Tema 3

VALIDAR UN DOCUMENTO. XML SCHEMA

Espacios de nombres en XML

XML NAMESPACES

XML Namespaces

- La idea de los namespaces en XML es **evitar conflictos** en el nombre de los elementos

```
<table>
    <tr>
    <td>Uno</td>
    <td>Dos</td>
    </tr>
</table>
```

```
<table>
    <nombre>iPhone 4</nombre>
    <ancho>5,86</ancho>
    <largo>11,52</largo>
</table>
```

Misma etiqueta. Diferente contenido y significado

Posible solución. Prefijos

```
<h:table>
    <h:tr>
    <h:td>Uno</h:td>
    <h:td>Dos</h:td>
    </h:tr>
</h:table>

<m:table>
    <m:nombre>iPhone 4</m:nombre>
    <m:ancho>5,86</m:ancho>
    <m:largo>11,52</m:largo>
</m:table>
```

Atributo xmlns

- Este atributo sirve para declarar un espacio de nombres asociado a un prefijo
- El espacio de nombres se define mediante este atributo (**`xmlns`**) en la **etiqueta de inicio del elemento**

Atributo xmlns

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

  <m:table xmlns:m="http://www.upsa.es/movilidad">
     <m:nombre>iPhone 4</m:nombre>
     <m:ancho>5,86</m:ancho>
     <m:largo>11,52</m:largo>
  </m:table>
</root>
```

Atributo xmlns

```
<root
    xmlns:h="http://www.w3.org/TR/html4/"
    xmlns:m="http://www.upsa.es/movilidad">
  <h:table>
     <h:tr>
     <h:td>Uno</h:td>
     <h:td>Dos</h:td>
     </h:tr>
  </h:table>

  <m:table>
     <m:nombre>iPhone 4</m:nombre>
     <m:ancho>5,86</m:ancho>
     <m:largo>11,52</m:largo>
  </m:table>
</root>
```

Nombre de un espacio de nombres

- El nombre debe ser único

```
<h:table xmlns:h="http://www.w3.org/TR/html4/">
```

- En realidad es una cadena pero debería declararse como un URI.
- Utilizar **URI** reduce las posibilidades de que diferentes espacios de nombres usen identificadores duplicados

DTD y espacio de nombres

- Los nombres de los elementos son ahora todo.

```
<m:table>
    <m:nombre>iPhone 4</m:nombre>
    <m:ancho>5,86</m:ancho>
    <m:largo>11,52</m:largo>
</m:table>
```

.xml

```
<!ELEMENT m:nombre (#PCDATA)>
```

.dtd

Validar un documento con un esquema XML

XML Schema (W3C)

XML Schema (W3C)

INTRODUCCIÓN

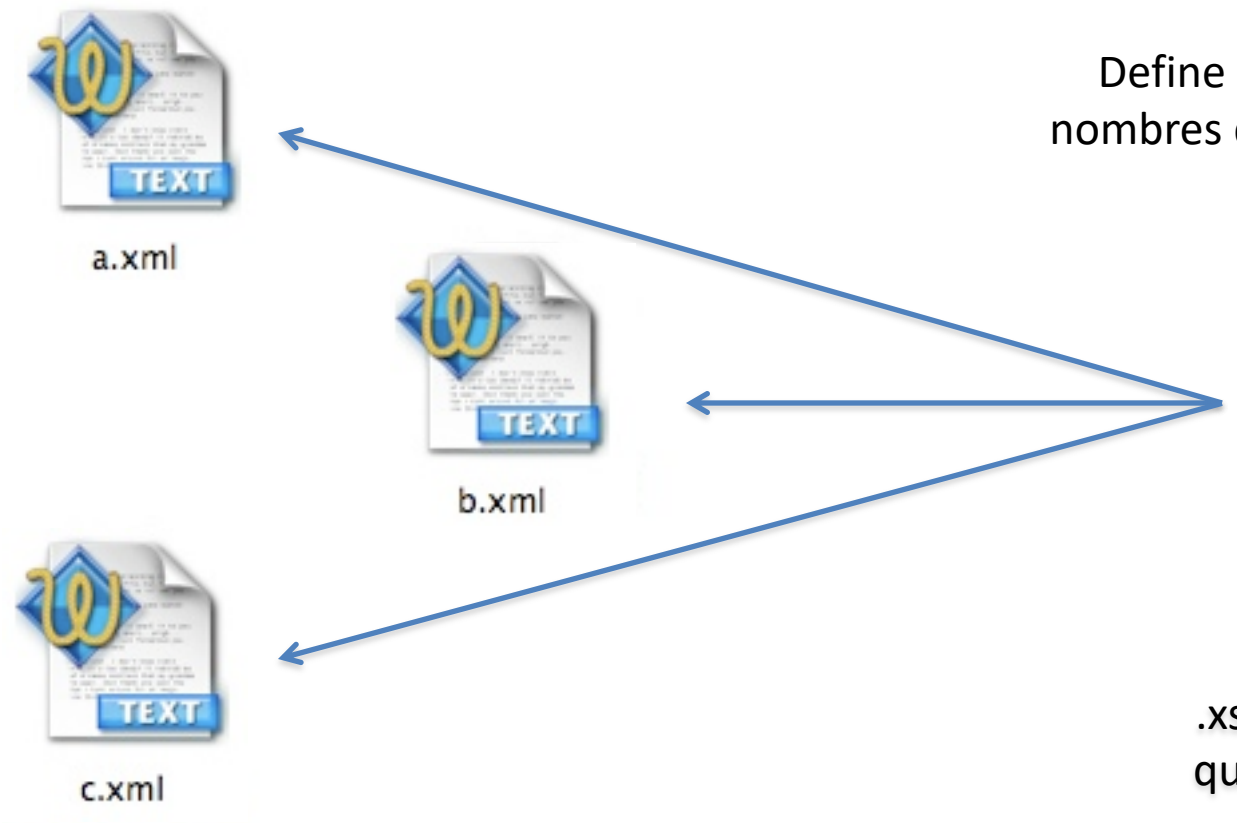
XML Schema (W3C)

- Lenguaje para definir un conjunto de reglas para **validar** documentos XML
- Recomentación del **W3C** (2001)
- Más **potente** que DTD
  - Tipos de datos
- **Es XML** y usa *espacios de nombres XML*

Schema vs. DTD

- XML Schemas son extensibles
- XML Schemas son más potentes que los DTD
- XML Schemas son XML
- XML Schemas soportan tipos de datos
- XML Schemas soportan namespaces

Schema



Define la estructura, etiquetas, nombres de los elementos, orden …

.xsd es el documento que define el schema XML

Conectarlo a un XML

- Se usan espacios de nombres

Esto se incluye en el XML

