



# Introducción a XML

---

JUAN CARLOS CONDE RAMÍREZ

*WEB-TECHNOLOGIES*

# Objetivos

---

- Comprender qué es XML.
- Conocer los antecedentes que hicieron posible la creación de XML.
- Conocer las ventajas del uso de estándares y tecnologías basados en XML.
- Conocer la estructura base de un documento XML para después entender su funcionamiento.
- Visualizar la aplicación de estos conceptos mediante la muestra de ejemplos simples.

# Introducción a XML I

---

- XML significa lenguaje de marcas generalizado (*eXtensible Markup Language*).
- Es un lenguaje usado para estructurar información en un documento, por ejemplo, archivos de configuración de un programa o una tabla de datos.
- Es un estándar abierto y libre, creado por el Consorcio World Wide Web, W3C (creadores de la WWW), en colaboración con un panel que incluye representantes de las principales compañías productoras de software.

# Introducción a XML II

---

- XML fue propuesto en 1996, y la primera especificación apareció en 1998.
- Su uso ha tenido un crecimiento acelerado; hoy en día parece que ya todo el mundo está usando, o quiere usar, XML.

# Antecedentes de XML I

---

- Antes de ser lanzado el XML, ya existían otros lenguajes de marcas, como por ejemplo el HTML, basados en el lenguaje generalizado de marcas (SGML).
- El problema con el SGML es que por ser muy flexible y muy general, complica el análisis sintáctico de un documento y la especificación de la estructura (incluido en un DTD).
- XML es más exigente que SGML en la sintaxis, lo cual resulta en documentos mejor estructurados y por lo tanto más fácil la construcción de librerías para procesarlo.

# Antecedentes de XML II

---

- Por ejemplo, en LaTeX existen también “marcas” que permiten estructurar un documento como:
  - Identificar el *nombre del autor* y el *título del documento* – los comandos `\author` y `\title`
- Sin embargo no existe forma de obligar a los autores de documentos a que usen estas marcas y algunos de ellos pueden introducir el título de forma que aparezca visualmente igual a lo que se obtiene cuando se usa `\author` y `\maketitle`
- Esto conlleva a problemas cuando queremos extraer de forma automática el título de varios documentos.

