

LMSGI03.2 Validación



≡ Validación de documentos

≡ Validación con DTD

≡ Validación con esquema

≡ Ejemplos

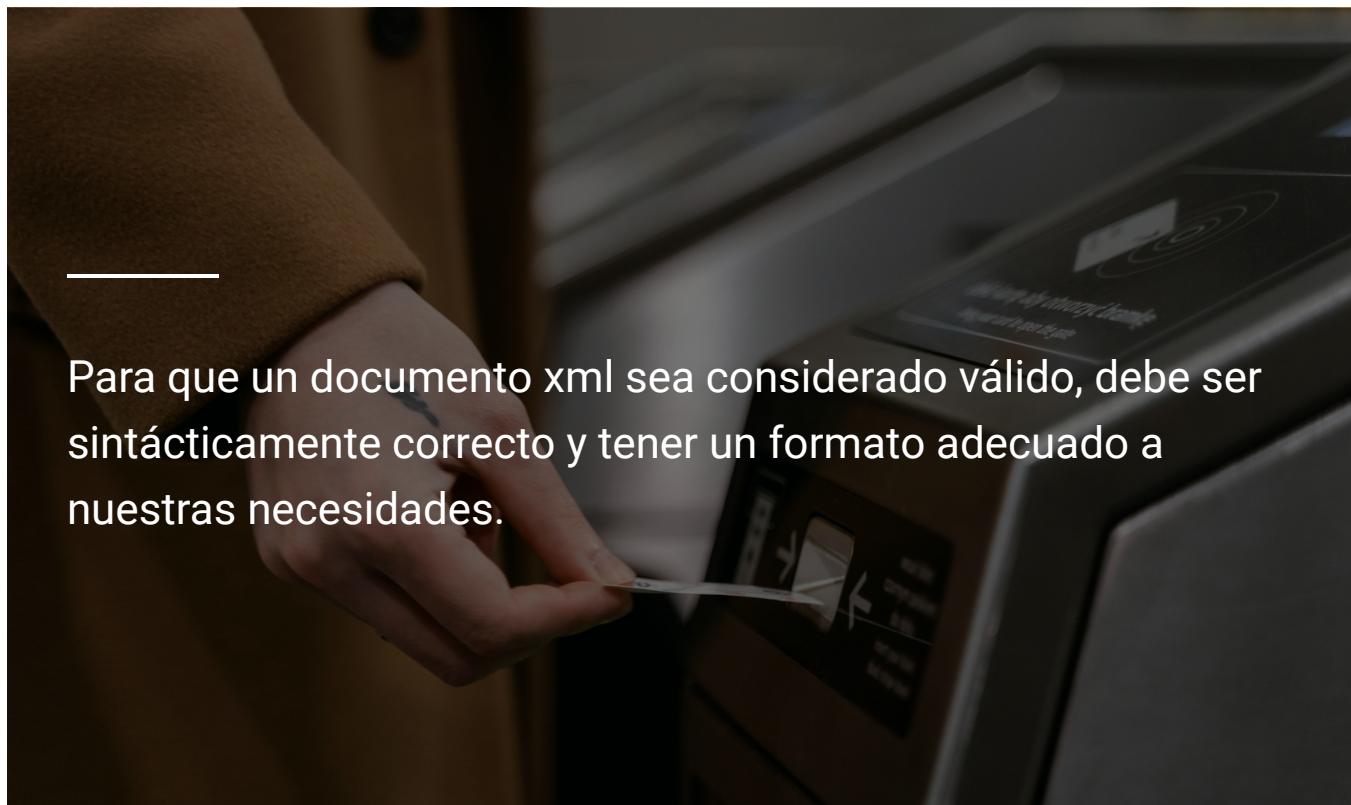
?

Autoevaluación

CONTENIDO PDF

≡ Contenido PDF

Validación de documentos

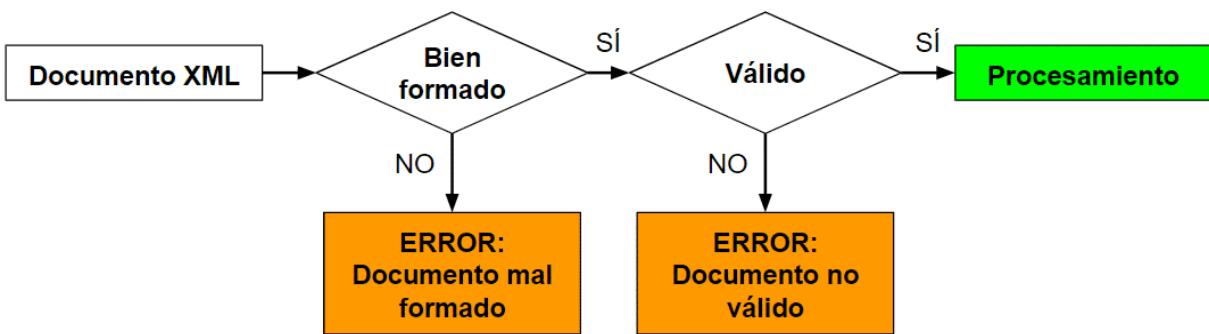


Para que un documento xml sea considerado válido, debe ser sintácticamente correcto y tener un formato adecuado a nuestras necesidades.

Introducción

Un documento XML válido proporciona confianza en su exactitud y facilita su procesamiento por parte de aplicaciones y sistemas que esperan un formato específico. La validación garantiza la integridad y consistencia de los datos, así como su conformidad con los estándares establecidos.

La validación de un documento XML es el proceso de verificar si el documento cumple con las reglas y estructuras especificadas en un esquema ó DTD (Document Type Definition). Esto implica asegurar que el documento esté correctamente formado y que siga las reglas semánticas definidas para los elementos y atributos. Esto ayuda a detectar errores y problemas de sintaxis durante la etapa de desarrollo, lo que contribuye a la calidad y fiabilidad del documento y de las aplicaciones que lo utilizan.



Elaboración propia

DTD vs Esquema

La validación de un documento XML utilizando un esquema (XSD) y un DTD (Document Type Definition) presenta diferencias significativas en cuanto a su estructura y funcionalidad:

1

Estructura y sintaxis: Los esquemas XML (XSD) proporcionan una forma más flexible y poderosa de definir la estructura y restricciones de un documento XML en comparación con los DTD. Los esquemas XML se escriben en XML, lo que facilita su comprensión y manipulación, mientras que los DTD tienen su propia sintaxis específica.

2

Capacidad de expresión: Los esquemas XML son más expresivos y pueden representar restricciones más complejas en comparación con los DTD. Por ejemplo, los esquemas XML permiten definir tipos de datos específicos y establecer relaciones más sofisticadas entre los elementos, lo que facilita la validación de

datos más precisos y detallados. Además, los DTD no soportan espacios de nombres.

3

Reutilización y modularidad: Los esquemas XML fomentan la reutilización y la modularidad al permitir la definición de elementos y tipos de datos que pueden ser compartidos entre múltiples documentos XML. Esto facilita la creación y mantenimiento de esquemas coherentes y consistentes en entornos de desarrollo complejos.

4

Compatibilidad y estándares: Aunque los DTD son más antiguos y están más ampliamente soportados en herramientas y aplicaciones, los esquemas XML son el estándar preferido en la mayoría de los casos debido a su mayor flexibilidad y capacidad. Sin embargo, algunos sistemas heredados ó herramientas pueden requerir el uso de DTD por razones de compatibilidad.

En resumen, mientras que los DTD son más simples y tienen una sintaxis más compacta, los esquemas XML ofrecen una mayor flexibilidad, expresividad y capacidades de reutilización, lo que los convierte en la opción preferida para la validación de documentos XML en entornos modernos y complejos.

Validación con DTD



El DTD es un documento de texto plano ó reglas que definen las normas a usar para crear un documento xml válido.

Introducción

El DTD (Document Type Definition-Definición de Tipo de Documento) es un archivo de texto ó reglas cuyo contenido son las definiciones de las etiquetas y sus atributos con los que se puede trabajar en un determinado documento xml.

Es decir, el DTD contiene la estructura de los datos en un documento XML con todas las restricciones a tener en cuenta y así permitir su validación. De esta forma, es correcto también llamarlo gramática ó plantilla.

Es necesario decir que la utilización de DTD es opcional, aunque conveniente. Además, la sintaxis es distinta a la de XML, ya que no se pueden definir tipos de datos por parte de los usuarios.

