Arquitectura Orientada a Servicios

Introducción a XML

Edwin Salvador

20 de abril de 2015

Sesión 3

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Introducción a XML

- eXtensible Markup Language

Introducción a XML

- eXtensible Markup Language
- Similar a HTML (markup)

Introducción a XML

- eXtensible Markup Language
- Similar a HTML (markup)
- Diseñado para describir datos, no para presentarlos.

Introducción a XML

- eXtensible Markup Language
- Similar a HTML (markup)
- Diseñado para describir datos, no para presentarlos.
- Las etiquetas no están predefinidas. Se debe definir las necesarias.

Introducción a XML

- eXtensible Markup Language
- Similar a HTML (markup)
- Diseñado para describir datos, no para presentarlos.
- Las etiquetas no están predefinidas. Se debe definir las necesarias.
- auto-descriptivo

Introducción a XML

- eXtensible Markup Language
- Similar a HTML (markup)
- Diseñado para describir datos, no para presentarlos.
- Las etiquetas no están predefinidas. Se debe definir las necesarias.
- auto-descriptivo
- Recomendado por la W3C

XML vs HTML

- XML no reemplaza a HTML

XML vs HTML

- XML no reemplaza a HTML
- Diseñados con objetivos diferentes

XML vs HTML

- XML no reemplaza a HTML
- Diseñados con objetivos diferentes
  - XML: describir *¿qué son los datos?*

XML vs HTML

- XML no reemplaza a HTML
- Diseñados con objetivos diferentes
  - XML: describir *¿qué son los datos?*
  - HTML: presentar *¿Cómo se ven los datos?*

XML vs HTML

- XML no reemplaza a HTML
- Diseñados con objetivos diferentes
  - XML: describir *¿qué son los datos?*
  - HTML: presentar *¿Cómo se ven los datos?*
- HTML presenta la información, XML contiene la información.

XML **NO HACE NADA**

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- La nota tiene emisor, receptor, encabezado y cuerpo.

XML **NO HACE NADA**

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- La nota tiene emisor, receptor, encabezado y cuerpo.
- Pero no hace nada solamente tiene información dentro de etiquetas.

XML **NO HACE NADA**

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- La nota tiene emisor, receptor, encabezado y cuerpo.
- Pero no hace nada solamente tiene información dentro de etiquetas.
- Es necesario escribir un programa que lo envíe, reciba o lo presente.

Tu defines tus etiquetas

- Las etiquetas <to> o <from> no están definidas en ningún estándar.

Tu defines tus etiquetas

- Las etiquetas <to> o <from> no están definidas en ningún estándar.
- Se puede "inventar" las propias etiquetas.

Tu defines tus etiquetas

- Las etiquetas <to> o <from> no están definidas en ningún estándar.
- Se puede "inventar" las propias etiquetas.
- No existen etiquetas predefinidas a diferencia de HTML.

XML no reemplaza a HTML

- XML complementa a HTML no lo reemplaza.

XML no reemplaza a HTML

- XML complementa a HTML no lo reemplaza.
- XML es una herramienta independiente de software y hardware para transmitir información

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Uso de XML

- Se lo utiliza mucho en el desarrolla web para simplificar el almacenamiento de datos y el compartir.

Uso de XML

- Se lo utiliza mucho en el desarrolla web para simplificar el almacenamiento de datos y el compartir.
- XML separa los datos del HTML o cualquier otro lenguaje.

Uso de XML

- Se lo utiliza mucho en el desarrolla web para simplificar el almacenamiento de datos y el compartir.
- XML separa los datos del HTML o cualquier otro lenguaje.
- Se puede guardar los datos de una página web para evitar cambiar el HTML o CSS cuando existe un cambio en los datos.

Uso de XML

- Se lo utiliza mucho en el desarrolla web para simplificar el almacenamiento de datos y el compartir.
- XML separa los datos del HTML o cualquier otro lenguaje.
- Se puede guardar los datos de una página web para evitar cambiar el HTML o CSS cuando existe un cambio en los datos.
- Se puede acceder al documento XML a través de JS para actualizar los datos en la página.

XML simplifica el compartir de datos

- Un problema general en los sistemas de computación y bases de datos es la incompatibilidad de los formatos de datos.

XML simplifica el compartir de datos

- Un problema general en los sistemas de computación y bases de datos es la incompatibilidad de los formatos de datos.
- Con XML se puede almacenar información en un formato plano que proporciona independencia de software y hardware.

XML simplifica el compartir de datos

- Un problema general en los sistemas de computación y bases de datos es la incompatibilidad de los formatos de datos.
- Con XML se puede almacenar información en un formato plano que proporciona independencia de software y hardware.
- Facilita el compartir datos entre diferentes aplicaciones, lo cuál es perfecto para SOA.

XML simplifica el compartir de datos

- Un problema general en los sistemas de computación y bases de datos es la incompatibilidad de los formatos de datos.
- Con XML se puede almacenar información en un formato plano que proporciona independencia de software y hardware.
- Facilita el compartir datos entre diferentes aplicaciones, lo cuál es perfecto para SOA.
- Esto evita perder el tiempo (y a veces datos) en convertir datos de un formato a otro por incompatibilidad.