```
<?xml version="1.0"?>
<root
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="biblioteca.xsd">
    <libro>...</libro>
</root>
```

Debe ser validado contra un esquema

Dónde está el esquema.

Ejemplo Schema

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="libro" type="tipoLibro"/>

   <xs:complexType name="tipoLibro">
       <xs:sequence>
           <xs:element name="autor" type="xs:string"/>
           <xs:element name="titulo" type="xs:string"/>
       </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Quitando los *namespace*

```
<?xml version="1.0"?>
<schema>
    <element name="libro" type="tipoLibro"/>

   <complexType name="tipoLibro">
       <sequence>
           <element name="autor" type="string"/>
           <element name="titulo" type="string"/>
       </sequence>
    </complexType>
</schema>
```

No parece tan raro después de todo

XML Schema (W3C)

La base del Schema

La base

- Elementos simples
- Atributos
- Tipos de datos

La base

- **Elementos simples**
- Atributos
- Tipos de datos

Elementos **simples**

- Aquellos que contienen solo texto.
- Se definen usando el elemento **`element`**

```
<xs:element name="nombreEle" type="tipo"/>
```

IMPORTANTE!!
NO pueden tener atributos

Elementos **simples**

- Valores por defecto (atributo **`default`**)
- Valores fijos (atributo **`fixed`**)

```
<xs:element name="color" type="xs:string" default="red"/>
```

```
<xs:element name="color" type="xs:string" fixed="red"/>
```

La base

- Elementos simples
- **Atributos**
- Tipos de datos

Atributos

- Se definen usando el elemento **attribute**

```
<xs:attribute name="nombreAtributo" type="tipo"/>
```

```
<xs:attribute name="lang" type="xs:string"/>
```

Atributos

- Valores por defecto (atributo **`default`**)
- Valores fijos (atributo **`fixed`**)