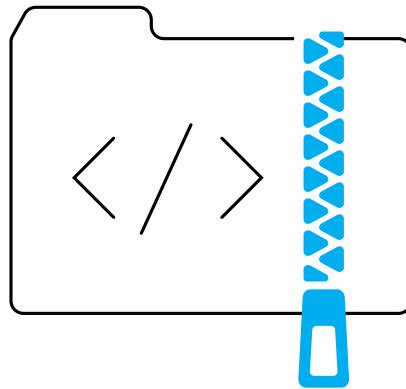
Asociar DTD en XML

Existen dos maneras de utilizar DTD en un documento XML. **El DTD puede estar incluido en el propio documento ó ser un documento externo:**

- Cuando las reglas del DTD están definidas dentro del documento XML, se ubican entre corchetes después del nombre del ejemplar en el elemento <!DOCTYPE>.

```
<!DOCTYPE NombreEjemplar[  
    ... declaraciones ...  
]>
```

- Cuando las reglas del DTD está en un documento externo, definimos el DTD externo en un fichero de texto plano con extensión ".dtd" y se especifica un URI donde puede localizarse. Además, en el documento XML se puede incluir el atributo `standalone="no"` en la declaración de XML.



Custom code isn't available in PDF format

Por ejemplo, en la imagen vemos que el documento "prueba02.xml" quiere validar su contenido con un documento dtd externo que está en la misma carpeta llamado "films.dtd".

The screenshot shows two windows of the Notepad+ application. The top window is titled 'D:\Mis Descargas\prueba02.xml - Notepad+' and contains the following XML code:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE catalogo SYSTEM "films.dtd">
<catalogo>
  <films>
    <film codfilm="COD1">Los siete samuráis</film>
    <film codfilm="COD2">Pulp Fiction</film>
    <film codfilm="COD3">Rashomon</film>
    <film codfilm="COD4">Dersu Uzala</film>
  </films>
  <directores>
    <director filmografia="COD2">Quentin Tarantino</director>
    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
  </directores>
</catalogo>
```

The bottom window is titled 'D:\Mis Descargas\films.dtd - Notepad+' and contains the following DTD code:

```
<!ELEMENT catalogo (films, directores)>
<!ELEMENT films (film)*>
<!ELEMENT film (#PCDATA)>
<!ATTLIST film codfilm ID #REQUIRED>
<!ELEMENT directores (director)*>
<!ELEMENT director (#PCDATA)>
<!ATTLIST director filmografia IDREFS #REQUIRED>
```

The screenshot shows two windows of Notepad++ side-by-side. The left window, titled 'prueba02.xml', contains the following XML code:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE catalogo SYSTEM "films.dtd">
3 <catalogo>
4   <films>
5     <film codfilm="COD1">Los siete samuráis</film>
6     <film codfilm="COD2">Pulp Fiction</film>
7     <film codfilm="COD3">Rashomon</film>
8     <film codfilm="COD4">Dersu Uzala</film>
9   </films>
10  <directores>
11    <director filmografia="COD2">Quentin Tarantino</director>
12    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
13  </directores>
14 </catalogo>
```

The right window, titled 'films.dtd', contains the following DTD code:

```
1 <!ELEMENT catalogo (films, directores)>
2 <!ELEMENT films (film)*>
3 <!ELEMENT film (#PCDATA)>
4 <!ATTLIST film codfilm ID #REQUIRED>
5 <!ELEMENT directores (director)*>
6 <!ELEMENT director (#PCDATA)>
7 <!ATTLIST director filmografia IDREFS #REQUIRED>
8
```

DTD externo

Con esta línea de código, se determina dónde está el DTD.

```
D:\Mis Descargas\prueba02.xml - Notepad++
Archivo Editar Buscar Vista Codificación Lenguaje Configuración Herramientas Macro Ejecutar Complementos Pestañas
prueba02.xml ✘ ✗
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE catalogo SYSTEM "films.dtd">
3  <catalogo>
4    <films>
5      <film codfilm="COD1">Los siete samuráis</film>
6      <film codfilm="COD2">Pulp Fiction</film>
7      <film codfilm="COD3">Rashomon</film>
8      <film codfilm="COD4">Dersu Uzala</film>
9    </films>
10   <directores>
11     <director filmografia="COD2">Quentin Tarantino</director>
12     <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
13   </directores>
14 </catalogo>
D:\Mis Descargas\films.dtd - Notepad++ +
```

```
D:\Mis Descargas\films.dtd - Notepad++ +
Archivo Editar Buscar Vista Codificación Lenguaje Configuración Herramientas Macro Ejecutar Complementos Pestañas
films.dtd ✘ ✗
films.dtd ✘ ✗
1  <!ELEMENT catalogo (films, directores)>
2  <!ELEMENT films (film)*>
3  <!ELEMENT film (#PCDATA)>
4  <!ATTLIST film codfilm ID #REQUIRED>
5  <!ELEMENT directores (director)*>
6  <!ELEMENT director (#PCDATA)>
7  <!ATTLIST director filmografia IDREFS #REQUIRED>
8
```

Ruta al DTD

Se puede observar que tanto el documento XML como el DTD están en la misma carpeta.

```

D:\Mis Descargas\prueba02.xml - Notepad++
Archivo Editar Buscar Vista Codificación Lenguaje Configuración Herramientas Macro Ejecutar Complementos Pestañas
prueba02.xml ✘ ✗
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE catalogo SYSTEM "films.dtd">
3  <catalogo>
4    <films>
5      <film codfilm="COD1">Los siete samuráis</film>
6      <film codfilm="COD2">Pulp Fiction</film>
7      <film codfilm="COD3">Rashomon</film>
8      <film codfilm="COD4">Dersu Uzala</film>
9    </films>
10   <directores>
11     <director filmografia="COD2">Quentin Tarantino</director>
12     <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
13   </directores>
14 </catalogo>
D:\Mis Descargas\films.dtd - Notepad++
Archivo Editar Buscar Vista Codificación Lenguaje Configuración Herramientas Macro Ejecutar Complementos Pestañas
films.dtd ✘ ✗
films.dtd ✘ ✗
1  <!ELEMENT catalogo (films, directores)>
2  <!ELEMENT films (film)*>
3  <!ELEMENT film (#PCDATA)>
4  <!ATTLIST film codfilm ID #REQUIRED>
5  <!ELEMENT directores (director)*>
6  <!ELEMENT director (#PCDATA)>
7  <!ATTLIST director filmografia IDREFS #REQUIRED>
8

```

Reglas del DTD

Conjunto de reglas que define la estructura y las normas que debe cumplir el documento XML para considerarse válido.

Elaboración propia

Componentes de un DTD

Los componentes principales de un DTD son:

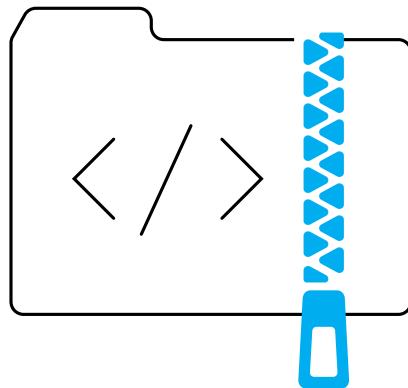
- Elementos
- Atributos
- Entidades

Elementos de un DTD

Constituyen la base de la estructura de un DTD. Lo primero que hay que hacer es la dependencia jerárquica, especificando el nodo principal y, entre paréntesis, sus descendientes si los tuviera, separados por el carácter ",". Por ejemplo:

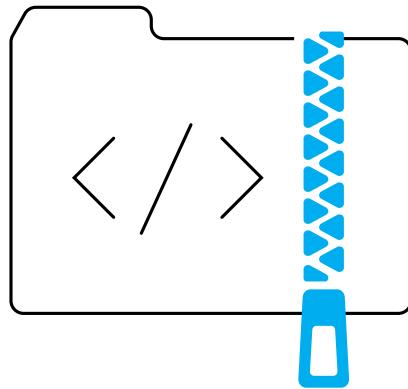
```
<!ELEMENT A (B, C)>
```

Además, cada uno de los elementos debe ser de un tipo entre:



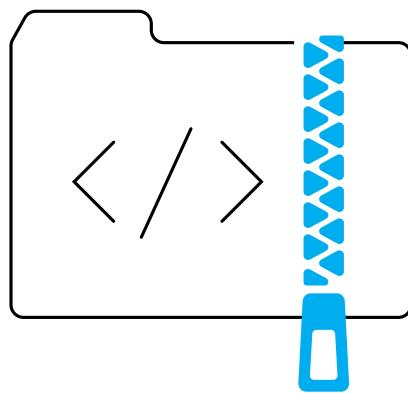
Custom code isn't available in PDF format

Para especificar la ocurrencia de un elemento en un documento XML se utilizan:



Custom code isn't available in PDF format

Por ejemplo, un documento xml con un dtd interno:



Custom code isn't available in PDF format

Atributos de un DTD

Los atributos sirven para enriquecer ó añadir información a un elemento y conviene no abusar de su uso y es mejor descomponer en otros elementos. Si un elemento tiene más de un

atributo, se definen uno detrás de otro, separados por espacios. Por ejemplo:

```
<!ATTLIST Elemento Atributo Tipo Valor>
```

Donde:

Elemento —

Nombre del elemento al que se quiere añadir el atributo.

