir y validar la estructura y el contenido de los datos XML. El esquema XML define los elementos, atributos y tipos de datos.

Examples

Un ejemplo de documento XSD

A continuación se proporciona un XSD que describe una información de contacto sobre una compañía.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://NamespaceTest.com/CommonTypes"
                  xmlns:xs="http://www.w3.org/2001/XMLSchema"
                  elementFormDefault="qualified">
<xs:element name="contact">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="name" type="xs:string" />
            <xs:element name="company" type="xs:string" />
            <xs:element name="phone" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

En el ejemplo anterior los atributos en segunda línea.

> <xs: schema targetNamespace = " http://NamespaceTest.com/CommonTypes " xmlns: xs = " http://www.w3.org/2001/XMLSchema "
> elementFormDefault = "calificado">

Los atributos 'targetnamespace' y elementFormDefault son opcionales.

Lea Esquema XML en línea: https://riptutorial.com/es/xml/topic/8983/esquema-xml

Creditos

S. No	Capítulos	Contributors
1	Empezando con xml	Al.G., Burkart, Community, Elizaveta Revyakina, Ghislain Fourny, Joe, JohnRC, Kin, MAZux, Mohammad Arman, RamenChef, TuringTux, w5m, Wolfgang Schindler
2	Bloques de construcción	Ghislain Fourny, Hoylen
3	Catálogos XML	Mistletoe
4	DTD	Dipesh Poudel, Ghislain Fourny
5	Entidades	Ghislain Fourny
6	Escapando	Ghislain Fourny
7	Espacios de nombres	Ghislain Fourny
8	Esquema XML	Dipesh Poudel