```
<xs:attribute name="color" type="xs:string" default="red"/>
```

```
<xs:attribute name="color" type="xs:string" fixed="red"/>
```

Atributos

- Opcional (por defecto)
- Obligatorio (**`use="required"`**)

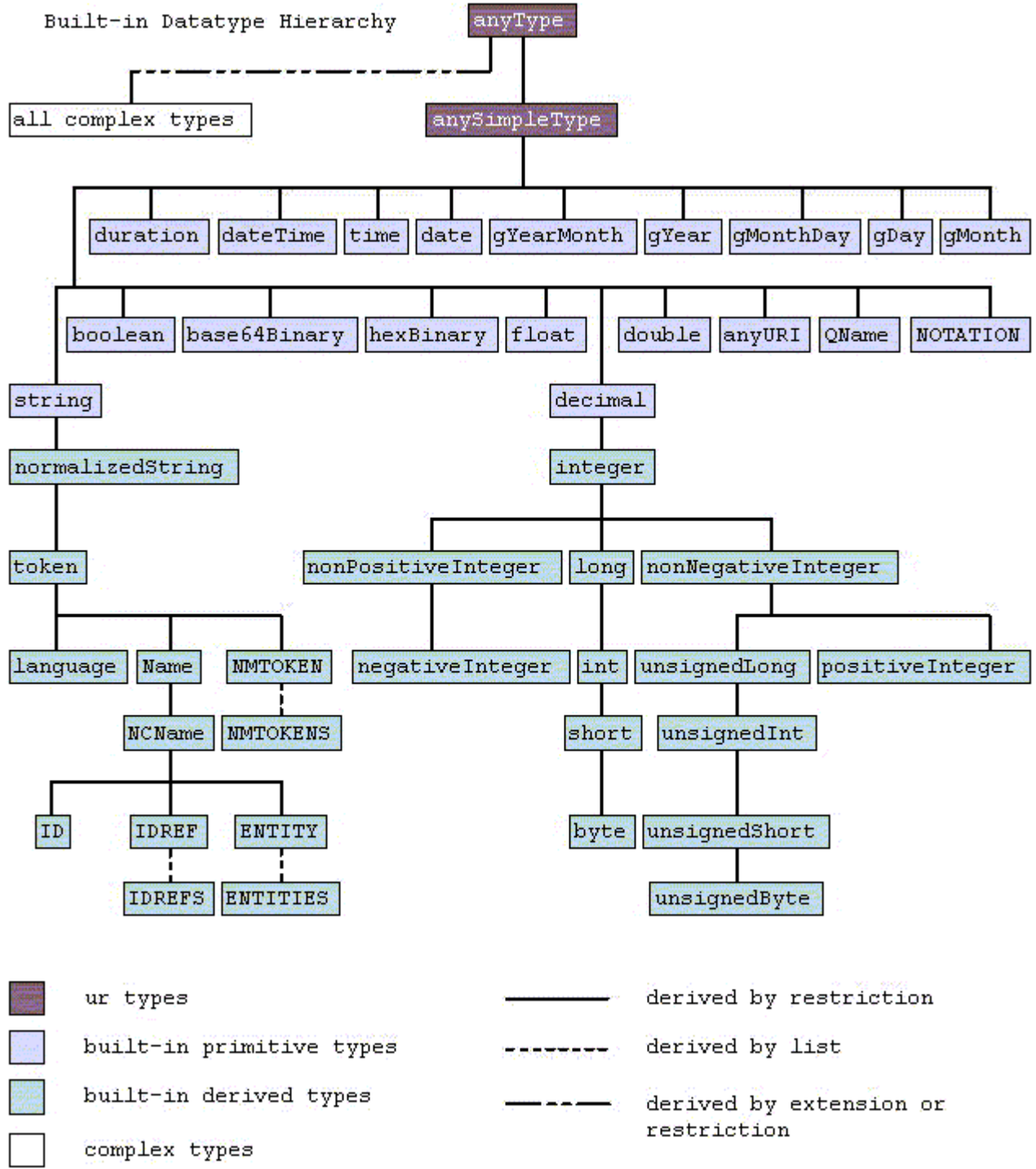
```
<xs:attribute name="lang" type="xs:string" use="required"/>
```

La base

- Elementos simples
- Atributos
- **Tipos de datos**

Tipos de datos

- Tipos definidos (W3C)
  - http://www.w3.org/TR/xmlschema-2/
- Mis propios tipos de datos


Built-in Datatype Hierarchy
anyType
all complex types
anySimpleType
duration
dateTime
time
date
gYearMonth
gYear
gMonthDay
gDay
gMonth
boolean
base64Binary
hexBinary
float
double
anyURI
QName
NOTATION
string
decimal
normalizedString
integer
token
nonPositiveInteger
long
nonNegativeInteger
language
Name
NMTOKEN
negativeInteger
int
unsignedLong
positiveInteger
NCName
NMTOKENS
short
unsignedInt
ID
IDREF
ENTITY
byte
unsignedShort
IDREFS
ENTITIES
unsignedByte
ur types
built-in primitive types
built-in derived types
complex types
derived by restriction
derived by list
derived by extension or restriction

Tipos definidos **más habituales**

string	**boolean**	**decimal**
time (hh:mm:ss.sss)	**date** (yyyy-mm-dd)	**integer**
anyURI	ID, IDREF, IDREFS	NMTOKEN, NMTOKENS

Ejemplo de uso. Tipos de datos

```
<xs:element name="nombre" type="xs:string" />
<xs:element name="sabeNadar" type="xs:boolean"/>
<xs:element name="FechaNacimiento" type="xs:date"/>
```

.xsd

```
<nombre>Encarna</nombre>
<sabeNadar>true</sabeNadar>
<fechaNacimiento>1978-03-27</fechaNacimiento>
```

.xml

Tipos de datos

- Tipos definidos (W3C)
  - http://www.w3.org/TR/xmlschema-2/
- Mis propios tipos de datos

Tipos de datos propios

1. Tipos con nombre
2. Tipos anónimos

1) Tipos con nombre

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="nota" type="tipoNota"/>

    <xs:simpleType name="tipoNota">
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="10"/>
        </xs:restriction>
    </xs:simpleType>


</xs:schema>
```