Atributo —

Nombre del atributo que se quiere añadir.

Tipo —

Tipo	Descripción
CDATA	Para almacenar valores alfanuméricos (texto).
Enumeraciones	Para especificar varios posibles valores: <!ATTLIST cliente vip (S I NO) #REQUIRED>
ID	Su contenido es único, identificando de manera única cada elemento.
IDREF	Hace referencia al identificador de otro elemento. <!DOCTYPE catalogo [<!ELEMENT catalogo (films, directores)>

Tipo	Descripción
	<pre><!ELEMENT films (film)*> <!ELEMENT film (#PCDATA)> <!ATTLIST film codfilm ID #REQUIRED> <!ELEMENT directores (director)*> <!ELEMENT director (#PCDATA)> <!ATTLIST director filmografia IDREFS #REQUIRED>]></pre>
IDREFS	Hace referencia a un conjunto de identificadores.
NOTATION	Se especifican datos que no son XML.
ENTITY	El atributo es una entidad: <pre><!ENTITY copyright "Copyright BirtLH."></pre>
NMTOKEN	El atributo contiene letras, digitos y los caracteres punto("."), guion ("-"), subrayado ("_") y dos puntos ("::"). El valor puede aparecer varias veces. <pre><!ATTLIST articulo codigo NMTOKEN #REQUIRED></pre>

Valor —

Valor	Descripción
#IMPLIED	El atributo es opcional.
#REQUIRED	El atributo es obligatorio.
#FIXED	Se especifica un valor que no se puede cambiar.

Valor	Descripción
#DEFAULT	Se especifica un valor que se puede cambiar.

Entidades de un DTD

Las entidades se utilizan para definir atajos para caracteres especiales, promoviendo así la modularidad, la claridad y la reutilización del código, facilitando así la creación y el mantenimiento de documentos XML coherentes y consistentes. Por ejemplo:

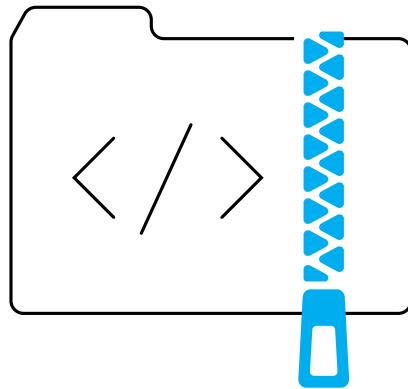
```
<!ENTITY nombre "valor">
```

Donde:

- nombre: es el nombre de la entidad.
- "valor": es el contenido por el que se sustituirá la entidad.

Ejemplo 1

Por ejemplo, en el siguiente documento xml y utilizando un dtd interno, definimos el atributo **codigo** y la entidad **copyright**:



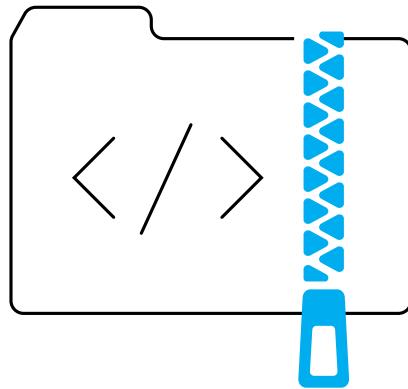
Custom code isn't available in PDF format

Guardando el código anterior en un documento xml y abriendo dicho documento en un navegador como Firefox, veríamos algo similar a esta imagen.

```
-<catalogo>
  -<articulo codigo="A-456.321-ASA">
    <nombre>Articulo1</nombre>
    <licencia>Copyright BirtLH.</licencia>
    <precio>13.25</precio>
    <stock>78</stock>
  </articulo>
  -<articulo>
    <nombre>Articulo2</nombre>
    <precioiva>22</precioiva>
    <stock>135</stock>
  </articulo>
</catalogo>
```

Elaboración propia

Ejemplo 2



Custom code isn't available in PDF format

```
▼<catalogo>
  ▼<films>
    <film codfilm="COD1">Los siete samuráis</film>
    <film codfilm="COD2">Pulp Fiction</film>
    <film codfilm="COD3">Rashomon</film>
    <film codfilm="COD4">Dersu Uzala</film>
  </films>
  ▼<directores>
    <director filmografia="COD2">Quentin Tarantino</director>
    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
  </directores>
</catalogo>
```

Elaboración propia

Guardando el código en un documento xml y abriendo dicho documento en un navegador como Chrome, veríamos algo similar a esta imagen.

Uso de herramientas

NotePad++

A la hora de validar un documento XML con un DTD podemos utilizar, por ejemplo, NotePad++.

Hay que preparar la herramienta, instalando el complemento XML Tools:

- En el menú "Complementos-Administrar complementos..", hay que seleccionar el complemento XML Tools para su instalación.
- En el menú "Complementos-XML Tools-Options", poner la opción "ProhibitDTD" a False.

Posteriormente, para validar un documento XML:

- Abrir el documento XML (con DTD interno ó referencia a DTD externo).
- En el menú "Complementos-XML Tools-Validate now".
 - Si el DTD es interno y todo es correcto, se mostrará un mensaje de "No error detected".
 - Si es externo, tratará de acceder a un esquema (xsd), se pulsa sobre "OK" y si todo es correcto, mostrará el mensaje de "No error detected".

The screenshot shows the Notepad++ interface with the file 'pruebadtd02.xml' open. The XML code defines a catalogo element with films and directores sub-elements, each having ID attributes. The 'Complementos' menu is open, and the 'XML Tools' submenu is selected. A context menu for 'Validate now' is displayed, listing options like 'Enable XML syntax auto-check', 'Validate now', and 'Pretty print'.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE catalogo [
  <!ELEMENT catalogo (films, directores)>
  <!ELEMENT films (film)*>
  <!ELEMENT film (#PCDATA)>
  <!ATTLIST film codfilm ID #REQUIRED>
  <!ELEMENT directores (director)*>
  <!ELEMENT director (#PCDATA)>
  <!ATTLIST director filmografia IDREFS #REQUIRED>
]>
<catalogo>
  <films>
    <film codfilm="COD1">Los siete samuráis</film>
    <film codfilm="COD2">Pulp Fiction</film>
    <film codfilm="COD3">Rashomon</film>
    <film codfilm="COD4">Dersu Uzala</film>
  </films>
  <directores>
    <director filmografia="COD2">Quentin Tarantino</director>
    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
  </directores>
</catalogo>
```

Elaboración propia

The screenshot shows the Notepad++ interface with the same XML file. A modal dialog box titled 'XML Tools plugin' displays the message 'No error detected.' with an 'Aceptar' button. The XML code is identical to the one in the first screenshot.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE catalogo [
  <!ELEMENT catalogo (films, directores)>
  <!ELEMENT films (film)*>
  <!ELEMENT film (#PCDATA)>
  <!ATTLIST film codfilm ID #REQUIRED>
  <!ELEMENT directores (director)*>
  <!ELEMENT director (#PCDATA)>
  <!ATTLIST director filmografia IDREFS #REQUIRED>
]>
<catalogo>
  <films>
    <film codfilm="COD1">Los siete samuráis</film>
    <film codfilm="COD2">Pulp Fiction</film>
    <film codfilm="COD3">Rashomon</film>
    <film codfilm="COD4">Dersu Uzala</film>
  </films>
  <directores>
    <director filmografia="COD2">Quentin Tarantino</director>
    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
  </directores>
</catalogo>
```

Elaboración propia

La herramienta Notepad++ ha ido cambiando y, como se puede observar en el vídeo, según la versión utilizada pueden variar los mensajes que muestra.

