ESQUEMA DE VALIDACIÓN

PÉREZ SELLERS, ALEJANDRO

LENGUAJE DE MARCAS 1º de DAW

Ejercicio 1

Cantidades limitadas con atributo divisa

Se desea crear un esquema para validar XML en los que haya un solo elemento raíz llamado cantidad en el que se debe poner siempre un atributo «divisa» que indique en qué moneda está una cierta cantidad. El atributo divisa siempre será una cadena y la cantidad siempre será un tipo numérico que acepte decimales (por ejemplo `float`). El esquema debe validar los archivos siguientes:

```
<cantidad divisa="euro">20</cantidad>
```

```
<cantidad divisa="dolar">18.32</cantidad>
```

Pero no debe validar ninguno de los siguientes:

```
<cantidad>20</cantidad>
```

```
<cantidad divisa="dolar">abc</cantidad>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="cantidad">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:float">
          <xs:attribute name="divisa" type="xs:string" use="required" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 2.

Cantidades limitadas con atributo divisa con solo ciertos valores

Queremos ampliar el ejercicio anterior para evitar que ocurran errores como el siguiente:

```
<cantidad divisa="aaaa">18.32</cantidad>
```

Vamos a indicar que el atributo solo puede tomar tres posibles valores: «euros», «dolares» y «yenes».

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="cantidad">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:float">
          <xs:attribute name="divisa" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="euros" />
                <xs:enumeration value="dolares" />
                <xs:enumeration value="libras" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 3

Ejercicio: codigos y sedes

Se necesita tener un esquema que valide un fichero en el que hay un solo elemento llamado `codigo`

- Dentro de código hay una cadena con una estructura rígida: 2 letras mayúsculas, seguidas de 2 cifras, seguidas a su vez de 3 letras.
- El elemento `código` debe llevar un atributo `sede` que será de tipo cadena.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="codigo">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="sede" type="xs:string" use="required" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name="codigo">
    <xs:simpleType>
      <xs:restriction>
        <xs:pattern value="[A-Z]{2}[0-9]{2}[A-Z]{3}" />
      </xs:restriction>
       </simpleType>
</sx:schema>
```

Ejercicio 4.

Ejercicio: productos con atributos

Se desea crear un esquema que permita validar un elemento raíz llamado `producto` de tipo `xsd:string`. El producto tiene dos atributos:

- Un atributo se llamará `cantidad` y es obligatorio. Debe aceptar solo enteros positivos.
- También habrá un atributo llamado `unidad` que solo acepta los `xsd:string` «cajas» y «pales».

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="producto" type="xs:string">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">

          <xs:attribute name="cantidad" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:positiveInteger" />
            </xs:simpleType>
          </xs:attribute>


          <xs:attribute name="unidad" use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="cajas" />
                <xs:enumeration value="pales" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 5.

Ejercicio: clientes con información adicional

Se desea crear un esquema XML que permita validar un elemento llamado `cliente` que puede almacenar un `xsd:string`. El cliente contiene:

- Un atributo obligatorio llamado `codigo` que contiene el código del cliente, que siempre consta de tres letras mayúsculas de tres números.
- Un atributo optativo llamado `habitual` que se usará para saber si es un cliente habitual o no. Acepta valores «true» y «false».
- Un atributo optativo llamado `cantidad` que indica su compra. Es un entero con valores de entre 0 y 1000.

```xml
<cliente codigo="AAA222" habitual="true" cantidad="1000">
    Pepe Pérez
</cliente>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="cliente">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="codigo" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:pattern value="[A-Z]{3}[0-9]{3}" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="habitual" use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="true" />
                <xs:enumeration value="false" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
            <xs:attribute name="cantidad" use="optional">
            <xs:simpleType>
              <xs:restriction base="xs:integer">
                <xs:minInclusive value="0" />
                <xs:maxInclusive value="1000" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 6.

Ejercicio: lista de códigos

Se nos pide crear un esquema que permita validar un fichero como el siguiente:

```
<listacodigos>
  <codigo>AAA2DD</codigo>
  <codigo>BBB2EE</codigo>
  <codigo>BBB2EE</codigo>
</listacodigos>
```

En concreto, todo código tiene la estructura siguiente:

1. Primero van tres mayúsculas
2. Despues va exactamente un digito.
3. Por último hay exactamente dos mayúsculas.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="listacodigos">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="codigo" maxOccurs="unbounded">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:pattern value="[A-Z]{3}[0-9]{1}[A-Z]{2}" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 7

Ejercicio: otra lista de clientes

Ahora se nos pide crear un esquema que permita validar un fichero como el siguiente, en el que hay una lista de clientes y el nombre es optativo, aunque los apellidos son obligatorios:

```xml
<listaclientes>
  <cliente>
    <nombre>Juan</nombre>
    <apellidos>Sanchez</apellidos>
  </cliente>
  <cliente>
    <nombre>Jose</nombre>
    <apellidos>Diaz</apellidos>
  </cliente>
</listaclientes>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="listaclientes">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="cliente" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nombre" type="xs:string" minOccurs="0"/>
              <xs:element name="apellidos" type="xs:string" minOccurs="1"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 8.

Ejercicio: lista de alumnos

Se desea construir un esquema para validar listas de alumnos en las que:

- La raíz es `listaalumnos`.
- Dentro de ella hay uno o más `alumno`. Todo `alumno` tiene siempre un atributo DNI que es obligatorio y que tiene una estructura formada por 7 u 8 cifras seguidas de una mayúscula.
- Todo `alumno` tiene un elemento `nombre` y un `ap1` obligatorios.
- Todo `alumno` puede tener despues del `ap1` un elemento `ap2` y uno `edad`, ambos son optativos.
- El elemento `edad` debe ser entero y positivo.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="listaalumnos">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="alumno" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nombre" type="xs:string" />
              <xs:element name="ap1" type="xs:string" />
              <xs:element name="ap2" type="xs:string" minOccurs="0" />
              <xs:element name="edad" minOccurs="0">
                <xs:simpleType>
                  <xs:restriction base="xs:positiveInteger" />
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="DNI" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:pattern value="\d{7,8}[A-Z]" />
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ejercicio 9.

Ejercicio: lista de articulos (con atributos optativos)

Supongamos el fichero siguiente con las reglas que se explicitan en los comentarios:

```xml
<listaproductos>
    <articulo>
        <!--Estructura 2 letras,2 cifras-->
        <codigo>CD12</codigo>
        <!--Descripcion es optativo y su atributo autor tb-->
        <descripcion autor="Pepe">Monitor</descripcion>
    </articulo>
    <articulo>
        <codigo>CA12</codigo>
    </articulo>
    <articulo>
        <codigo>AA99</codigo>
        <descripcion>Teclado</descripcion>
    </articulo>
</listaproductos>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="listaproductos">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="articulo" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="codigo">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:pattern value="[A-Z]{2}[0-9]{2}" />
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
              <xs:element name="descripcion" minOccurs="0">
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:string">
                      <xs:attribute name="autor" type="xs:string" use="optional" />
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```