2) Tipos anónimos

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="nota">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:minInclusive value="0"/>
                <xs:maxInclusive value="10"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

</xs:schema>
```

No aparece type

Ventaja de los tipos con nombre

- Reutilización

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="nota"       type="tipoNota"/>
    <xs:element name="notaFinal" type="tipoNota"/>

    <xs:simpleType name="tipoNota">
         <xs:restriction base="xs:integer">
              <xs:minInclusive value="0"/>
              <xs:maxInclusive value="10"/>
         </xs:restriction>
    </xs:simpleType>



</xs:schema>
```

Tipos de datos propios

- Tipos complejos. (`xs:complexType`)
- Tipos simples. (`xs:simpleType`)

Tipos de datos propios

- Tipos complejos. (`xs:complexType`)
- Tipos simples. (`xs:simpleType`)

Tipos complejos

- Se utilizan para definir **elementos complejos**. Aquellos que contienen otros elementos y/o atributos
- Sirven para especificar la estructura del documento

Elemento complejo.
Elemento que NO es simple, es decir tiene atributos o elementos hijo

Elementos complejos

1
Elementos vacíos que **sólo contienen atributos**

2
Elementos que contienen elementos **hijos** (también pueden tener atributos)

3
Elementos que contienen **texto y atributos**

4
Elementos **mixtos** que contienen tanto texto como hijos (también pueden tener atributos)

Elementos complejos

1

Elementos vacíos que **sólo contienen atributos**

Elemento complejo vacío

```
<producto prodid="1345" />
```

.xml

```
<xs:element name="producto">
   <xs:complexType>
     <xs:attribute name="prodid" type="xs:integer"/>
   </xs:complexType>
 </xsd:element>
```

.xsd

```
<xsd:element name="producto" type="tipoProducto"/>

 <xs:complexType name="tipoProducto">
   <xs:attribute name="prodid" type="xs:integer"/>
 </xs:complexType>
```

.xsd

Elementos complejos

2

Elementos que contienen elementos **hijos** (también pueden tener atributos)

Elemento con elementos hijos

```xml
<persona>
   <firstname>Encarna</firstname>
   <lastname>Beato</lastname>
 </persona>
```

.xml

```xsd
<xs:element name="persona" type="tipoPersona"/>

 <xs:complexType name="tipoPersona">
   <xs:sequence>
     <xs:element name="firstname" type="xs:string"/>
     <xs:element name="lastname"  type="xs:string"/>
   </xs:sequence>
 </xs:complexType>
```

.xsd

Elemento con elementos hijos y atributos

```xml
<persona codigo = "ABC">
   <firstname>Encarna</firstname>
   <lastname>Beato</lastname>
 </persona>
```

.xml

```xml
<xs:element name="persona" type="tipoPersona"/>

 <xs:complexType name="tipoPersona">
   <xs:sequence>
     <xs:element name="firstname" type="xs:string"/>
     <xs:element name="lastname"  type="xs:string"/>
   </xs:sequence>
   <xs:attribute name="codigo" type="xs:string"/>
 </xs:complexType>
```

.xsd

Elementos con hijos. Indicadores de orden

- Existen elementos en el XMLSchema que indica cómo aparecen los elementos hijo
  - **`all`** Todos en cualquier orden
  - **`choice`** Uno de ellos
  - **`sequence`** Todos en ese orden

¡IMPORTANTE!
Pueden tener también indicadores de cardinalidad

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:all>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:all>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <lastname>Beato</lastname>
   <firstname>Encarna</firstname>
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:all>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:all>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <firstname>Encarna</firstname>
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:choice>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:choice>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <firstname>Encarna</firstname>
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:choice>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:choice>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <firstname>Encarna</firstname>
   <lastname>Beato</lastname
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <lastname>Beato</lastname>
   <firstname>Encarna</firstname>
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
   <firstname>Encarna</firstname>
</persona>
```

¿Válido?

.xml

Indicadores de orden

```
<xs:element name="persona">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="firstname" type="xs:string"/>
       <xs:element name="lastname"  type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
 </xs:element>
```

.xsd

```
<persona>
  <firstname>Encarna</firstname>
  <lastname>Beato</lastname>
</persona>
```

¿Válido?

.xml

Elementos complejos

3

Elementos que contienen **texto y atributos**

Elementos sólo con texto y atributos

```
<xs:element name="nombreElemento">
   <xs:complexType>
     <xs:simpleContent>
       <xs:extension base="tipoBase">
            <xs:attribute name="nombreAtt" type="tipoAtt"/>
               ...
       </xs:extension>
     </xs:simpleContent>
   </xs:complexType>
 </xs:element>
```

Elementos sólo con texto y atributos