LMSGI04 XML validar con DTD



XML Copy Editor

También se puede validar un documento XML con un DTD utilizando XML Copy Editor. En este caso, para validar un documento XML:

- Abrir el documento XML (con DTD interno ó referencia a DTD externo).
- Pulsar sobre el ícono de "Validar".

prueba01.xml - XML Copy Editor

Fichero Editar Ver Insertar XML Herramientas Ayuda

prueba01.xml 1 2

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE catalogo [
3      <!ELEMENT catalogo (films, directores)>
4      <!ELEMENT films (film)*>
5      <!ELEMENT film (#PCDATA)>
6      <!ATTLIST film codfilm ID #REQUIRED>
7      <!ELEMENT directores (director)*>
8      <!ELEMENT director (#PCDATA)>
9      <!ATTLIST director filmografia IDREFS #REQUIRED>
10     >
11     <catalogo>
12         <films>
13             <film codfilm="COD1">Los siete samuráis</film>
14             <film codfilm="COD2">Pulp Fiction</film>
15             <film codfilm="COD3">Rashomon</film>
16             <film codfilm="COD4">Dersu Uzala</film>
17         </films>
18         <directores>
19             <director filmografia="COD2">Quentin Tarantino</director>
20             <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
21         </directores>
22     </catalogo>
```

Información 3

prueba01.xml is valid

Ln 16 Col 46

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE catalogo [
3      <!ELEMENT catalogo (films, directores)>
4      <!ELEMENT films (film)*>
5      <!ELEMENT film (#PCDATA)>
6      <!ATTLIST film codfilm ID #REQUIRED>
7      <!ELEMENT directores (director)*>
8      <!ELEMENT director (#PCDATA)>
9      <!ATTLIST director filmografia IDREFS #REQUIRED>
10     >
11    <catalogo>
12      <films>
13        <film codfilm="COD1">Los siete samuráis</film>
14        <film codfilm="COD2">Pulp Fiction</film>
15        <film codfilm="COD3">Rashomon</film>
16        <film codfilm="COD4">Dersu Uzala</film>
17      </films>
18      <directores>
19        <director filmografia="COD2">Quentin Tarantino</director>
20        <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
21      </directores>
22    </catalogo>
```

Información

prueba01.xml is valid

Comprobar bien-formado

Se utiliza para saber si la sintaxis del documento es correcto. Es decir, si está **bien formado**.

The screenshot shows the XML Copy Editor application window. The title bar reads "prueba01.xml - XML Copy Editor". The menu bar includes "Fichero", "Editar", "Ver", "Insertar", "XML", "Herramientas", and "Ayuda". Below the menu is a toolbar with icons for file operations and validation. A status bar at the bottom right shows "Ln 16 Col 46".

The main pane displays the XML code:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE catalogo [
3      <!ELEMENT catalogo (films, directores)>
4      <!ELEMENT films (film)*>
5      <!ELEMENT film (#PCDATA)>
6      <!ATTLIST film codfilm ID #REQUIRED>
7      <!ELEMENT directores (director)*>
8      <!ELEMENT director (#PCDATA)>
9      <!ATTLIST director filmografia IDREFS #REQUIRED>
10     >
11    <catalogo>
12      <films>
13        <film codfilm="COD1">Los siete samuráis</film>
14        <film codfilm="COD2">Pulp Fiction</film>
15        <film codfilm="COD3">Rashomon</film>
16        <film codfilm="COD4">Dersu Uzala</film>
17      </films>
18      <directores>
19        <director filmografia="COD2">Quentin Tarantino</director>
20        <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
21      </directores>
22    </catalogo>
```

An orange circle with the number "2" is drawn around the "catalogo" element in the XML code.

A small "Information" window is open below the main editor, containing the message "prueba01.xml is valid".

Validar

Sirve para validar el documento xml.

The screenshot shows the XML Copy Editor interface with the file "prueba01.xml" open. The XML code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE catalogo [
  <ELEMENT catalogo (films, directores)>
  <ELEMENT films (film)*>
  <ELEMENT film (#PCDATA)>
  <!ATTLIST film codfilm ID #REQUIRED>
  <ELEMENT directores (director)*>
  <ELEMENT director (#PCDATA)>
  <!ATTLIST director filmografia IDREFS #REQUIRED>
]>
<catalogo>
  <films>
    <film codfilm="COD1">Los siete samuráis</film>
    <film codfilm="COD2">Pulp Fiction</film>
    <film codfilm="COD3">Rashomon</film>
    <film codfilm="COD4">Dersu Uzala</film>
  </films>
  <directores>
    <director filmografia="COD2">Quentin Tarantino</director>
    <director filmografia="COD1 COD3 COD4">Akira Kurosawa</director>
  </directores>
</catalogo>
```

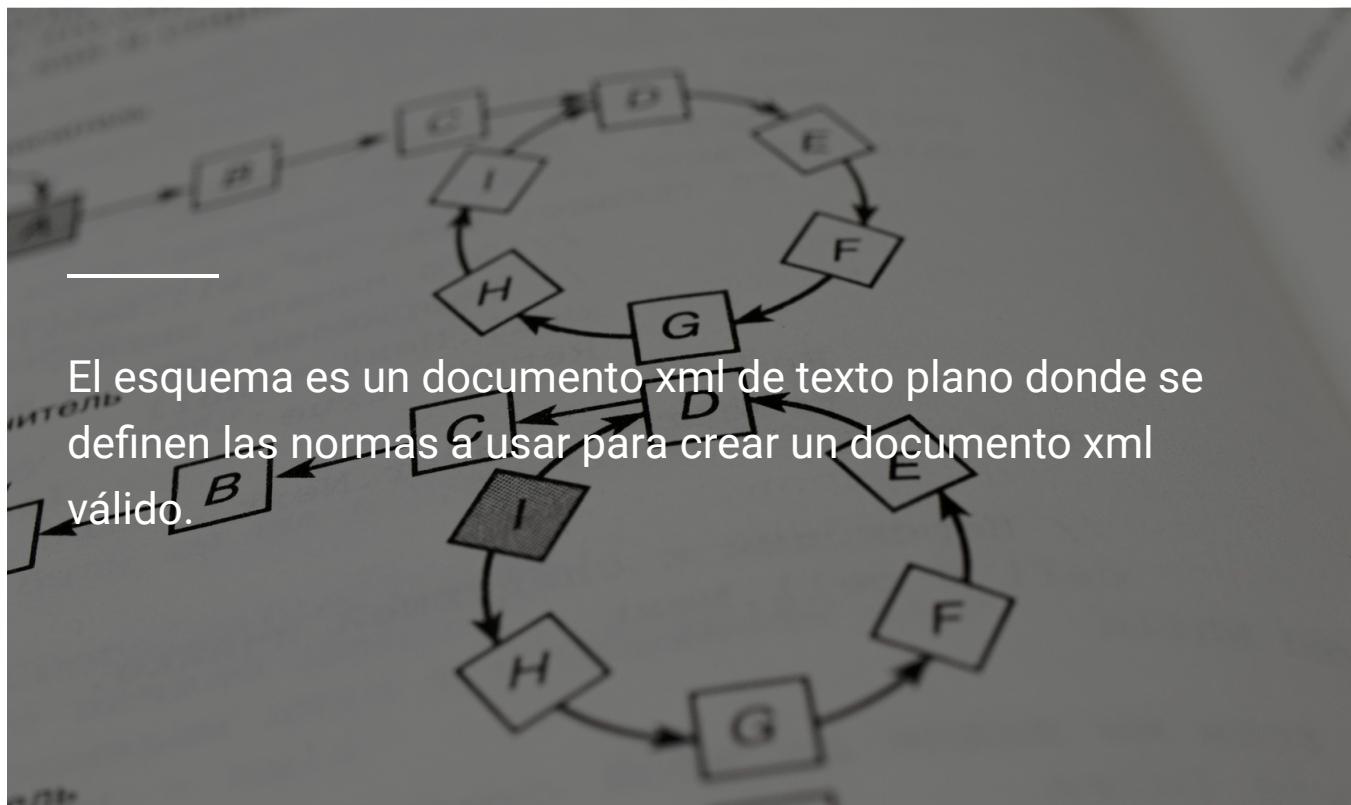
The status bar at the bottom right indicates "Ln 16 Col 46". A tooltip window titled "Información" is visible, stating "prueba01.xml is valid" with a count of 3 errors.

Mensajes

En esta zona se muestran los mensajes de los resultados de las operaciones. En este caso, se ha pulsado sobre "Validar" y el documento xml es válido respecto al DTD interno definido.