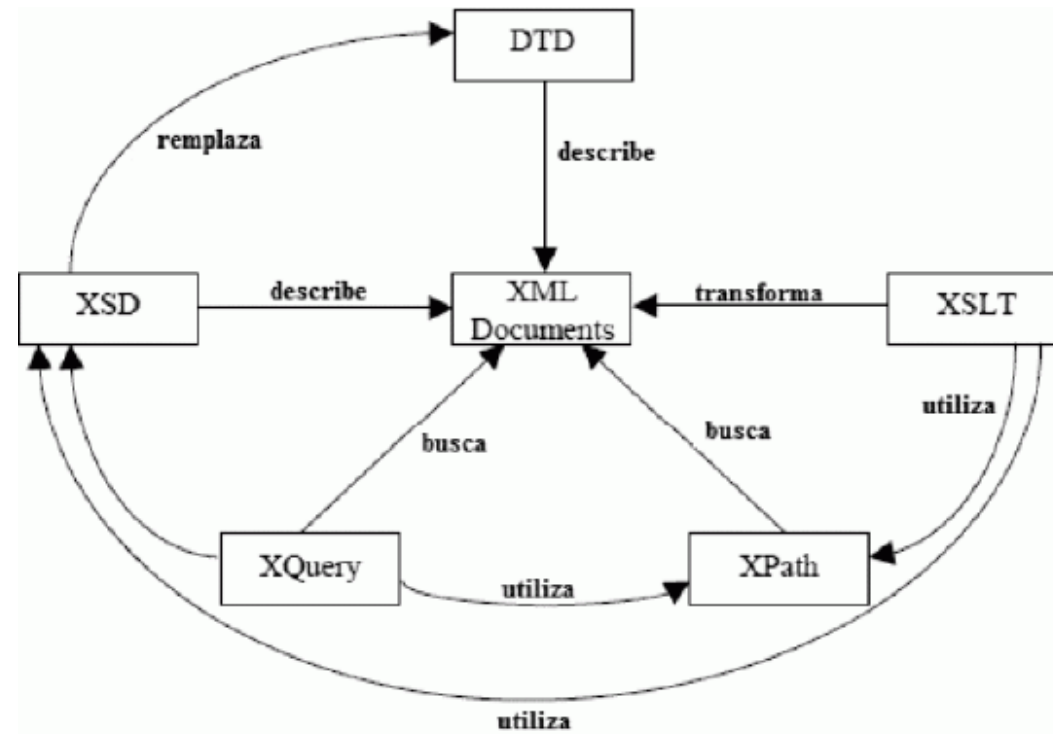
# Estándares de XML I

---

- Al ser posible exigir la estructura que deben tener un tipo determinado de documentos, se vuelve posible extraer la información de varios documentos automáticamente, por ejemplo, para crear bases de datos o listados con información sobre todos los documentos.
- El poder de XML radica no solo en que es rápido y sencillo en su implementación, también cuenta con estándares que nos permite especificar los documentos, transformaciones y búsquedas.
- En la siguiente figura podemos observar a estos estándares.

# Estándares de XML II

---





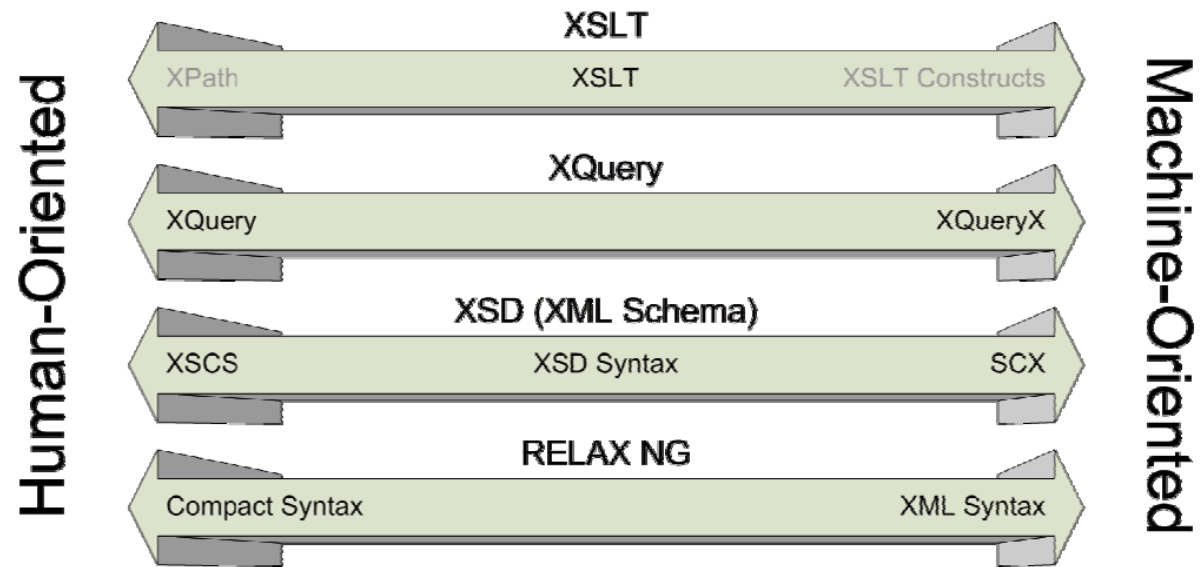
# Estándares de XML III

---

- **XSD (XML Schema)**: es un lenguaje que sirve para describir formalmente un vocabulario.
- **XSLT (eXtensible Stylesheet Language Transformations)**: es útil para transformar un documento XML basado en un esquema a un documento en diferentes formatos.
- **XPath (XML Path Language)**: forma un conjunto de expresiones útiles para crear localización de datos

# Estándares XML IV

---




# Ventajas de XML

---

- Fácilmente procesable tanto por humanos como por software.
- Separa radicalmente la información o el contenido de su presentación o formato.
- Diseñado para ser utilizado en cualquier lenguaje o alfabeto.
- Un documento, muchas formas de presentación.
- Formato ideal para transacciones B2B (transmisión de información referente a transacciones comerciales).
- Permite poderosas técnicas de extracción de información y *data-mining*.
- XML + validación = datos autodefinidos.
- Las estrictas reglas para la composición de un documento XML permiten su fácil análisis sintáctico.