```
<zapato pais="España">37</zapato>
```

.xml

```
<xs:element name="zapato" type="tipoZapato"/>

<xs:complexType name="tipoZapato">
   <xs:simpleContent>
     <xs:extension base="xs:integer">
       <xs:attribute name="pais" type="xs:string" />
     </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

.xsd

Elementos complejos

4

Elementos **mixtos** que contienen tanto texto como hijos (también pueden tener atributos)

Elementos mixtos

```
<carta>
   Buenos días <nombre>Encarna Beato</nombre>.
   Su pedido <pedidoid>1032</pedidoid>
   se entrega el  <fechaEntrega>2001-07-13</fechaEntrega>.
</carta>
```

.xml

```
<xs:element name="carta" type="tipoCarta"/>

<xs:complexType name="tipoCarta" mixed="true">
   <xs:sequence>
     <xs:element name="nombre"        type="xs:string"/>
     <xs:element name="pedidoid"      type="xs:integer"/>
     <xs:element name="fechaEntrega" type="xs:date"/>
   </xd:sequence>
</xs:complexType>
```

.xsd

Tipos de datos propios

- Tipos complejos. (`xs:complexType`)
- Tipos simples. (**`xs:simpleType`**)

Tipos Simples

- Para almacenar sólo datos
- Basados en un tipo definido al que aplico alguna restricción.

Restricciones/aspectos

- Se usan para restringir los valores que pueden tener los elementos o atributos en XML
- Se definen asociados a un tipo de datos

Restricciones/aspectos

- Se definen dentro de un tipo (`xs:simpleType`)

```
<xs:simpleType name="nombreTipo">
   <xs:restriction base="tipoBase">
       <xs:algunaRestriccion value="valor"/>
         ...
   </xs:restriction>
</xs:simpleType>
```

Tipo base sobre el que se hace la restricción

Restricciones/aspectos

- **length, minLength, maxLength**
- **pattern**
- **enumeration**
- **whiteSpace**
- **maxInclusive, maxExclusive, minInclusive, minExclusive**
- **totalDigits**
- **fractionDigits**

Restricciones/aspectos

- **length, minLength, maxLength**
- pattern
- enumeration
- whiteSpace
- maxInclusive, maxExclusive, minInclusive, minExclusive
- totalDigits
- fractionDigits

Restricciones/aspectos

- **length.** Número de caracteres o lista de items permitidos
- **minLength.** Mínimo
- **maxLength.** Máximo

```
<xs:element name="password">
   <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:minLength value="5"/>
        <xs:maxLength value="8"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Restricciones/aspectos

- length, minLength, maxLength
- **pattern**
- enumeration
- whiteSpace
- maxInclusive, maxExclusive, minInclusive, minExclusive
- totalDigits
- fractionDigits

Restricciones/aspectos

- **`pattern`** Define el patrón que deben tener los datos que se almacenan. Usa sintaxis de **expresiones regulares**

Expresiones regulares (resumen)

Patrón	Coincide con
`(A)`	Una cadena que coincide con A
`A | B`	Una cadena que coincide con A o con B
`AB`	Una cadena que coincide con A seguida de otra que coincide con B
`A?`	Opcionalidad de A
`A*`	0 o más repeticiones de A
`A+`	1 0 más repeticiones de A
`A{n}`	n repeticiones de A
`A{n,m}`	De n a m repeticiones de A
`[abcd]`	Alguno de los caracteres a b c o d
`[a-z]`	Alguna letra minúscula
`[^ab]`	Un carácter que no sea ni a ni b

Ejemplo pattern

```
<xs:element name="letra">
   <xs:simpleType>
     <xs:restriction base="xs:string">
        <xs:pattern value="[a-z]"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejemplo pattern

```
<xs:element name="iniciales">
  <xs:simpleType>
     <xs:restriction base="xs:string">
        <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejemplo pattern

```
<xs:element name="genero">
  <xs:simpleType>
     <xs:restriction base="xs:string">
        <xs:pattern value="masculino|femenino"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejemplo pattern

```
<xs:element name="palabra">
  <xs:simpleType>
     <xs:restriction base="xs:string">
        <xs:pattern value="([a-z])*"/>
    </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejercicio pattern

Describe un patrón para que el elemento password contenga sólo letras o dígitos y tenga longitud 8

```
<xs:element name="password">
  <xs:simpleType>
     <xs:restriction base="xs:string">
       <xs:pattern value="([a-zA-Z0-9]){8}"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejercicio pattern

Describe un patrón para que el elemento identificador contenga una secuencia de letras y dígitos que comienza por una letra

```
<xs:element name="identificador">
  <xs:simpleType>
     <xs:restriction base="xs:string">
       <xs:pattern value="[a-zA-Z]([a-zA-Z0-9])*"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Ejercicio pattern

Describe un patrón para que el elemento numero contenga un valor positivo par

```
<xs:element name="numero">
  <xs:simpleType>
     <xs:restriction base="xs:integer">
       <xs:pattern value="([0-9])*([02468])+"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Restricciones/aspectos