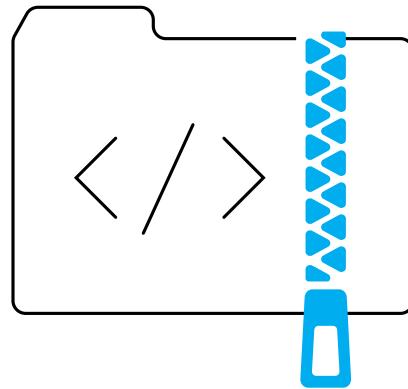
Elaboración propia

Validación con esquema

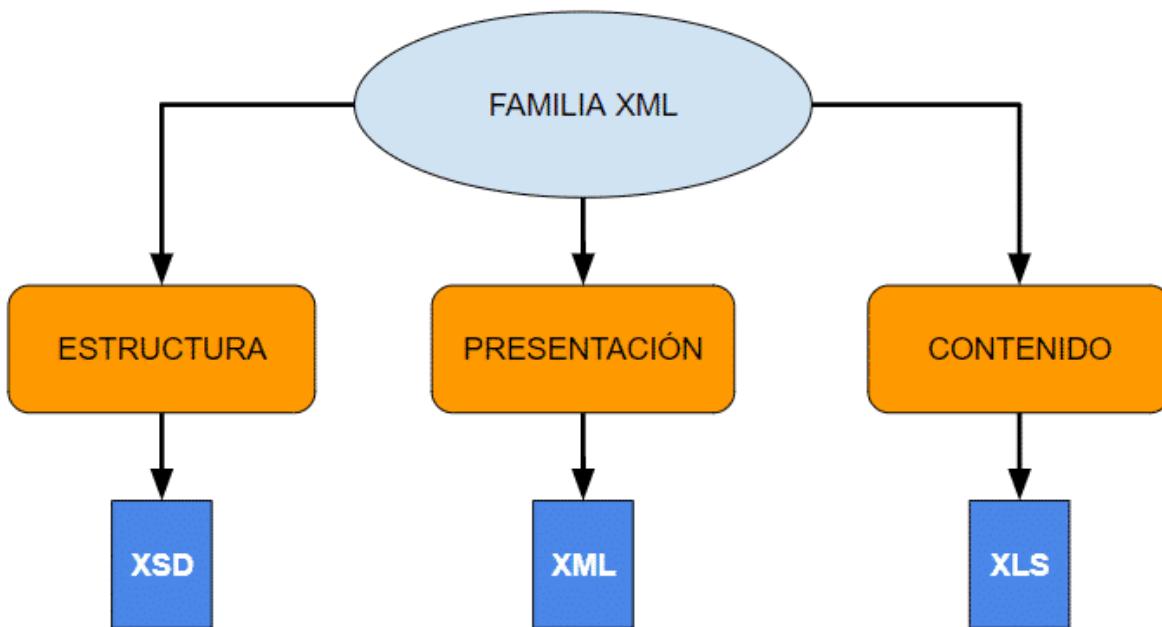


Introducción

Los esquemas XML son archivos en texto plano con extensión ".xsd". XSD (XML Schema Definition) es un lenguaje, normalmente conocido como **XML Schema que es utilizado para definir la estructura de un documento XML y así permitir su validación.**

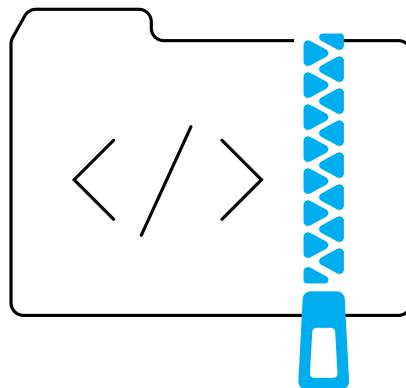


Custom code isn't available in PDF format



Asociar XSD con XML

La asociación de un documento XSD a un documento XML se realiza mediante un espacio de nombres con una serie de atributos. Por ejemplo:



Custom code isn't available in PDF format

Donde:

- En el documento XML:
 - El atributo **xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"**, define el espacio de nombres donde están definidos los elementos.
 - Con **xmlns:noNamespaceSchemaLocation="MiEsquema.xsd"**, se define dónde está definido el esquema que sigue el documento xml.

- En el esquema XSD en `<xs:schema`
- `xmlns:xs="http://www.w3.org/2001/XMLSchema"`, se distinguen:
- El elemento `<xs:schema>` que es la raíz de todo documento xsd.
 - El atributo `xmlns:xs="http://www.w3.org/2001/XMLSchema"`, especifica el espacio de nombres de donde provienen los elementos y tipos usados. El alias puede ser "xs" ó cualquier otro, pero debe ser el mismo en todo el documento porque hacen referencia a las etiquetas del espacio de nombres.
 - Se pueden utilizar los atributos "`elementFormDefault`" y "`attributeFormDefault`", que indican si es obligatorio ó no el uso del espacio de nombres delante de los nombres de elementos ó atributos según los valores posibles "qualified" y "unqualified". Por defecto, se usa la segunda opción, pero es mejor utilizar la primera, sobre todo con los elementos, que evita muchos errores.
-

Componentes de un XSD

Al igual que en los DTD, este lenguaje permite trabajar con datos simples y con estructuras de datos complejas, compuestos por el anidamiento de otros datos simples o compuestos.

Indicadores de ocurrencias y facetas

Para indicar las veces que un elemento debe aparecer ó repetirse se pueden usar los atributos:

- **minOccurs**: define el número mínimo de veces que puede aparecer el elemento, cuyo valor por defecto es 1. Con "unbounded" es ilimitado.
- **maxOccurs**: define el número máximo de veces que puede aparecer el elemento, cuyo valor por defecto es 1. Con "unbounded" es ilimitado.

¿Y qué restricciones podemos aplicar sobre los valores de los datos de un elemento o atributo? Están definidos por las facetas, que solo pueden aplicarse sobre tipos simples utilizando el elemento `<xs:restriction>`. Se expresan como un elemento dentro de una restricción y se pueden combinar para lograr restringir más el valor del elemento. Son, entre otros:

Longitud

Para definir la longitud del tipo de datos: length, minlength, maxlength.

Por ejemplo:

Descripción	Código
Para indicar que un código tendrá 6 caracteres.	<pre><xs:simpleType name="TipoLongitud"> <xs:restriction base="xs:string"> <xs:length value="6"/> </xs:restriction> </xs:simpleType></pre>

Conjunto de valores

Para restringir los valores a un determinado conjunto: enumeration.

Por ejemplo:

Descripción	Código
Para limitar los colores que un elemento ó atributo pueda tomar.	<pre><xs:simpleType name="TipoColor"> <xs:restriction base="xs:string"> <xs:enumeration value="Azul"/> <xs:enumeration value="Rojo"/> <xs:enumeration value="Verde"/> </xs:restriction> </xs:simpleType></pre>

Espacios

Para definir el tratamiento de espacios en blanco, tabulaciones, saltos de línea:whiteSpace. Donde:

- preserve: se mantienen.
- replace: para sustituirlas por espacios en blanco.
- collapse: después de reemplazarlas por espacios en blanco, eliminar todos los espacios en blanco únicos y sustituir varios espacios en blanco seguidos por un único espacio en blanco.

Por ejemplo:

Descripción	Código
Para eliminar espacios de una dirección:	<pre><xs:simpleType name="TipoDireccion"> <xs:restriction base="xs:string"> <xs:whiteSpace value="collapse"/> </xs:restriction> </xs:simpleType></pre>

Patrones

—

Para construir máscaras que han de cumplir los datos de un elemento: pattern.

Patrón	Descripción
[A-Z a-z]	Letra.
[A-Z]	Letra mayúscula.
[a-z]	Letra minúscula.
[0-9]	Dígitos decimales.
\D	Cualquier carácter excepto un dígito del 0 al 9.

Patrón	Descripción
(A)	Cadena que coincide con A.
A B	Cadena que es igual a la cadena A o a la B.
AB	Cadena que es la concatenación de las cadenas A y B.
A?	Cero o una vez la cadena A.
A⁺	Una o más veces la cadena A.
A[*]	Cero o más veces la cadena A.
[abcd]	Alguno de los caracteres que están entre corchetes.
[^abcd]	Cualquier carácter que no esté entre corchetes.
\t	Tabulador.

Por ejemplo:

Descripción	Código
Para indicar que una contraseña tendrá 8 caracteres utilizando letras minúsculas y mayúsculas y números.	<pre><xs:element name="password"> <xs:simpleType> <xs:restriction> base="xs:string"> <xs:pattern value="[a-zA-Z0-9]{8}" /> </xs:restriction> </xs:simpleType></pre>

Descripción	Código
	</xs:element>

Rangos

Para definir los límites superiores/inferiores del tipo de datos: minExclusive, minInclusive, maxExclusive, maxInclusive.

Cuando son Inclusive el valor que se determine es parte del conjunto de valores válidos para el dato, mientras que cuando se utiliza Exclusive, el valor dado no pertenece al conjunto de valores válidos.