# Terminología XML útil

---

◦ XML	eXtended Mark-up Language
◦ SGML	Standard Generalised Mark-up Language
◦ XML “bien-formado”	Validación
◦ DTD	Document Type Definition
◦ XSL	eXtended Stylesheet Language
◦ Parser	Analizador sintáctico
◦ DOM	Document Object Model
◦ RDF	Resource Description Framework
◦ SML	Simple Mark-up Language
◦ SMIL	Synchronized Multimedia Integration Language
◦ XHTML	XML and HTML
◦ Elemento	 Conceptos XML
◦ Atributo	
◦ XLink, Xpointer	
◦ Schema	

# Tecnologías XML I

---

- **XSL:** Lenguaje Extensible de Hojas de Estilo, cuyo objetivo principal es mostrar cómo debería estar estructurado el contenido, cómo debería ser diseñado el contenido de origen y cómo debería ser paginado en un medio de presentación como puede ser una ventana de un navegador Web o un dispositivo móvil, o un conjunto de páginas de un catálogo, informe o libro.
- **XPath:** Lenguaje de Rutas XML, es un lenguaje para acceder a partes de un documento XML.
- **XLink:** Lenguaje de Enlace XML, es un lenguaje que permite insertar elementos en documentos XML para crear enlaces entre recursos XML.

# Tecnologías XML II

---

- **XPointer**: Lenguaje de Direccionamiento XML, es un lenguaje que permite el acceso a la estructura interna de un documento XML, esto es, a sus elementos, atributos y contenido.
- **XQL**: Lenguaje de Consulta XML, es un lenguaje que facilita la extracción de datos desde documentos XML. Ofrece la posibilidad de realizar consultas flexibles para extraer datos de documentos XML en la Web.
- Para más información consulta:

<http://www.w3.org/XML/>

# Tecnologías XML III

---

- Especificación XML v1.0
- Definición de documentos DTD or Schemas
- Definición de estilos XSL = XSLT + Xpath
- Enlazado de documentos XLL = XLink + Xpointer
- Aplicaciones RDF, SMIL, HTML, etc.

# Construcción de documentos XML I

---

- La siguiente figura ilustra el elemento y sus atributos:



- **XML “bien-formado” (well-formed)**: Se dice que un documento XML es “bien formado” cuando cumple una serie de reglas descritas en la especificación oficial de XML v1.0



# Construcción de documentos XML II

---

- La estructura jerárquica de los elementos:
  - Los elementos deben seguir una estructura de “árbol”, es decir, estrictamente jerárquica.
  - Los elementos deben estar correctamente anidados.
  - Los elementos no se pueden superponer entre ellos.
- Sólo puede haber un elemento raíz, en el que están contenidos todos los demás.

# El prólogo

---

- El prólogo es opcional.
  - La primera línea permite especificar la versión de XML (de momento sólo “1.0”), la codificación de carácter (US-ASCII, UTF-8, UTF-7, UCS-2, EUCJP, Big5, ISO- 8859-1, ISO-8859-7, etc.), y algunas otras cosas.
  - La segunda línea define el tipo de documento, especificando que DTD valida y define los datos que contiene.