- length, minLength, maxLength
- pattern
- **enumeration**
- whiteSpace
- maxInclusive, maxExclusive, minInclusive, minExclusive
- totalDigits
- fractionDigits

Restricciones/aspectos

- **`Enumeration`** Uno de los valores especificados

```
<xs:element name="coche">
  <xs:simpleType>
     <xs:restriction base="xs:string">
       <xs:enumeration value="Audi"/>
       <xs:enumeration value="Golf"/>
       <xs:enumeration value="BMW"/>
     </xs:restriction>
   </xs:simpleType>
 </xs:element>
```

Restricciones/aspectos

- length, minLength, maxLength
- pattern
- enumeration
- **whiteSpace**
- maxInclusive, maxExclusive, minInclusive, minExclusive
- totalDigits
- fractionDigits

Restricciones/aspectos

- **whiteSpace** Tres valores posibles:
  - **preserve** Los conserva tal cual
  - **replace** Reemplaza tabulador, retorno por espacio
  - **collapse** Reemplaza y luego deja en uno sólo los espacios consecutivos

```
<xs:element name="direccion">
  <xs:simpleType>
     <xs:restriction base="xs:string">
       <xs:whiteSpace value="preserve"/>
     </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Restricciones/aspectos

- length, minLength, maxLength
- pattern
- enumeration
- whiteSpace
- **maxInclusive, maxExclusive, minInclusive, minExclusive**
- totalDigits
- fractionDigits

Restricciones/aspectos

- **maxInclusive, maxExclusive, minInclusive, minExclusive**
  Para valores numéricos. Máximo y mínimo incluidos o no

```
<xs:element name="edad">
  <xs:simpleType>
     <xs:restriction base="xs:integer">
       <xs:minInclusive value="0"/>
       <xs:maxInclusive value="120"/>
     </xs:restriction>
   </xs:simpleType>
 </xs:element>
```

Restricciones/aspectos

- length, minLength, maxLength
- pattern
- enumeration
- whiteSpace
- maxInclusive, maxExclusive, minInclusive, minExclusive
- **totalDigits**
- **fractionDigits**

Restricciones/aspectos

- **totalDigits**
- **fractionDigits**

  Para valores numéricos. Longitud y precisión

```
<xs:element name="peso">
  <xs:simpleType>
     <xs:restriction base="xs:decimal">
       <xs:fractionDigits value="2"/>
     </xs:restriction>
   </xs:simpleType>
 </xs:element>
```

Claridad del .xsd

TIPOS ANÓNIMOS VS. TIPOS CON NOMBRE

Tipos anónimos vs. con nombre

```
<?xml version="1.0"?>
<estudiantes>
  <estudiante>
     <nombre>
       <firstname>Mario</firstname>
     </nombre>
  </estudiante>
</estudiantes>
```

.xml

Tipos anónimos

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="estudiantes">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="estudiante">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="nombre">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="firstname"
                                                    type="xs:string"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

.xsd

Alguien puede encontrar el error aquí?

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="estudiantes">
<xs:complexType>
<xs:sequence>
<xs:element name="estudiante">
  <xs:complexType>
   <xs:sequence>
<xs:element name="nombre">
 <xs:complexType>
   <xs:sequence>
 <xs:element name="firstname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
 </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>

</xs:schema>
```

Tipos con nombre

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="students" type="tipo_estudiantes" />

  <xs:complexType name="tipo_estudiantes">
    <xs:sequence>
        <xs:element name="estudiante" name="tipo_estudiante" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="tipo_estudiante">
    <xs:sequence>
        <xs:element name="nombre" name="tipo_nombre" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="tipo_nombre">
    <xs:sequence>
        <xs:element name="firstname" name="xs:string" />
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

XMLSchema W3C

INDICADORES

Indicadores

- Dicen cómo aparecen los elementos
  - De orden
  - De cardinalidad
  - De grupo

Indicadores

- Dicen cómo aparecen los elementos
  - **De orden**
  - De cardinalidad
  - De grupo

Indicadores de orden

- Dicen cómo aparecen los elementos
  - **`all`** Todos en cualquier orden
  - **`choice`** Uno de ellos
  - **`sequence`** Todos en ese orden

Indicadores

- Dicen cómo aparecen los elementos
  - De orden
  - **De cardinalidad**
  - De grupo

Indicadores de cardinalidad

- Cuántos elementos de este tipo pueden aparecer
  - **`minOccurs`**
  - **`maxOccurs`**

- Por defecto y sin límite
  - Por defecto **`minOccurs`** es **`1`**
  - Por defecto **`maxOccurs`** es igual a **`1`**
  - **`maxOccurs="unbounded"`**

Indicadores de cardinalidad

```
<xs:element name="fecha" type="xs:date" maxOccurs="2"/>
```

.xsd

```
<fecha>2012-03-13</fecha>
<fecha>2010-06-20</fecha>
```

¿Válido?