Por ejemplo:

Descripción	Código
Para indicar que un descuento estará entre 10 y 30.	<pre data-bbox="866 952 1372 1199"><xs:simpleType name="TipoDescuento"> <xs:restriction base="xs:integer"> <xs:minInclusive value="10"/> <xs:maxInclusive value="30"/> </xs:restriction> </xs:simpleType></pre>

Dígitos

Para definir el número de dígitos totales y decimales de un número: totalDigits, fractionDigits.

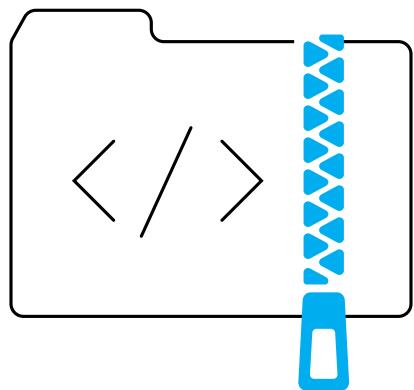
Por ejemplo:

Descripción	Código
Por ejemplo, para indicar que un número tendrá 3 dígitos.	<pre data-bbox="953 1744 1356 1945"><xs:simpleType name="TipoDigitos"> <xs:restriction base="xs:integer"> <xs:totalDigits value="3"/></pre>

Descripción	Código
	</xs:restriction> </xs:simpleType>

Elementos simples XSD

Son los elementos sin descendientes ni atributos conteniendo solamente valores de un determinado tipo.



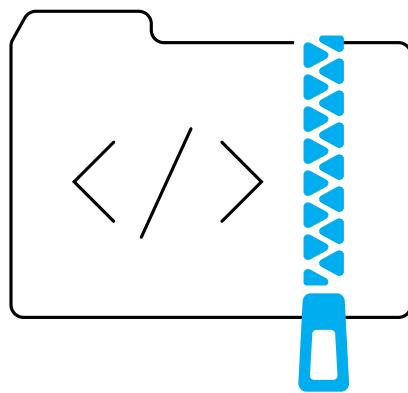
Custom code isn't available in PDF format

Por ejemplo, para definir un elemento XSD llamado color de tipo cadena de texto y valor por defecto "verde":

```
<xsd:element name="color" type="xsd:string" default="verde"/>
```

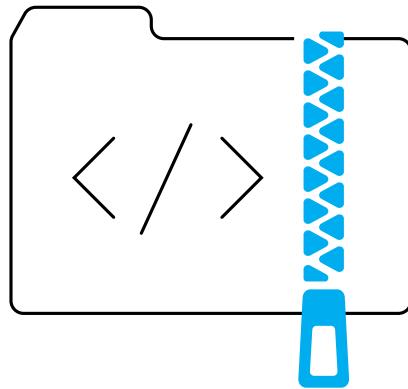
Elementos complejos XSD

Estos elementos contienen otros elementos y atributos.



Custom code isn't available in PDF format

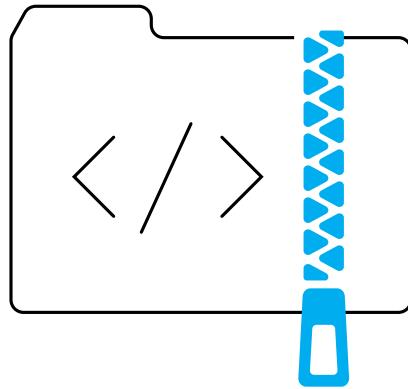
Por ejemplo:



Custom code isn't available in PDF format

Tipos de datos XSD

En XML Schema hay muchos tipos de datos predefinidos para los elementos o atributos, siendo los más comunes:



Custom code isn't available in PDF format

Basados en estos, se pueden crear tipos de datos personalizados:

Tipos simples

Los elementos de este tipo son elementos sin atributos que sólo contienen datos.

Ejemplo	Código
Para definir un elemento de nombre "edad" que pueda tomar valores entre 0 y 100, se puede hacer de la forma:	<pre><xs:element name="edad"> <xs:simpleType> <xs:restriction base="xs:integer"> <xs:minInclusive value="0"/> <xs:maxInclusive value="100"/> </xs:restriction> </xs:simpleType> </xs:element></pre>

Tipos complejos

Los elementos de este tipo son elementos que pueden tener atributos, no tener contenido ó contener otros elementos.

Ejemplo	Código
Para definir un mensaje con diferentes partes.	<pre><xs:element name="mensaje"> <xs:complexType> <xs:sequence> <xs:element name="para" type="xs:string"/> <xs:element name="de" type="xs:string"/> <xs:element name="asunto" type="xs:string"/> <xs:element name="contenido" type="xs:string"/> </xs:sequence> </xs:complexType> </xs:element></pre>

En un tipo complejo de XSD, la opción `mixed="true"` indica que el elemento puede contener tanto texto como otros elementos hijos. Si no ponemos `mixed="true"`, solo se permitirán los elementos definidos, sin texto libre intercalado.

```
<xs:complexType name="Nota"
mixed="true">
<xs:sequence>
<xs:element name="autor" type="xs:string"
minOccurs="0"/>
</xs:sequence>
<xs:attribute name="fecha" type="xs:date"
use="required"/>
</xs:complexType>
```

Para definir un tipo complejo de nombre "InfoCompleta", que deriva de otro tipo complejo "Info", al que se añaden: dirección, ciudad y país:

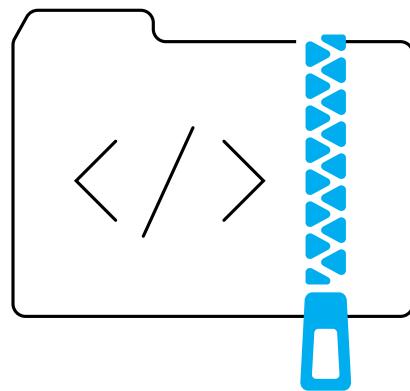
```
<xs:element name="empleado"
type="InfoCompleta"/>

<xs:complexType name="Info">
<xs:sequence>
<xs:element name="Nombre"
type="xs:string"/>
<xs:element name="Apellido"
type="xs:string"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="InfoCompleta">
<xs:complexContent>
<xs:extension base="Info">
<xs:sequence>
<xs:element name="Direccion"
type="xs:string"/>
<xs:element name="Ciudad"
type="xs:string"/>
<xs:element name="Pais" type="xs:string"/>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

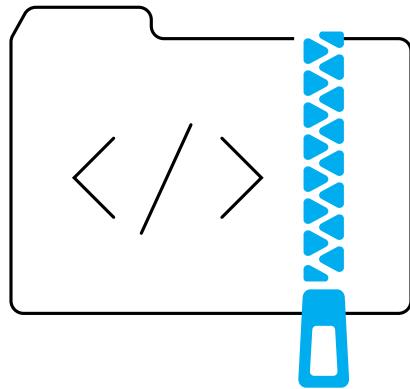
Atributos en XSD

En XML Schema los atributos poseen la misma funcionalidad que los atributos de un elemento DTD. Hay que tener en cuenta que los elementos con atributos se consideran elementos complejos. Los atributos pueden tomar por valor tipos simples y existe la posibilidad de añadir restricciones a sus tipos.



Custom code isn't available in PDF format

Por ejemplo:



Custom code isn't available in PDF format

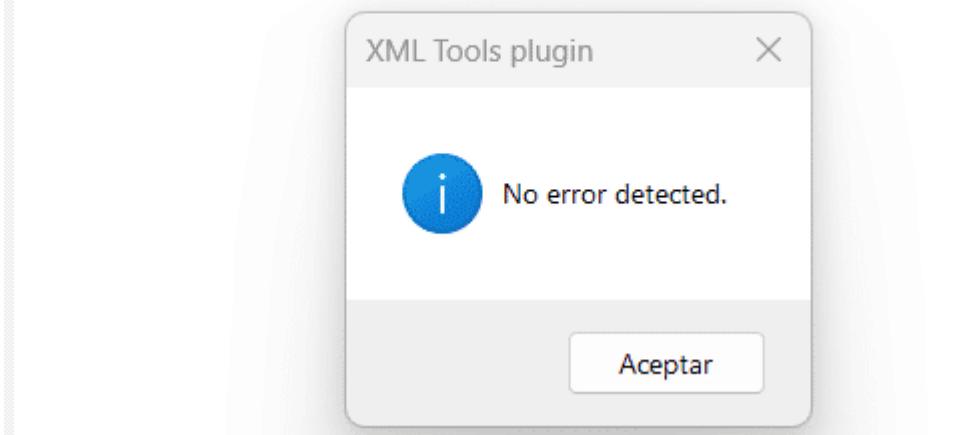
Uso de herramientas

Notepad++

A la hora de validar un documento XML con un esquema podemos utilizar, por ejemplo, Notepad++. Hay que preparar la herramienta, instalando el complemento XML Tools:

- En el menú "Complementos-Administrar complementos..", hay que seleccionar el complemento XML Tools para su instalación.