## Ejemplo 1

```
<?xml version="1.0" encoding="UTF-7"?>  
<!DOCTYPE mensaje SYSTEM "mensaje.dtd">
```

## Ejemplo 2

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<!DOCTYPE HTML PUBLIC "-//W3C/ DTD HTML 3.2 Final/ EN">
```

## Ejemplo 3

```
<?xml version="1.0" encoding="Big5"?>
```

# Elementos I

---

- Los elementos XML pueden tener contenido (más elementos, caracteres, o ambos a la vez), o bien ser elementos vacíos.
- Un elemento con contenido es, por ejemplo:

```
<nombre>Fulgencio Menéndez</nombre>
```

```
<aviso tipo="emergencia" gravedad="mortal">Que no cunda el pánico</aviso>
```

- Siempre empieza con una `<etiqueta>` que puede contener atributos o no, y termina con una `</etiqueta>` que debe tener el mismo nombre. Al contrario que HTML, en XML siempre se debe "cerrar" un elemento.

# Elementos II

---

- Hay que tener en cuenta que el símbolo "<" siempre se interpreta como inicio de una etiqueta XML.
- Si no es el caso, el documento no estará bien-formado.
- Para usar ciertos símbolos se usan las entidades predefinidas, que se explican más adelante.

# Elementos vacíos

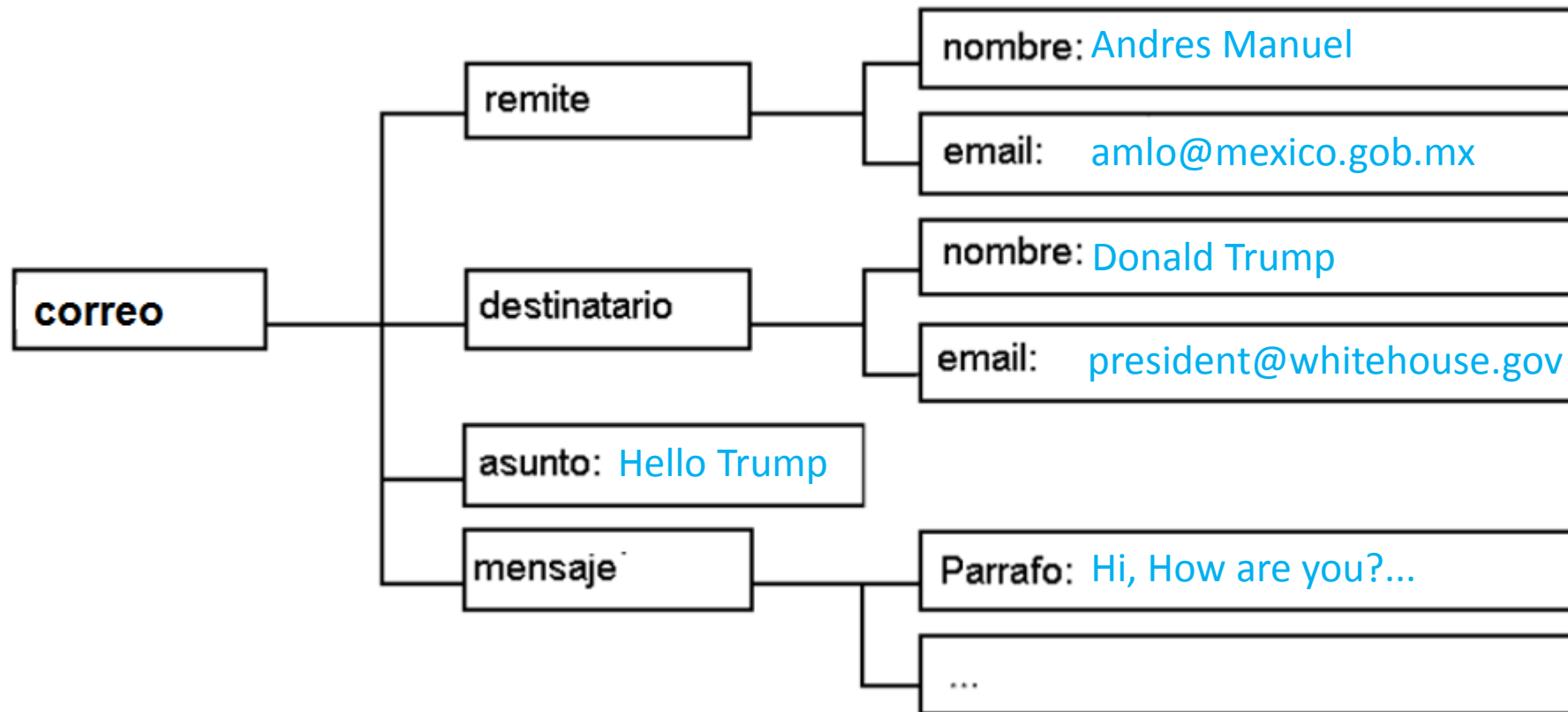
---

- Un elemento vacío, es el que no tiene contenido (claro).