.xml

Indicadores de cardinalidad

```
<xs:element name="contacto" type="tContacto"/>

<xs:complexType name="tContacto">
   <xs:sequence>
      <xs:element name="email" type="xs:string"/>
      <xs:element name="redesSociales" type="xs:string" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

.xsd

```
<contacto>
    <email>ebeatogu@upsa.es</email>
    <redesSociales>@ebeatogu</redesSociales>
    <redesSociales>@ebeatogu2</redesSociales>
</contacto>
```

.xml

¿Válido?

Sí

Indicadores de cardinalidad

```
<xs:element name="contacto" type="tContacto"/>

<xs:complexType name="tContacto">
   <xs:sequence>
      <xs:element name="email" type="xs:string"/>
      <xs:element name="redesSociales" type="xs:string" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

.xsd

```
<contacto>
   <redesSociales>@ebeatogu</redesSociales>
   <email>ebeatogu@upsa.es</email>
   <email>ebeato@upsa.es</email>
   <redesSociales>@ebeatogu2</redesSociales>
</contacto>
```

.xml

¿Válido?

Indicadores de cardinalidad

```
<xs:element name="contacto" type="tContacto"/>

<xs:complexType name="tContacto">
    <xs:choice>
        <xs:element name="email" type="xs:string" maxOccurs="unbounded"/>
        <xs:element name="redesSociales" type="xs:string" maxOccurs="unbounded"/>
    </xs:choice>
</xs:complexType>
```

.xsd

```
<contacto>
   <redesSociales>@ebeatogu</redesSociales>
   <email>ebeatogu@upsa.es</email>
   <email>ebeato@upsa.es</email>
   <redesSociales>@ebeatogu2</redesSociales>
</contacto>
```

.xml

¿Válido?

Indicadores de cardinalidad

```
<xs:element name="contacto" type="tContacto"/>

<xs:complexType name="tContacto">
    <xs:choice maxOccurs="unbounded">
        <xs:element name="email" type="xs:string" />
        <xs:element name="redesSociales" type="xs:string"/>
    </xs:choice>
</xs:complexType>
```

.xsd

```
<contacto>
   <redesSociales>@ebeatogu</redesSociales>
   <email>ebeatogu@upsa.es</email>
   <email>ebeato@upsa.es</email>
   <redesSociales>@ebeatogu2</redesSociales>
</contacto>
```

.xml

¿Válido?

Indicadores

- Dicen cómo aparecen los elementos
  - De orden
  - De cardinalidad
  - **De grupo**

Indicadores de grupo

- Definen conjuntos de elementos o atributos
  - **`group`**
  - **`attributeGroup`**

Indicadores de grupo

```
<xs:group name="persongroup">
   <xs:sequence>
     <xs:element name="firstname" type="xs:string"/>
     <xs:element name="lastname"  type="xs:string"/>
     <xs:element name="birthday"  type="xs:date"/>
   </xs:sequence>
</xs:group>

<xs:element name="person" type="personinfo"/>

<xs:complexType name="personinfo">
   <xs:sequence>
     <xs:group ref="persongroup"/>
     <xs:element name="country" type="xs:string"/>
   </xs:sequence>
</xs:complexType>
```

Indicadores de grupo

```
<xs:attributeGroup name="personattrgroup">
   <xs:attribute name="firstname" type="xs:string"/>
   <xs:attribute name="lastname"  type="xs:string"/>
   <xs:attribute name="birthday"  type="xs:date"/>
</xs:attributeGroup>

<xs:element name="person">
   <xs:complexType>
     <xs:attributeGroup ref="personattrgroup"/>
   </xs:complexType>
</xs:element>
```

XMLSchema W3C

EXCLUSIVIDAD Y CLAVES

Exclusividad y Claves

- Permiten expresar relaciones entre los valores de los elementos y de los atributos.
- Más potente que **`ID`** e **`IDREF`**

Exclusividad. **`unique`**

- **`unique`** define que el valor de un elemento o atributo debe ser único en un ámbito
- El elemento **`unique`** debe contener
  - Un atributo **`name`**
  - Un elemento **`selector`**
  - Uno o varios elementos **`field`**

Exclusividad

- El elemento **`selector`** debe contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que seleccionará sobre qué elementos va a actuar
- Los elementos **`field`** deben contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que especifica qué valores deben ser únicos de entre los seleccionados por el selector
  - Si hay más de un elemento **`field`**, la combinación de los elementos **`field`** debe ser única.

xPath

- **Lenguaje** para **encontrar información** en un documento XML
- Lo hace a través del **árbol**
- **Similar** a la navegación por el árbol de **directorios**

Expresiones *path (resumen)*

- XPath utiliza *expresiones path* para seleccionar elementos en un documento XML

Expresión	Descripción
nombreNodo	Selecciona todos los nodos con ese nombre hijos del actual
nodo1/nodo2	Selecciona todos los nodos2 que estén por debajo de nodo1 en el árbol
@nombreAtributo	Selecciona ese atributo
.//nombreNodo	Selecciona todos los nodos con ese nombre que estén por debajo del actual en uno o más pasos
.	Selecciona el nodo actual

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

`/facultad/alumno`

¿Qué selecciona?