```
<?xml version="1.0" encoding="utf-8"?>
<nombre dni="12345678A">Xabier</nombre>
```



Elaboración propia

Para validar un documento XML con un esquema XSD:

- Abrir el documento XML.
- En el menú "**Complementos-XML Tools-Validate now**". Si en el documento XML hay referencia al esquema se procederá a validar el documento y si no hay referencia da la posibilidad de buscar el esquema y se procederá a validar. Si todo es correcto, mostrará el mensaje de "No error detected".

La herramienta Notepad++ ha ido cambiando y, como se puede observar en el vídeo, según la versión utilizada pueden variar los mensajes que muestra.

LMSGI04 XML validar con XSD



XML Copy Editor

Para validar un documento XML con un esquema XSD:

- Abrir el documento XML.
- Si en el documento XML hay referencia al esquema se procederá a validar el documento y si no hay referencia habrá que buscar el esquema ("XML-Asociar-XML Schema..."), añade de forma temporal su ubicación y se procederá a validar. Si todo es correcto, mostrará el mensaje de ".xml is valid".

The screenshot shows the XML Copy Editor interface. In the main editor area, there are two lines of XML code:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <nombre dni="12345678A" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

Below the editor, a message box displays the validation result:

información ×

prueba03.xml is valid

Documentación del esquema

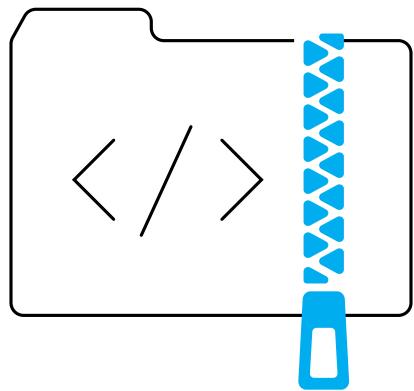
Una vez que hemos visto como crear un esquema vamos a ver el modo de incorporar cierta documentación (quién es el autor, limitaciones de derechos de autor, utilidad del esquema, etc.) al mismo.

Podemos pensar que un método para añadir esta información es utilizar comentarios. El problema es que los analizadores no garantizan que los comentarios no se modifiquen al procesar los documentos y por tanto, que los datos añadidos no se pierdan en algún proceso de transformación del documento.

En lugar de usar los comentarios, XML Schema tiene definido el elemento **xs:annotation** que permite guardar información adicional sobre el documento. Este elemento a su vez puede contener:

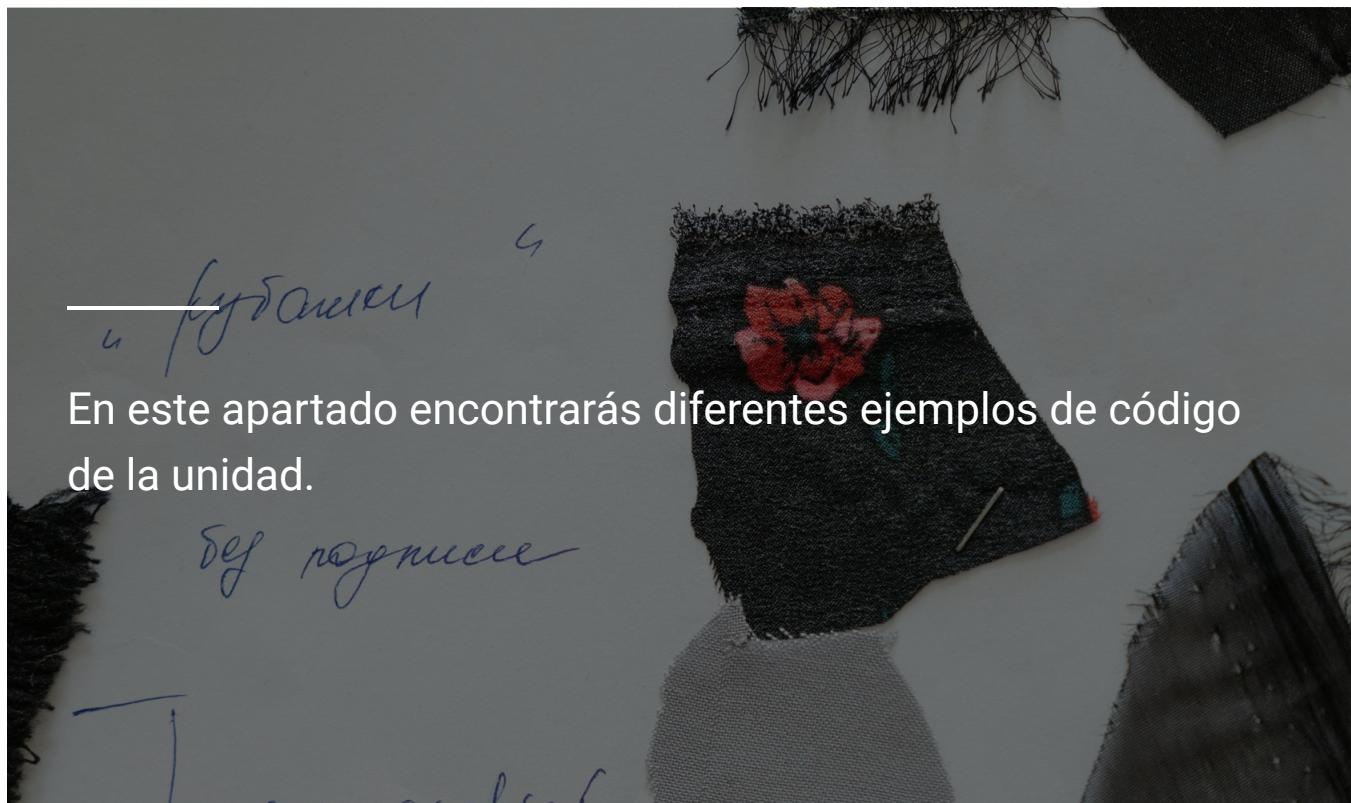
- **xs:documentation**, para añadir comentarios al esquema y tiene dos atributos opcionales: "source" y "xml:lang".
- **xs:appinfo**, se utiliza para guardar información sobre la aplicación y tiene un atributo opcional llamado "source".

Por ejemplo, el siguiente esquema xsd:



Custom code isn't available in PDF format

Ejemplos



En este apartado encontrarás diferentes ejemplos de código de la unidad.

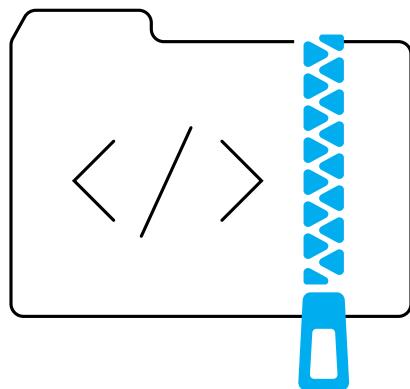
Introducción

Aquí se muestran diferentes documentos para su comprobación y ver ejemplos de validación de documentos XML con un DTD y con un esquema.