XML simplifica el compartir de datos

- Un problema general en los sistemas de computación y bases de datos es la incompatibilidad de los formatos de datos.
- Con XML se puede almacenar información en un formato plano que proporciona independencia de software y hardware.
- Facilita el compartir datos entre diferentes aplicaciones, lo cuál es perfecto para SOA.
- Esto evita perder el tiempo (y a veces datos) en convertir datos de un formato a otro por incompatibilidad.
- XML puede ser utilizado en diferentes aplicaciones, sistemas, sistemas operativos, navegadores, etc sin pérdida de datos.

Lenguajes de Internet escritos en XML

- XHTML

Lenguajes de Internet escritos en XML

- XHTML
- XML Schema (utilizado en WS)

Lenguajes de Internet escritos en XML

- XHTML
- XML Schema (utilizado en WS)
- SVG

Lenguajes de Internet escritos en XML

- XHTML
- XML Schema (utilizado en WS)
- SVG
- WSDL (Web Service Definition Language)

Lenguajes de Internet escritos en XML

- XHTML
- XML Schema (utilizado en WS)
- SVG
- WSDL (Web Service Definition Language)
- RSS

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Árbol XML

Los documentos XML tienen una estructura de árbol (Raíz, ramas, hojas).

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **1:** declaración XML. Define versión (1.0)

Árbol XML

Los documentos XML tienen una estructura de árbol (Raíz, ramas, hojas).

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **1:** declaración XML. Define versión (1.0)
- **2:** <note> raíz (indica que el documento es una "nota").

Árbol XML

Los documentos XML tienen una estructura de árbol (Raíz, ramas, hojas).

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **1:** declaración XML. Define versión (1.0)
- **2:** <note> raíz (indica que el documento es una "nota").
- **3, 4, 5, 6:** elementos hijos.

Árbol XML

Los documentos XML tienen una estructura de árbol (Raíz, ramas, hojas).

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **1:** declaración XML. Define versión (1.0)
- **2:** <note> raíz (indica que el documento es una "nota").
- **3, 4, 5, 6:** elementos hijos.
- **7:** fin

Árbol XML

- Los documentos XML **deben** tener un elemento raíz.

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

Árbol XML

- Los documentos XML **deben** tener un elemento raíz.

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

- Padres, hijos y hermanos son las posibles relaciones entre elementos.

Árbol XML

- Los documentos XML **deben** tener un elemento raíz.

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

- Padres, hijos y hermanos son las posibles relaciones entre elementos.
- Todos los elementos pueden contener texto y atributos (como en HTML)