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

`/facultad/alumno`

¿Qué selecciona?

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

`profesor`

¿Qué selecciona?

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

`profesor`

¿Qué selecciona?

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

```
.//nombre
```

¿Qué selecciona?

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Expresión

```
.//nombre
```

¿Qué selecciona?

Exclusividad. **`unique`**

- **`unique`** define que el valor de un elemento o atributo debe ser único en un ámbito
- El elemento **`unique`** debe contener
  - Un atributo **`name`**
  - Un elemento **`selector`**
  - Uno o varios elementos **`field`**

Claves. **`key`**

- **`key`** define que el valor de un elemento o atributo debe ser único, no vacío y estar siempre presente
- El elemento **`key`** debe contener
  - Un atributo **`name`**
  - Un elemento **`selector`**
  - Uno o varios elementos **`field`**

Claves

- El elemento **`selector`** debe contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que seleccionará sobre qué elementos va a actuar
- Los elementos **`field`** deben contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que especifica que valores deben ser únicos de entre los seleccionados por el selector
  - Si hay más de un elemento **`field`**, la combinación de los elementos **`field`** debe ser clave.

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Ejemplo

No puede haber dos asignaturas con el mismo nombre

Claves

```
<xs:element name="facultad">
    ...
    <xs:unique name="NombreAsignatura">
        <xs:selector xpath="asignatura"/>
        <xs:field    xpath="nombreAsig"/>
    </xs:unique>
</xs:element>
```

.xsd

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Ejemplo

No puede haber dos profesores con el mismo nif

Claves

```
<xs:element name="facultad">
   ...
   <xs:key name="profesor">
        <xs:selector xpath="profesor"/>
        <xs:field    xpath="@nif"/>
   </xs:key>
</xs:element>
```

.xsd

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Ejemplo

No puede haber dos alumnos con el mismo nombre y apellidos

Claves

```
<xs:element name="facultad">
   ...
   <xs:key name="Profesor">
        <xs:selector xpath="alumno"/>
        <xs:field    xpath="nombre"/>
        <xs:field    xpath="apellidos"/>
   </xs:key>
</xs:element>
```

.xsd

Referencia a claves. **keyref**

- **keyref** especifica que el valor o conjunto de valores de un atributo o elemento corresponden a los del elemento **key** o **unique** especificados.
- El elemento **keyref** debe contener
  - Un atributo **name**
  - Un atributo **refer**
  - Un elemento **selector**
  - Uno o varios elementos **field**

Referencia a claves. **keyref**

- **keyref** especifica que el valor o conjunto de valores de un atributo o elemento corresponden a los del elemento **key** o **unique** especificados.
- Atributo **name**. Nombre para el **keyref**
- Atributo **refer**. Nombre del elemento **key** o **unique** al que se refiere

Claves

- El elemento **`selector`** debe contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que seleccionará sobre qué elementos va a actuar
- Los elementos **`field`** deben contener
  - Un atributo **`xpath`** (expresión **`xPath`**) que especifica que valores deben estar entre los seleccionados por la referencia (**`refer`**)
  - Si hay más de un elemento **`field`**, la combinación de los elementos **`field`** debe estar entre los de **`refer`**.

```
<facultad>
  <asignatura profesores="_07856902Q">
    <nombreAsig>Programación</nombreAsig>
    <nCreditos>6</nCreditos>
    <matriculado alumno = "_45214"/>
    <matriculado alumno = "_33991"/>
  </asignatura>
<!-- alumnos -->
  <alumno nExpediente = "_33991">
    <nombre>Luis Antonio</nombre>
    <apellidos>Martín Blázquez
    </apellidos>
  </alumno>
  <alumno nExpediente = "_45214">
    <nombre>Pedro</nombre>
    <apellidos>Martín García</apellidos>
  </alumno>
  <profesor nif="_07856902Q">
    <nombre>Encarna</nombre>
    <apellidos>Beato Gutiérrez</apellidos>
  </profesor>
</facultad>
```

Ejemplo

Los profesores de las asignaturas debe ser alguno de los de la facultad (nif)

Claves

```
<xs:element name="facultad">
   ...
   <xs:key name="ProfesorKey">
       <xs:selector xpath="profesor"/>
       <xs:field    xpath="@nif"/>
   </xs:key>
<xs:keyref name="profesorAsignatura" refer="ProfesorKey">
       <xs:selector xpath="asignatura"/>
       <xs:field    xpath="@profesores"/>
   </xs:keyref>
</xs:element>
```

.xsd

¿Preguntas?

Tema 3

VALIDAR UN DOCUMENTO. XML SCHEMA