```
<identificador referencia="23123244"/>  
<linea-horizontal/>
```

- Al no tener una etiqueta de "cierre" que delimite un contenido, se utiliza la forma `<etiqueta/>`, que puede contener atributos o no.
  - La sintaxis de HTML permite etiquetas vacías tipo `<hr>` o ``.
  - En HTML reformulado para que sea un documento XML bien formado, se debería usar `<hr/>` o ``

# Estructura jerárquica de elementos II



# Estructura jerárquica de elementos I

---

```
<?xml version="1.0" encoding="UTF-7"?>
<!DOCTYPE mensaje SYSTEM "mensaje.dtd">
<correo>
  <remite>
    <nombre>Andres Manuel</nombre>
    <email>amlo@mexico.gob.mx</email>
  </remite>
  <destinatario>
    <nombre>Donald Trump</nombre>
    <email>president@whitehouse.gov</email>
  </destinatario>
  <asunto>Hello Trump</asunto>
  <mensaje>
    <parrafo>Hi, How are you?...</parrafo>
  </mensaje>
</correo>
```

# Etiquetas XML

- Todas las etiquetas tienen que estar debidamente “cerradas” siempre, es decir, con una etiqueta de cierre que se corresponda con la de apertura.
- Las etiquetas “vacías” (es decir, sin contenido) tienen una sintaxis especial.

```
<animal>Perro  
  <raza tipo="Cocker Spaniel">  
<animal>Vaca  
  <raza tipo="Holstein">
```



```
<animal>Perro  
  <raza tipo="Cocker Spaniel"/>  
</animal>  
<animal>Vaca  
  <raza tipo="Holstein"/>  
</animal>
```

```
<animal>Perro  
  <raza tipo="Cocker Spaniel">  
</animal>  
<animal>Vaca  
  <raza tipo="Holstein">  
</animal>
```



```
<animal>  
  <nombre>Perro</nombre>  
  <raza tipo="Cocker Spaniel"/>  
</animal>  
<animal>  
  <nombre>Vaca</nombre>  
  <raza tipo="Holstein"/>  
</animal>
```



# Atributos XML I

---

- Los valores de los atributos de los elementos siempre deben estar marcados con las comillas dobles (“ ”) o sencillas (‘ ’)

- `<a href="http://www.disney.com">Esto es correcto</a>` ← más utilizado
- `<a href='http://www.disney.com'>Esto es correcto</a>`
- `<a href=http://www.disney.com>Esto NO es correcto</a>`

- Sintaxis para los *nombres*

- Un nombre de elemento, atributo, entidad, etc. empieza por una letra, y continua con letras, dígitos, guiones, rayas, punto o dos puntos.
- Las letras “XML” (o “xml” o “xMI”, etc.) no pueden usarse como caracteres iniciales de un nombre de elemento, atributo, entidad, etc.

# Atributos XML II

---

- Otras reglas:
  - XML es “case-sensitive”, es decir, que no es lo mismo `<autor>` que `<Autor>`, por ejemplo.
  - El uso del espacio blanco y los saltos de línea, en general funciona como en HTML (sólo se toma en cuenta cuando aparece en el valor de un atributo, o cuando se indica su significancia).
- Como ya se mencionó, los elementos pueden tener atributos, que son una manera de incorporar características o propiedades a los elementos de un documento.