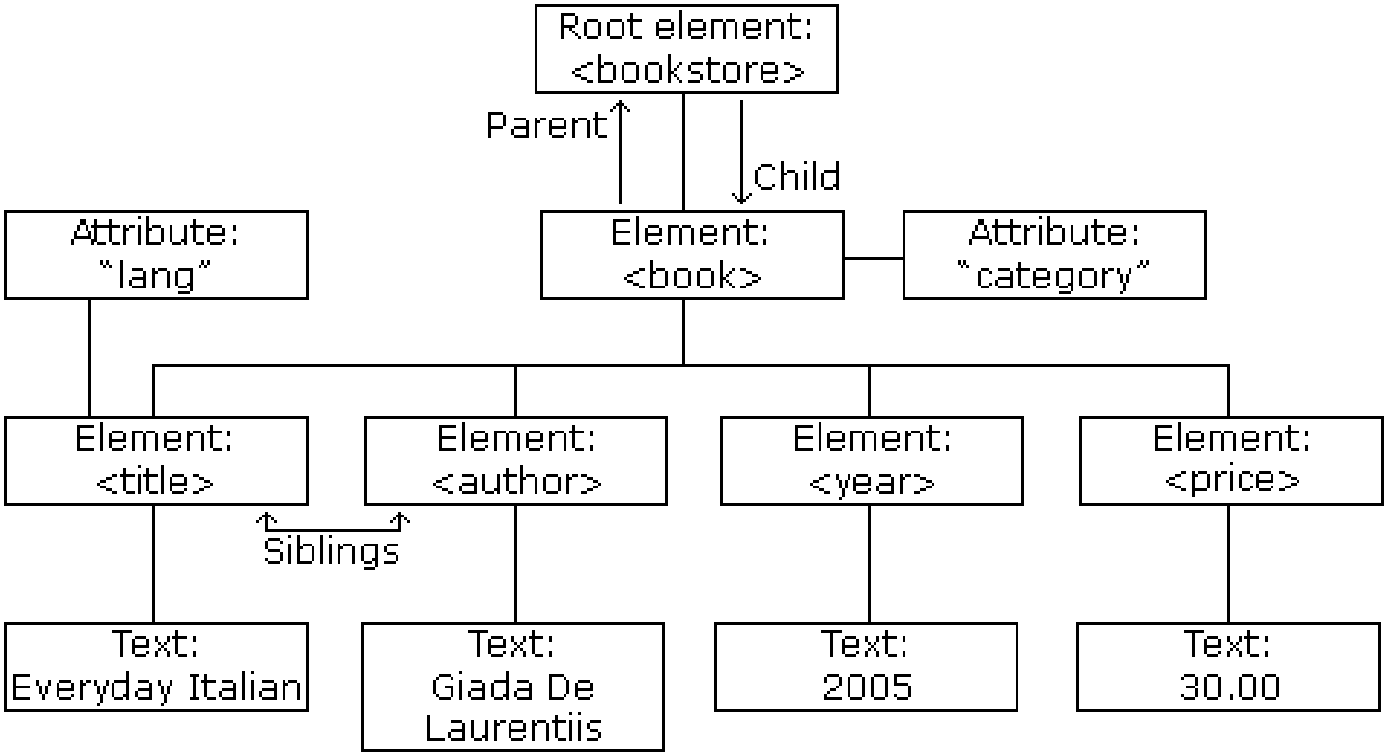
Ejemplo

Árbol XML

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Ejemplo

Árbol XML



Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. <Letter> es diferente de <letter>.
- Elementos XML se deben anidar correctamente.

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)
- Se debe tener **obligatoriamente** un elemento raíz.

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)
- Se debe tener **obligatoriamente** un elemento raíz.
- Los valores de los atributos deben ir entre comillas (“”)

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)
- Se debe tener **obligatoriamente** un elemento raíz.
- Los valores de los atributos deben ir entre comillas (“”)

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)
- Se debe tener **obligatoriamente** un elemento raíz.
- Los valores de los atributos deben ir entre comillas (“”)

Incorrecto

```
<note date=12/11/2007>
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Reglas de Sintaxis XML

- La sintaxis XML es simple y fácil de aprender.
- La declaración XML (`<?xml version="1.0" encoding="UTF-8"?>`) no tiene una etiqueta de cierre porque no es parte del documento XML.
- las etiquetas XML son *case sensitive*. `<Letter>` es diferente de `<letter>`.
- Elementos XML se deben anidar correctamente.
  - `<b><i>This text is bold and italic</b></i>` (incorrecto)
  - `<b><i>This text is bold and italic</i></b>` (correcto)
- Se debe tener **obligatoriamente** un elemento raíz.
- Los valores de los atributos deben ir entre comillas (“”)

Incorrecto

```
<note date=12/11/2007>
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Correcto

```
<note date="12/11/2007">
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Caracteres especiales en XML

- Algunos caracteres son reservados para XML.

Caracteres especiales en XML

- Algunos caracteres son reservados para XML.
- Si incluimos el signo $<$ dentro de un elemento XML nos generará un error.

```
<message>if salary <1000 then</message>
```

Caracteres especiales en XML

- Algunos caracteres son reservados para XML.
- Si incluimos el signo $<$ dentro de un elemento XML nos generará un error.

```
<message>if salary <1000 then</message>
```

- Debemos escribirlo así:

```
<message>if salary &lt; 1000 then</message>
```

Caracteres especiales en XML

- Algunos caracteres son reservados para XML.
- Si incluimos el signo $<$ dentro de un elemento XML nos generará un error.

```
<message>if salary <1000 then</message>
```

- Debemos escribirlo así:

```
<message>if salary &lt; 1000 then</message>
```

- Hay 5 caracteres especiales definidos en XML:

&lt;	<	menor que
&gt;	>	mayor que
&amp;	&	ampersand
&apos;	’	apostrofe
&quot;	"	comillas

Comentarios

El mismo que en HTML

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

Espacios en blanco

- A diferencia de HTML los espacios en blanco no son truncados.

Espacios en blanco

- A diferencia de HTML los espacios en blanco no son truncados.
- XML: `Hola  como estás?`

Espacios en blanco

- A diferencia de HTML los espacios en blanco no son truncados.
- XML: `Hola  como estás?`
- HTML: `Hola como estás?`

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.
- Puede contener:

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.
- Puede contener:
  - otros elementos

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.
- Puede contener:
  - otros elementos
  - texto

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.
- Puede contener:
  - otros elementos
  - texto
  - atributos

Elementos XML

- Un elemento se considera desde la etiqueta de apertura hasta la de cierre.
- Puede contener:
  - otros elementos
  - texto
  - atributos
  - una combinación de lo anterior

Elementos XML

¿Cuáles son los elementos en el ejemplo?

```
<bookstore>
  <book category="CHILDREN">
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Elementos vacíos

- Un elemento sin contenido se escribe así:

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

Elementos vacíos

- Un elemento sin contenido se escribe así:

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

- O así:

```
<element />
```

Elementos vacíos

- Un elemento sin contenido se escribe así:

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

- O así:

```
<element />
```

- Ambos producen el mismo efecto.

Elementos vacíos

- Un elemento sin contenido se escribe así:

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

- O así:

```
<element />
```

- Ambos producen el mismo efecto.
- Un elemento vacío puede también tener atributos.