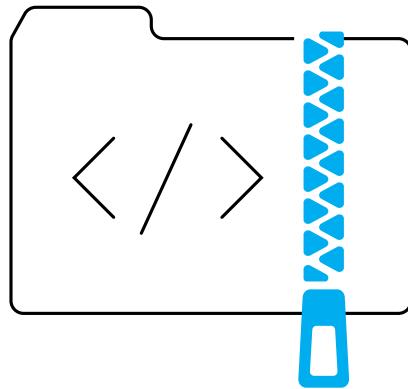
Ejemplos de documentos bien formados

Ejemplo1

Trata de encontrar los errores del siguiente documento XML.



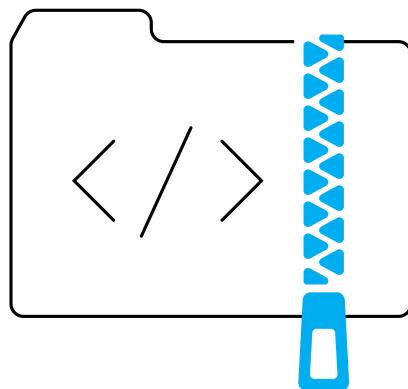
Custom code isn't available in PDF format



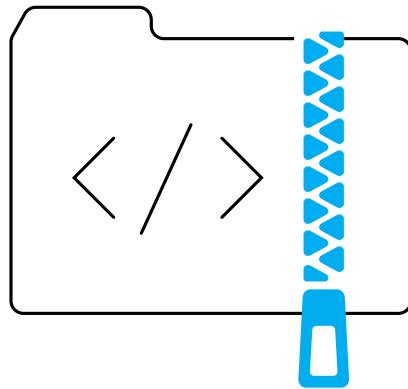
Custom code isn't available in PDF format

Ejemplo2

Trata de encontrar los errores del siguiente documento XML.



Custom code isn't available in PDF format

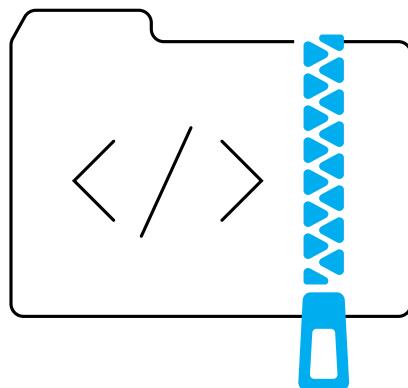


Custom code isn't available in PDF format

Ejemplos de validación con DTD

Ejemplo3

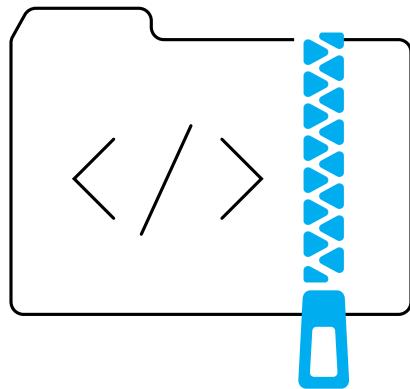
Comprueba si es válido el siguiente documento XML según el DTD.



Custom code isn't available in PDF format

Ejemplo4

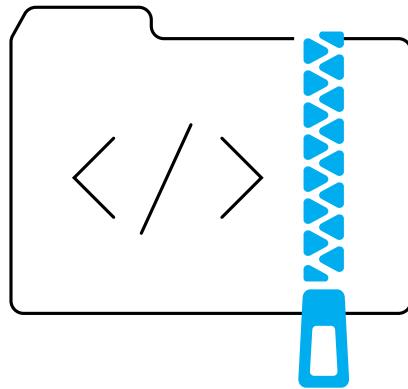
Comprueba si es válido el siguiente documento XML según el DTD.



Custom code isn't available in PDF format

Ejemplo5

Comprueba si es válido el siguiente documento XML según el DTD.



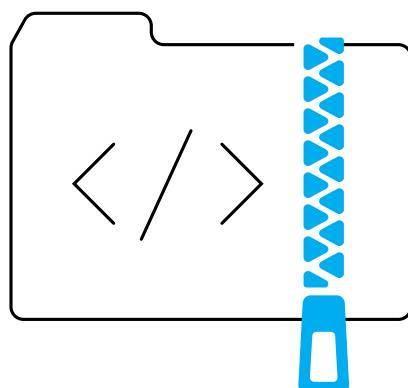
Custom code isn't available in PDF format

Ejemplos de validación con XSD

Trata de encontrar los errores de los siguientes documentos XML.

Ejemplo6

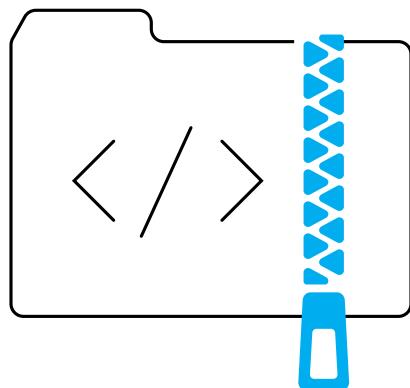
Comprueba si es válido el siguiente documento XML según el XML Schema.



Custom code isn't available in PDF format

Ejemplo7

Comprueba si es válido el siguiente documento XML según el XML Schema.



Custom code isn't available in PDF format

Autoevaluación

Pregunta

01/03

La validación de un documento XML:

- Es el proceso de verificar si el documento cumple con las reglas y estructuras especificadas en un esquema ó DTD (Document Type Definition).
- Puede haber documentos XML válidos que estén mal formados.
- No aporta nada a una organización y es conveniente evitar su uso.
- Implica que el documento XML esté bien formado.

Pregunta

02/03

La validación de un documento XML:

- Solamente se puede realizar por medio de DTD (Document Type Definition).
- Se puede realizar por medio de DTD ó un esquema.
- Se puede realizar por medio de DTD.
- Solamente se puede realizar por medio de un esquema.

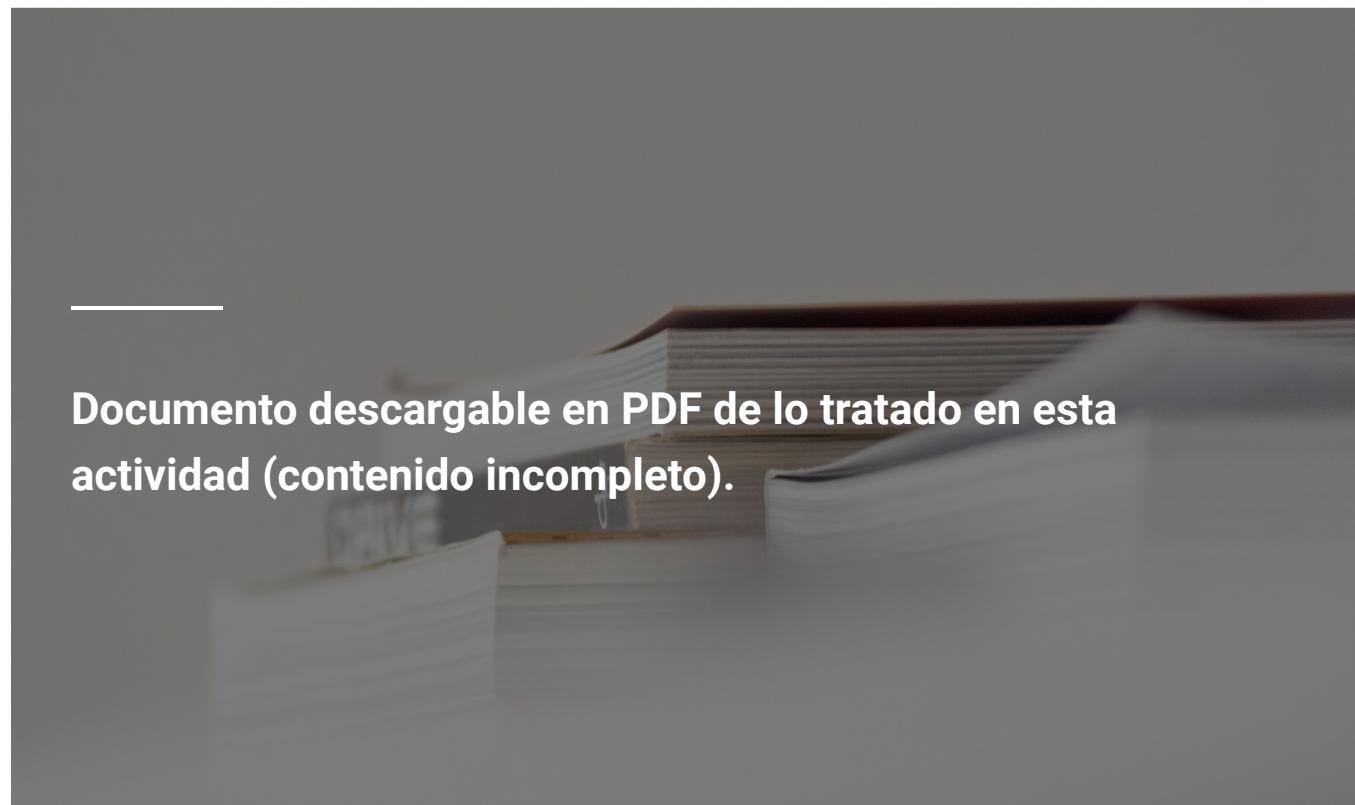
Pregunta

03/03

La validación de un documento XML:

- Es preferible realizarlo con un DTD porque es más específico.
- Es preferible realizarlo con un esquema porque es más específico.
- Se puede realizar sin un documento que lo valide.

Contenido PDF



El contenido del siguiente documento PDF está incompleto al no ofrecer los enlaces que aparecen en los apuntes ni las funciones de los elementos interactivos de la actividad. Su uso ha de ser por lo tanto, completado por dichos apuntes.

Puedes visualizar y descargar el contenido PDF a través de este enlace

IR AL ENLACE