```
<gato raza="Persa">Micifuz</gato>
```

# Atributos XML III

- Al igual que en otras cadenas literales de XML, los atributos pueden estar marcados entre comillas verticales ( ' ) o dobles ( " ). Cuando se usa uno para delimitar el valor del atributo, el otro tipo se puede usar dentro.

```
<verdura clase="zanahoria" longitud='15" y media'>  
<cita texto="'Hola buenos días', dijo él">
```

- A veces, un elemento con contenido, puede modelarse como un elemento vacío con atributos. Un concepto se puede representar de muy diversas formas, pero una vez elegida una, es aconsejable fijarla en el DTD, y usar siempre la misma regla.

1

```
<gato>  
  <nombre>Micifuz</nombre>  
  <raza>Persa</raza>  
</gato>
```

2

```
<gato raza="Persa">Micifuz</gato>
```

3

```
<gato raza="Persa" nombre="Micifuz"/>
```

# Marcado y datos

---

- Las construcciones como etiquetas, referencias de entidad y declaraciones se denominan “marcas” (mark-up).
- Estas son las partes del documento que el analizador sintáctico (parser) espera comprender.
- El resto del documento, que se encuentra entre las “marcas”, son los datos que resultan entendibles por las personas.
- Las marcas en un documento XML son aquellas que comienzan por el carácter “<” y terminan con “>”.
- En el caso de las referencias de entidad, el carácter inicial es “&” y el final es “;”

# Entidades predefinidas

---

- En XML 1.0, se definen cinco entidades para representar caracteres especiales y que no se interpreten como marcado por el procesador XML.
- Es decir, que así podemos usar el carácter "<" sin que se interprete como el comienzo de una etiqueta XML, por ejemplo.

Entidad	Carácter
&amp;	&
&lt;	<
&gt;	>
&apos;	'
&quot;	"

# Secciones CDATA I

---

- Existe otra construcción en XML que permite especificar datos, utilizando cualquier carácter, especial o no, sin que se interprete como marcado XML.
- La razón de esta construcción llamada CDATA (Character Data) es que a veces es necesario para los autores de documentos XML, poder leerlo fácilmente sin tener que descifrar los códigos de entidades. Especialmente cuando son muchas.

```
<ejemplo>
  &lt;HTML>
  &lt;HEAD>&lt;TITLE>Rock & Roll&lt;/TITLE>
  &lt;/HEAD>
</ejemplo>
```

```
<ejemplo>
  <![CDATA[
  <HTML>
  <HEAD><TITLE>Rock & Roll</TITLE></HEAD>
  ]]>
</ejemplo>
```

# Secciones CDATA II

---

- Como hemos visto, dentro de una sección CDATA podemos poner cualquier cosa, que no será interpretada como algo que no es.
- Existe empero una excepción, y es la cadena "]]>" con la que termina el bloque CDATA.
- Esta cadena no puede utilizarse dentro de una sección CDATA.

# Comentarios I

---

- A veces es conveniente insertar comentarios en el documento XML, que sean ignorados por el procesamiento de la información y las reproducciones del documento.
- Los comentarios tienen el mismo formato que los comentarios de HTML. Es decir, comienzan por la cadena "`<!--`" y terminan con "`-->`".
- Se pueden introducir comentarios en cualquier lugar de la instancia o del prólogo, pero nunca dentro de las declaraciones, etiquetas, u otros comentarios.



# Comentarios II

---

```
<?xml version="1.0"?>
<!-- Aquí va el tipo de documento -->
<!DOCTYPE EJEMPLO [
  <!-- Esto es un comentario -->
  <!ELEMENTO EJEMPLO (#PCDATA)>
  <!-- ¡Eso es todo por ahora! -->
]>
<EJEMPLO>
  texto texto texto bla bla bla
  <!-- Otro comentario -->
</EJEMPLO>
<!-- Ya acabamos -->
```