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.
- Deben empezar con una letra o guión bajo (_)

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.
- Deben empezar con una letra o guión bajo (_)
- No pueden empezar con las letras xml, XML, Xml, etc.

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.
- Deben empezar con una letra o guión bajo (_)
- No pueden empezar con las letras xml, XML, Xml, etc.
- Pueden contener letras, dígitos, guiones, guiones bajos y puntos.

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.
- Deben empezar con una letra o guión bajo (_)
- No pueden empezar con las letras xml, XML, Xml, etc.
- Pueden contener letras, dígitos, guiones, guiones bajos y puntos.
- No pueden contener espacios.

Reglas de nombres en XML

- Los nombres de los elementos son case-sensitive.
- Deben empezar con una letra o guión bajo (_)
- No pueden empezar con las letras xml, XML, Xml, etc.
- Pueden contener letras, dígitos, guiones, guiones bajos y puntos.
- No pueden contener espacios.
- El único nombre reservado es xml. Se puede usar cualquier otros nombre.

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).
- Deben ser cortos y simples: <titulo>. No complejos: <el_titulo_del_libro>.

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).
- Deben ser cortos y simples: <titulo>. No complejos: <el_titulo_del_libro>.
- Evitar guiones (-). Puede causar que algunos programas al leer el XML crean que se intenta restar. first-name.

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).
- Deben ser cortos y simples: <titulo>. No complejos: <el_titulo_del_libro>.
- Evitar guiones (-). Puede causar que algunos programas al leer el XML crean que se intenta restar. first-name.
- Evitar el punto (.). Puede causar que algunos programas al leer el XML crean que se trata de una propiedad. first.name.

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).
- Deben ser cortos y simples: <titulo>. No complejos: <el_titulo_del_libro>.
- Evitar guiones (-). Puede causar que algunos programas al leer el XML crean que se intenta restar. first-name.
- Evitar el punto (.). Puede causar que algunos programas al leer el XML crean que se trata de una propiedad. first.name.
- Evitar dos puntos (:). Reservados para espacio de nombres (namespaces)

Nombres: mejores prácticas

- Deben ser descriptivos (<persona>, <nombre>, apellido).
- Deben ser cortos y simples: <titulo>. No complejos: <el_titulo_del_libro>.
- Evitar guiones (-). Puede causar que algunos programas al leer el XML crean que se intenta restar. first-name.
- Evitar el punto (.). Puede causar que algunos programas al leer el XML crean que se trata de una propiedad. first.name.
- Evitar dos puntos (:). Reservados para espacio de nombres (namespaces)
- Evitar acentos (áéñ). Dificulta la lectura para algunos programas.

Estilo de nombres

Se debe elegir un estilo para nombrar los elementos y mantenerlos en todo el documento.

Estilo	**Ejemplo**	**Descripción**
Minúsculas	<firstname>	Todo en minúsculas
Mayúsculas	<FIRSTNAME>	Todo en mayúsculas
Guión bajo	<first_name	guión bajo separa palabras
Pascal	<FirstName>	Mayúscula primera letra de palabra
Camel case	<firstName>	Mayúscula primera letra de palabra excepto primera

Elementos extendibles

- Se pueden extender los elementos para llevar más información

Elementos extendibles

- Se pueden extender los elementos para llevar más información
- Supongamos que tenemos:

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <body>Don't forget me this weekend!</body>
</note>
```

Elementos extendibles

- Se pueden extender los elementos para llevar más información
- Supongamos que tenemos:

```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <body>Don't forget me this weekend!</body>
</note>
```

- Una aplicación extrae <to>, <from> y <body> y produce la salida:
  MENSAJE
  To: Tove
  From: Jani
  Don't forget me this weekend!

Elementos extendibles

- El autor de documento XML necesita aumentar más información y obtiene:

```
<note>
  <date>2008-01-10</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Elementos extendibles

- El autor de documento XML necesita aumentar más información y obtiene:

```
<note>
  <date>2008-01-10</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- ¿Qué pasara con la aplicación?

Elementos extendibles

- El autor de documento XML necesita aumentar más información y obtiene:

```
<note>
  <date>2008-01-10</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- ¿Qué pasara con la aplicación? Nada. XML es extendible y no debería haber problemas con esto.

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Atributos

- Los elementos XML pueden tener atributos (como en HTML)

Atributos

- Los elementos XML pueden tener atributos (como en HTML)
- Proporcionan información adicional sobre el elemento.

