Apuntes XSD

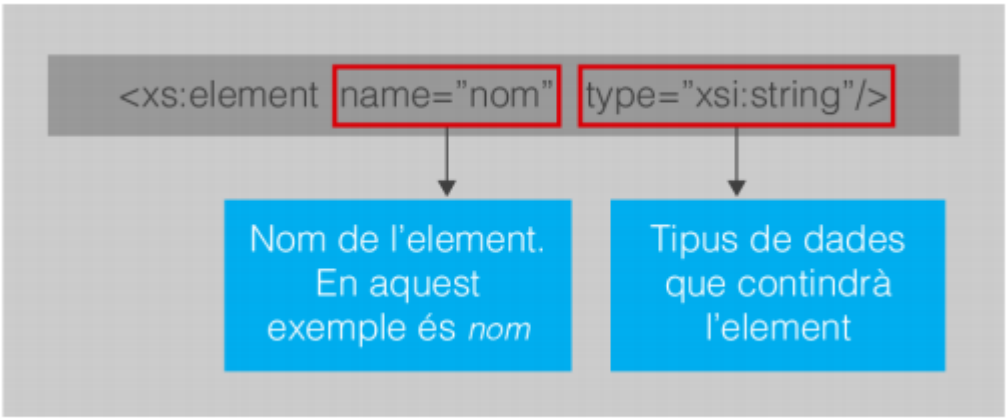
- Se creó para sustituir DTD.
- Está escrito en XML y por lo tanto no hace falta aprenderse otro lenguaje nuevo de esquemas XML.
- Tiene su propio sistema de datos.
- Soporta espacio de nombres.
- Permite ser reutilizado y sigue el modelo de programación con herencia de objeto y sustitución de tipos.

1. **Asociar un esquema a un fichero XML**

<realestate xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="realestate.xsd"> → Siempre va al principio y se utiliza para definir dónde está el fichero xsd.

2. **Definir un fichero de SCHEMA**

FIGURA 2.11. Definició d'un element



- XSD se divide en 2 grupos de datos:
  - Elementos de **TIPO SIMPLE:** Elementos sin atributos, sólo datos
  - Elementos de **TIPO COMPLEJO:** Elementos que pueden tener atributos, no tener contenido o contener elementos.

ELEMENTOS CON CONTENIDO DE TIPO SIMPLE

- **TIPO SIMPLE:** No tienen otros elementos ni atributos. El type=”xs: ” puede ser…

Taula 2.9. Tipus de dades més usats en XSD

Tipus	Dades que s'hi poden emmagatzemar
string	Cadenes de caràcters
decimal	Valors numèrics
boolean	Només pot contenir 'true' o 'false' o (1 o 0)
date	Dates en forma (AAAA-MM-DD)
anyURI	Referències a llocs (URL, camins de disc...)
base64binary	Dades binàries codificades en *base64*
integer	Nombres enters

XSD

<xs:element name=”posicio” type=”xs:integer” /> | *Se autocierra

</xs:schema>

XML

<posicio>Primer</posicio>

*Solo conseguirá validar un elemento si su contenido es un nombre entero.

Etiqueta	Exemple
<xs:element name="dia" type="xs:date" />	<dia>2011-09-15</dia>
<xs:element name="alçada" type="xs:integer"/>	<alçada>220</alçada>
<xs:element name="nom" type="xs:string"/>	<nom>Pere Puig</nom>
<xs:element name="mida" type="xs:float"/>	<mida>1.7E2</mida>
<xs:element name="lloc" type="xs:anyURI"/>	<lloc>http://www.ioc.cat</lloc>

Si nos fijamos la etiqueta **dia** es del tipo **date**.

LETRAS → **XS:STRING**

NUMEROS → **XS:INTEGER**

DECIMALES →**XS:FLOAT**

URL → **XS:ANYURI**

- **minOccurs:** Permite definir **cuantas veces** saldrá este elemento como mínimo. Añadir un valor **'0'** indica que este elemento **no** podrá salir.
- **maxOccurs:** Sirve para definir **cuantas veces** saldrá como máximo este elemento. **Unbounded** implica que **no** hay límite.

Fent servir els atributs es pot definir que l'element <nom> ha de sortir una vegada i l'element <cognom> un màxim de dues vegades.

```
<xs:element name="nom" />
<xs:element name="cognom" maxOccurs="2"/>
```

Saldrá como máximo 2 veces.

<u>También se pueden dar valores a los elementos con atributos **FIXED, DEFAULT y NULLABLE.**</u>

L'atribut fixed permet definir un valor obligatori per a un element:

```
<xs:element name="centre" type="xs:string" fixed="IOC"/>
```

<centre>IOC</centre>

DEFAULT → Deja un valor por defecto

LA ESTRUCTURA ES LA MISMA

1 <xs:element name="persona"> → Nombre del elemento/etiqueta

2 <xs:simpleType> → Tipo de XSD SIMPLE

3

4 </xs:simpleType>

5 </xs:element>

1 <xs:element name="partidos"> → Nombre del elemento/etiqueta

2 <xs:simpleType> → Tipo de XSD SIMPLE

3 <xs:lista itemType="xs:date"/> → **lista**

4 </xs:simpleType>

5 </xs:element>

Lista permitirá definir un elemento que puede contener listas de valores. LISTA DE DATOS.

L'etiqueta validaria amb una cosa com:

```
<partits> 2011-01-07 2011-01-15 2011-01-21</partits>
```

Los elementos **simpleType** también se pueden definir con un nombre fuera de los **elementos y** posteriormente utilizarlos como tipo de datos personal.