Atributos

- Los elementos XML pueden tener atributos (como en HTML)
- Proporcionan información adicional sobre el elemento.
- Esta información puede ser irrelevante a los datos pero puede ser útil para los programas que lean el XML.

```
<file type="gif">computer.gif</file>
```

Atributos

- Los elementos XML pueden tener atributos (como en HTML)
- Proporcionan información adicional sobre el elemento.
- Esta información puede ser irrelevante a los datos pero puede ser útil para los programas que lean el XML.

```
<file type="gif">computer.gif</file>
```

- Deben estar entre comillas (“”) o (‘’)

Atributos

- Los elementos XML pueden tener atributos (como en HTML)
- Proporcionan información adicional sobre el elemento.
- Esta información puede ser irrelevante a los datos pero puede ser útil para los programas que lean el XML.

```
<file type="gif">computer.gif</file>
```

- Deben estar entre comillas (“”) o (‘’)
- Si el valor del atributo contiene comillas (“”) se puede utilizar (‘’)

```
<gangster name='George "Shotgun" Ziegler'>
```

o

```
<gangster name="George &quot;Shotgun&quot; Ziegler">
```

Elementos vs Atributos

```
<person gender="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>

<person>
  <gender>female</gender>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

- Se puede utilizar atributos o elementos con el mismo efecto.

Elementos vs Atributos

```
<person gender="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>

<person>
  <gender>female</gender>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

- Se puede utilizar atributos o elementos con el mismo efecto.
- Se recomienda utilizar elementos para evitar confusión.

Otro ejemplo

Elementos vs Atributos

```
<note date="2008-01-10">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

<note>
  <date>2008-01-10</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Una manera más clara

Elementos vs Atributos

```
<note>
  <date>
    <year>2008</year>
    <month>01</month>
    <day>10</day>
  </date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Elementos vs Atributos

- Atributos no pueden contener múltiples valores

Elementos vs Atributos

- Atributos no pueden contener múltiples valores
- Atributos no pueden contener estructura de árbol

Elementos vs Atributos

- Atributos no pueden contener múltiples valores
- Atributos no pueden contener estructura de árbol
- Atributos no son expandible (cambio futuros)

Elementos vs Atributos

- Atributos no pueden contener múltiples valores
- Atributos no pueden contener estructura de árbol
- Atributos no son expandible (cambio futuros)
- Atributos son difíciles de mantener y leer.

```
<note day="10" month="01" year="2008"
to="Tove" from="Jani" heading="Reminder"
body="Don't forget me this weekend!">
</note>
```

Usar atributos para metadata

Atributos como ID para facilitar la búsqueda de elementos:

```
<messages>
  <note id="501">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>
  <note id="502">
    <to>Jani</to>
    <from>Tove</from>
    <heading>Re: Reminder</heading>
    <body>I will not</body>
  </note>
</messages>
```

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.
- Este problema se puede dar cuando se intenta mezclar documentos XML de diferentes aplicaciones.

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.
- Este problema se puede dar cuando se intenta mezclar documentos XML de diferentes aplicaciones.

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.
- Este problema se puede dar cuando se intenta mezclar documentos XML de diferentes aplicaciones.

Tabla HTML

```
<table>
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.
- Este problema se puede dar cuando se intenta mezclar documentos XML de diferentes aplicaciones.

Tabla HTML

```
<table>
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Mueble

```
<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

Espacios de nombres (namespaces)

- Evitan el conflicto entre nombres dentro del XML.
- Este problema se puede dar cuando se intenta mezclar documentos XML de diferentes aplicaciones.

Tabla HTML

```
<table>
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Mueble

```
<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

- Esto causará un problema porque se refieren a cosas diferentes pero no hay forma de manejar esta diferencia.

Namespace

- Los conflictos se pueden resolver utilizando un prefijo

Namespace

- Los conflictos se pueden resolver utilizando un prefijo
- Se puede incluir ambos elementos en un solo documento XML.

```
<h:table>
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

El atributo xmlns

Namespace

- Cuando se utilizan prefijos en XML se debe definir un namespace a través del atributo **xmlns** en la etiqueta de apertura.

El atributo xmlns

Namespace

- Cuando se utilizan prefijos en XML se debe definir un namespace a través del atributo **xmlns** en la etiqueta de apertura.
- Sintaxis: `xmlns:`*`prefix`*`= "URI"`

El atributo xmlns

Namespace

- Cuando se utilizan prefijos en XML se debe definir un namespace a través del atributo **xmlns** en la etiqueta de apertura.
- Sintaxis: `xmlns:`*`prefix`*`= "URI"`
- Todos los hijos con el mismo prefijo son asociados con el mismo namespace.

El atributo xmlns

Namespace

- Cuando se utilizan prefijos en XML se debe definir un namespace a través del atributo **xmlns** en la etiqueta de apertura.
- Sintaxis: `xmlns:`*`prefix`*`= "URI"`
- Todos los hijos con el mismo prefijo son asociados con el mismo namespace.
- El `URI` solo se utiliza para darle un único nombre pero algunas empresas pueden utilizar el `URI` para brindar más información sobre el namespace.

Ejemplo

El atributo xmlns

```
<root>
<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table xmlns:f="http://www.w3schools.com/furniture">
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
</root>
```

El atributo xmlns en la raíz

```
<root xmlns:h="http://www.w3.org/TR/html4/"
      xmlns:f="http://www.w3schools.com/furniture">
<h:table>
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
</root>
```

Namespace por defecto

- Se pueden declarar namespaces por defecto lo que nos evita usar los prefijos:

```
<table xmlns="http://www.w3.org/TR/html4/">
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>

<table xmlns="http://www.w3schools.com/furniture">
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Codificación XML

- Los documentos XML pueden contener caracteres especiales como acentos: áéíóúñ.

Codificación XML

- Los documentos XML pueden contener caracteres especiales como acentos: áéíóúñ.
- Para evitar errores se debe especificar la codificación utilizada o guardar el XML como UTF-8.

El Consorcio Unicode

- Desarrolla el estándar Unicode.

El Consorcio Unicode

- Desarrolla el estándar Unicode.
- Reemplaza los caracteres con su Formato de Transformación Unicode (UTF) *Unicode Transformation Format*.

El Consorcio Unicode

- Desarrolla el estándar Unicode.
- Reemplaza los caracteres con su Formato de Transformación Unicode (UTF) *Unicode Transformation Format*.
- Es un éxito y es implementado en HTML, XML, Java, JS, PHP, email, etc. Muchos SO y navegadores.

El Consorcio Unicode

- Desarrolla el estándar Unicode.
- Reemplaza los caracteres con su Formato de Transformación Unicode (UTF) *Unicode Transformation Format*.
- Es un éxito y es implementado en HTML, XML, Java, JS, PHP, email, etc. Muchos SO y navegadores.
- Los sets de carácteres más utilizados son UTF-8 y UTF-16.

El Consorcio Unicode

- Desarrolla el estándar Unicode.
- Reemplaza los caracteres con su Formato de Transformación Unicode (UTF) *Unicode Transformation Format*.
- Es un éxito y es implementado en HTML, XML, Java, JS, PHP, email, etc. Muchos SO y navegadores.
- Los sets de carácteres más utilizados son UTF-8 y UTF-16.
- **UTF-8** utiliza 1 byte (8 bits) para representar los caracteres latinos básicos y 2, 3 o 4 bytes para el resto. Es el estándar para Web (HTML, CSS, JS, PHP, SQL y XML).

El Consorcio Unicode

- Desarrolla el estándar Unicode.
- Reemplaza los caracteres con su Formato de Transformación Unicode (UTF) *Unicode Transformation Format*.
- Es un éxito y es implementado en HTML, XML, Java, JS, PHP, email, etc. Muchos SO y navegadores.
- Los sets de carácteres más utilizados son UTF-8 y UTF-16.
- **UTF-8** utiliza 1 byte (8 bits) para representar los caracteres latinos básicos y 2, 3 o 4 bytes para el resto. Es el estándar para Web (HTML, CSS, JS, PHP, SQL y XML).
- **UTF-16** utiliza 2 bytes (16 bits) para la mayoría de caracteres y 4 bytes para el resto.

El prólogo

- Es la primera línea de un documento XML.

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

El prólogo

- Es la primera línea de un documento XML.

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

- Es opcional. Si no se incluye, el defecto es UTF-8.

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

Errores XML

- Un documento XML es creado por una computadora, subido a otra computadora (servidor), y presentado en una tercera computadora (el cliente navegador).

Errores XML

- Un documento XML es creado por una computadora, subido a otra computadora (servidor), y presentado en una tercera computadora (el cliente navegador).
- Si la codificación no es interpretada correctamente, es probable que el navegador presente información errónea y se presente un mensaje de error.

Errores XML

- Un documento XML es creado por una computadora, subido a otra computadora (servidor), y presentado en una tercera computadora (el cliente navegador).
- Si la codificación no es interpretada correctamente, es probable que el navegador presente información errónea y se presente un mensaje de error.
- Para asegurar la calidad del documento se debe utilizar UTF-8 el cual cubre caracteres internacionales.

Recomendaciones de codificación

- Utilizar un editor de texto de soporte codificación.

Recomendaciones de codificación

- Utilizar un editor de texto de soporte codificación.
- Asegurarse de la codificación que utiliza en editor.

Recomendaciones de codificación

- Utilizar un editor de texto de soporte codificación.
- Asegurarse de la codificación que utiliza en editor.
- Especificar la codificación en la cabecera.

Recomendaciones de codificación

- Utilizar un editor de texto de soporte codificación.
- Asegurarse de la codificación que utiliza en editor.
- Especificar la codificación en la cabecera.
- UTF-8 es la codificación más segura.

Recomendaciones de codificación

- Utilizar un editor de texto de soporte codificación.
- Asegurarse de la codificación que utiliza en editor.
- Especificar la codificación en la cabecera.
- UTF-8 es la codificación más segura.
- UTF-8 es el estándar web.

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Visualizando documentos XML

- Los documentos XML se pueden visualizar en la mayoría de navegadores.

Visualizando documentos XML

- Los documentos XML se pueden visualizar en la mayoría de navegadores.
- No será desplegados como páginas HTML.

Visualizando documentos XML

- Los documentos XML se pueden visualizar en la mayoría de navegadores.
- No será desplegados como páginas HTML.
- http://www.w3schools.com/xml/note.xml

Visualizando documentos XML

- Los documentos XML se pueden visualizar en la mayoría de navegadores.
- No será desplegados como páginas HTML.
- http://www.w3schools.com/xml/note.xml