1. <xs:simpleType name=”dies”>
    a. <xs:lista itemType=”xs:date” />
2. </xs:simpleType>

3. <xs:element name=”partits” type=”dies” /> → fuera del elemento

---

Union permite que se puedan **mezclar tipos diferentes** en el contenido **de un elemento.**

La definició de l'element `<preu>` farà que l'element pugui ser de tipus `valor` o de tipus `simbol`.

```
<xs:element name="preus">
    <xs:sipleType>
        <xs:union memberTypes="valor simbol"/>
    </xs:simpleType>
</xs:element>
```

Amb això li podríem assignar valors com aquests:

```
<preus>25 €</preus>
```

RESTRICTION define restricciones para que se puedan aceptar algunos determinados valores.

<xs:simpleType name=”any_naixement”> →

<xs:restriction base=”xs:integer”> → Tipo entero del tipo RESTRICTION

<xs:maxInclusive value=”2011” /> → Valor máximo que puede llegar

<xs:minInclusive value=”1850” /> → Valor mínimo

<xs:restriction>

</xs.simpleType>

<xs:element name=”naixement” type=”any_naixement” />

1. Primero definir el elemento fuera y después con una simpleType o complexType crear sub elementos.

Es poden definir restriccions de molts tipus per mitjà d'atributs (taula 2.11). Normalment els valors de les restriccions s'especifiquen en l'atribut `value`:

TAULA 2.11. Atributs que permeten definir restriccions en XSD

Elements	Resultat
`maxInclusive / maxExclusive`	Es fa servir per definir el valor numèric màxim que pot agafar un element.
`minInclusive / minExclusive`	Permet definir el valor mínim del valor d'un element.
`length`	Amb `lenght` restringim la llargada que pot tenir un element de text. Podem fer servir `<xs:minLength>` i `<xs:maxLenght>` per ser més precisos.
`enumeration`	Només permet que l'element tingui algun dels valors especificats en les diferents línies `<enumeration>`.
`totalDigits`	Permet definir el nombre de dígits d'un valor numèric.
`fractionDigits`	Serveix per especificar el nombre de decimals que pot tenir un valor numèric.
`pattern`	Permet definir una expressió regular a la qual el valor de l'element s'ha d'adaptar per poder ser vàlid.

Enumeration

<xs:enumeration value="A" />

ATRIBUTO PATTERN → Restricciones de expresiones regulares con condiciones.

Símbolo	Equivalencia
.	Cualquier carácter
\d	Cualqueir dígito
\D	Cualquier carácter no digito
\s	Caracteres no imprimibles
\S	Cualquier carácter imprimible
x*	Saldrá 0 o más veces
x+	Saldrá 1 o más veces
x?	Puede salir o no
[abc]	Puede ser 'a' 'b' o 'c'
[0-9]	Del 0 al 9, extremos incluidos
x{5}	Habrá 5 veces el valor
x{5,}	Saldrá 5 o más veces
x{5,8}	Habrá entre 5 y 8 veces

exemple, podem definir que una dada ha de tenir la forma d'un DNI (8 xifres, un guió i una lletra majúscula) amb aquesta expressió:

```
<xs:simpleType name="dni">
    <xs:restriction base="xs:string">
        <xs:pattern value="[0–9]{8}–[A–Z]"/>
    </xs:restriction>
</xs:simpleType>
```

ELEMENTOS CON CONTENIDO DE TIPO COMPLEJO

Elementos que contienen otros elementos o no tienen contenido.

1. Tienen más datos. Son como los del tipo simple, pero con **atributos.**
2. Contenido con más elementos.
3. Los elementos vacíos.
4. Elementos con contenido mezclado.

<xs:complexType> → Para definir un XSD del tipo complejo

```
<xs:element name="classe">
    <xs:complexType>
          ....
    </xs:complexType>
</xs:element>
```

De la mateixa manera que amb els tipus simples, es poden definir tipus complexos amb nom per reutilitzar-los com a tipus personalitzats.

```
<xs:complexType name="curs">
     ...
</xs:complexType>

<xs:element classe type="curs"/>
```

Pueden utilizar atributos:

Els tipus de dades són els mateixos i, per tant, poden tenir tipus bàsics com en l'exemple següent:

```
<xs:attribute name="número" type="xs:integer" />
```

```
<xs:attribute name="any">
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:maxInclusive="2011"/>
        </xs:restriction>
</xs:attribute>
```

<attribute> tiene una serie de caracteristicas:

Atributo	Uso
Use	Obligatorio (required) | Ocional (optional)
Default	Valor por defecto
Fixed	Valor obligatorio
Form	Permite definir si el atributo puede ir con alias en el espacio de nombres (qualifies) o no (unqualified)

'simpleContent'

Va en complexType y será un simpleContent → Permite definir restricciones o extensiones de elementos.

En aquest exemple l'element <Mida> té contingut de tipus enter i defineix dos atributs, llargada i amplada, que també són enters.

```
<xs:complexType name="Mida">
    <xs:simpleContent>
        <xs:extension base="xs:integer">
           <xs:atribute name="llargada" type="xs:integer"/>
           <xs:atribute name="amplada" type="xs:integer"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

EL ORDEN ES IMPORTANTE

per validar han d'aparèixer en el mateix ordre en el qual es defineixen en la seqüència.

```
<xs:element name="persona">
    <xs:complexContent>
        <xs:sequence>
            <xs:element name="nom" type="xs:string"/>
            <xs:element name="cognom" type="xs:string" maxOccurs="2"/>
            <xs:element name="tipus" type="xs:string" />
        </xs:sequence>
    </xs:complexContent>
</xs:element>
```

En l'exemple anterior es defineix que abans de l'aparició de <tipus> poden aparèixer un o dos cognoms.

```
<persona>
    <nom>Marcel</nom>
    <cognom>Puig</cognom>
    <cognom>Lozano</cognom>
    <tipus>Professor</tipus>
</persona>
```