- Se presentan con colores, los signos (+) y (-) se pueden hacer clic para expandir o contraer (depende del visualizador del navegador).

Visualizando documentos XML

- Los documentos XML se pueden visualizar en la mayoría de navegadores.
- No será desplegados como páginas HTML.
- http://www.w3schools.com/xml/note.xml
- Se presentan con colores, los signos (+) y (-) se pueden hacer clic para expandir o contraer (depende del visualizador del navegador).
- Recordemos que los documentos XML no contienen información de como presentar los datos.

Visualizando un XML inválido

- La mayoría de visualizadores presentarán un error y otros lo presentarán incorrectamente.

Visualizando un XML inválido

- La mayoría de visualizadores presentarán un error y otros lo presentarán incorrectamente.
- http://www.w3schools.com/xml/note_error.xml

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".
- Un documento XML "*válido*" debe apegarse también a cierta definición de tipo de documento.

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".
- Un documento XML "*válido*" debe apegarse también a cierta definición de tipo de documento.
- Las reglas que definen los elementos y atributos legales y en un documento XML se llaman Definición de Tipo de Documento (DTD) *Document Type Definitions* o XML Schemas.

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".
- Un documento XML "*válido*" debe apegarse también a cierta definición de tipo de documento.
- Las reglas que definen los elementos y atributos legales y en un documento XML se llaman Definición de Tipo de Documento (DTD) *Document Type Definitions* o XML Schemas.
- **DTD** el documento original de definición del tipo de documento

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".
- Un documento XML "*válido*" debe apegarse también a cierta definición de tipo de documento.
- Las reglas que definen los elementos y atributos legales y en un documento XML se llaman Definición de Tipo de Documento (DTD) *Document Type Definitions* o XML Schemas.
- **DTD** el documento original de definición del tipo de documento
- **XML Schema** Una alternativa a DTD.

XML Doctypes

- Un documento XML "*Bien formado*" quiere decir que cumple con las reglas de **sintaxis** que hemos visto.
- Existen herramientas que nos ayudan a verificar si la sintaxis de nuestro documento es correcta (http://www.w3schools.com/xml/xml_validator.asp)
- Un documento XML "bien formado" no es lo mismo que uno "válido".
- Un documento XML "*válido*" debe apegarse también a cierta definición de tipo de documento.
- Las reglas que definen los elementos y atributos legales y en un documento XML se llaman Definición de Tipo de Documento (DTD) *Document Type Definitions* o XML Schemas.
- **DTD** el documento original de definición del tipo de documento
- **XML Schema** Una alternativa a DTD.
- Sirven para establecer un estándar para intercambiar datos y verificar que los datos recibidos son válidos.

Cuando no usar DTD o Schema

- No es necesario utilizar DTD siempre. Por lo general en documentos pequeños no es necesario.

Cuando no usar DTD o Schema

- No es necesario utilizar DTD siempre. Por lo general en documentos pequeños no es necesario.
- Al desarrollar aplicaciones es conveniente esperar a que la especificación interna de nuestro XML sea estable antes de empezar a incluir un DTD.

Errores en XML

- Los errores en XML pueden hacer que nuestra aplicación deje de funcionar.

Errores en XML

- Los errores en XML pueden hacer que nuestra aplicación deje de funcionar.
- La especificación de XML de W3C indica que un programa debe dejar de procesar un XML cuando este encuentra un error.

Errores en XML

- Los errores en XML pueden hacer que nuestra aplicación deje de funcionar.
- La especificación de XML de W3C indica que un programa debe dejar de procesar un XML cuando este encuentra un error.
- Esto permite que las aplicaciones sean pequeñas, rápidas y compatibles.

Errores en XML

- Los errores en XML pueden hacer que nuestra aplicación deje de funcionar.
- La especificación de XML de W3C indica que un programa debe dejar de procesar un XML cuando este encuentra un error.
- Esto permite que las aplicaciones sean pequeñas, rápidas y compatibles.
- A diferencia de XML, los documentos HTML si pueden ser presentados a pesar de contener errores.

Errores en XML

- Los errores en XML pueden hacer que nuestra aplicación deje de funcionar.
- La especificación de XML de W3C indica que un programa debe dejar de procesar un XML cuando este encuentra un error.
- Esto permite que las aplicaciones sean pequeñas, rápidas y compatibles.
- A diferencia de XML, los documentos HTML si pueden ser presentados a pesar de contener errores.
- Los errores no son permitidos en XML

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

XML DTD

- Un documento XML “válido” es un documento XML “Bien formado” que cumple con las reglas de un DTD.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

XML DTD

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

El DTD es interpretado así:

- **!DOCTYPE note:** define node como el elemento raíz.
- **!ELEMENT note:** define que el elemento node debe contener 4 elementos: to, from, heading, body.
- **!ELEMENT to:** define el elemento to como "#PCDATA" (parse-able text data).
- **!ELEMENT from:** · · ·

DTD para declarar entidades

- Se puede utilizar un DTD para definir caracteres especiales que se utilizarán en el documento:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
  <!ENTITY nbsp " ">
  <!ENTITY writer "Writer: Donald Duck.">
  <!ENTITY copyright "Copyright: W3Schools.">
]>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
  <footer>&writer; &copyright;</footer>
</note>
```

DTD para declarar entidades

- Se puede utilizar un DTD para definir caracteres especiales que se utilizarán en el documento:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
  <!ENTITY nbsp " ">
  <!ENTITY writer "Writer: Donald Duck.">
  <!ENTITY copyright "Copyright: W3Schools.">
]>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
  <footer>&writer; &copyright;</footer>
</note>
```

- Una entidad tiene 3 partes: el ampersand (&), nombre y punto y coma (;).

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

XML Schema

- Un Schema de describe la estructura de un documento XML, similar al DTD.

XML Schema

- Un Schema de describe la estructura de un documento XML, similar al DTD.
- Un XML validado contra un Schema es “válido” y “bien formado”.

XML Schema

- Un Schema de describe la estructura de un documento XML, similar al DTD.
- Un XML validado contra un Schema es “válido” y “bien formado”.
- Es una alternativa al DTD

Ejemplo XML Schema

```
<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

- `<xs:element name="note">` define el elemento "note".
- `<xs:complexType>` "note" es un tipo complejo.
- `<xs:sequence>` El tipo complejo es una secuencia de elementos.
- `<xs:element name="to" type="xs:string">` el elemento "to" es de tipo string (texto).
- `<xs:element name="from" type="xs:string">` el elemento "from" es de tipo string.
- `<xs:element name="heading" type="xs:string">` el elemento "heading" es de tipo string.
- `<xs:element name="body" type="xs:string">` el elemento "body" es de tipo string.

XML Schemas más poderosos que DTD

- Escritos en XML

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento
  - Fácil definir restricciones a los datos

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento
  - Fácil definir restricciones a los datos
  - Fácil validar que los datos estén correctos

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento
  - Fácil definir restricciones a los datos
  - Fácil validar que los datos estén correctos
  - Fácil convertir datos a otros tipos de datos

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento
  - Fácil definir restricciones a los datos
  - Fácil validar que los datos estén correctos
  - Fácil convertir datos a otros tipos de datos
- Soportan namespaces.

XML Schemas más poderosos que DTD

- Escritos en XML
- Extensibles
- Soportan tipos de datos
  - Fácil describir el contenido del documento
  - Fácil definir restricciones a los datos
  - Fácil validar que los datos estén correctos
  - Fácil convertir datos a otros tipos de datos
- Soportan namespaces.
- Schemas utilizan sintaxis XML

title

Contenido I

1. Introducción a XML
2. Uso de XML
3. Árbol XML
4. Reglas de Sintaxis XML
5. Elementos XML
6. Atributos
7. Espacios de nombres (namespaces)
8. Codificación XML
9. Visualizando documentos XML
10. Validación de XML
    - XML Doctypes
    - XML DTD
    - XML Schema
11. Ejercicios

Ejercicio 1

- Escribir un documento XML para representar cada uno de los siguientes casos:

Ejercicio 1

- Escribir un documento XML para representar cada uno de los siguientes casos:
  - Una colección de CDs

Ejercicio 1

- Escribir un documento XML para representar cada uno de los siguientes casos:
  - Una colección de CDs
  - Un catálogo de plantas de una tienda

Ejercicio 1

- Escribir un documento XML para representar cada uno de los siguientes casos:
  - Una colección de CDs
  - Un catálogo de plantas de una tienda
  - Un menú de un restaurante

Ejercicio 2

- Escribir un documento XML que contenga la declaración de una tabla HTML y abrir el documento en el navegador.

Ejercicio 2

- Escribir un documento XML que contenga la declaración de una tabla HTML y abrir el documento en el navegador.
  - ¿Cómo presenta el navegador el documento XML? ¿Muestra la tabla HTML o muestra el árbol XML? ¿Por qué?

Ejercicio 3

- El siguiente documento XML contiene errores, modificar el documento de tal manera que se pueda visualizar correctamente en el navegador.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:template match="/">
<html>
<body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr>
      <th style="text-align:left">Title</th>
      <th style="text-align:left">Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
    </xsl:for-each>
  </table>
</body>
</html>
</xsl:template>
```

Ejercicio 4

- Modificar el documento de la pregunta 3 de tal manera que se puedan omitir los prefijos "`xsl`" de las etiquetas.

Ejercicio 5

- Investigar que maneras existen para dar formato un documento XML de manera que se pueda visualizar en el navegador de manera similar a un HTML. ¿Qué técnica es la más recomendada según la W3C?

Ejercicio 5

- Investigar que maneras existen para dar formato un documento XML de manera que se pueda visualizar en el navegador de manera similar a un HTML. ¿Qué técnica es la más recomendada según la W3C?
- Elegir una de las técnicas y aplicarla a uno de los documentos del ejercicio 1.

Ejercicio 6

- Escribir un DTD que declare 3 entidades: su nombre, el nombre de la materia (Arquitectura Orientada a Servicios) y el nombre de la Universidad (EPN). Incluya estas entidades en los ejercicios 1 y compruebe su funcionamiento.