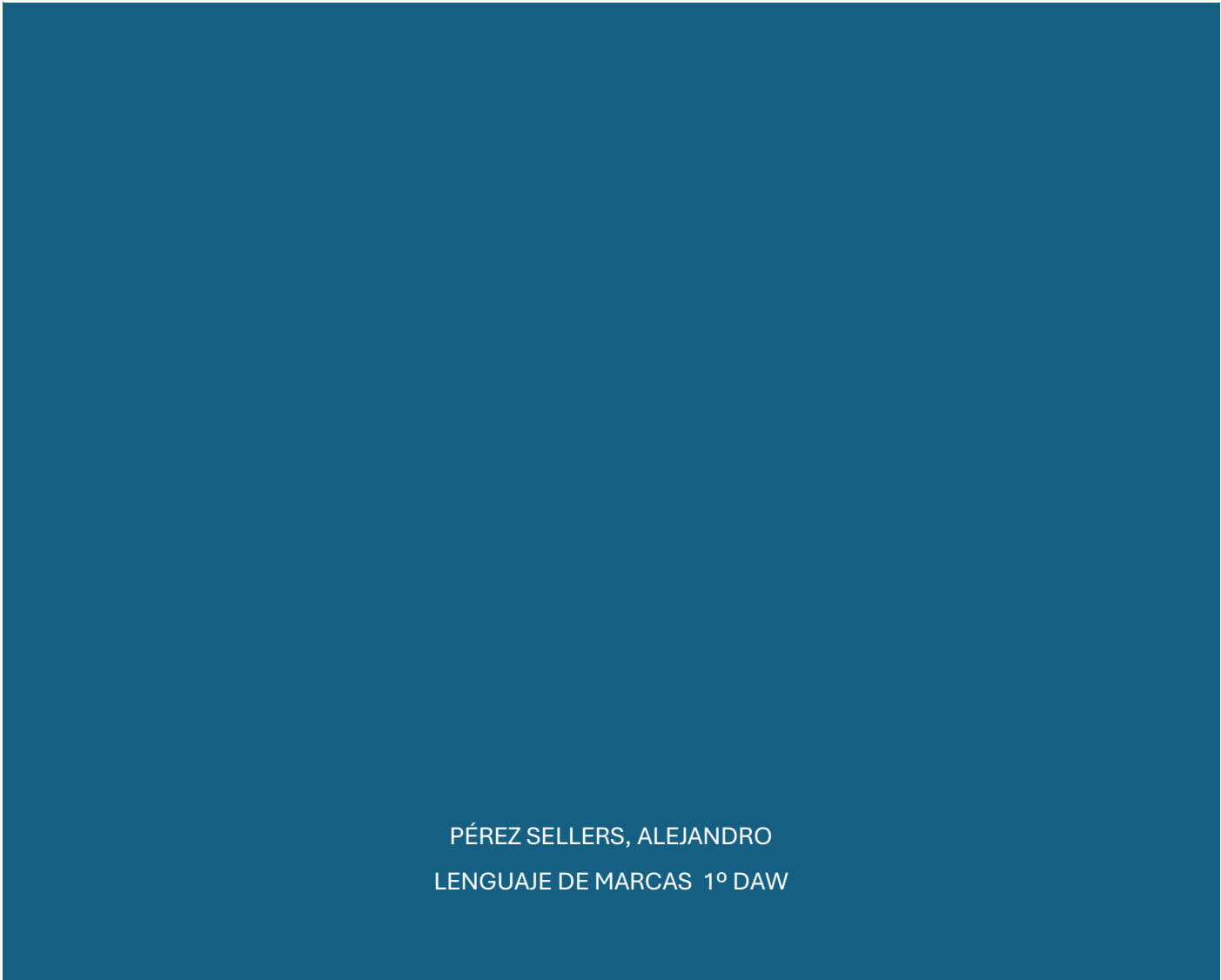




EJERCICIOS LIBRO



PÉREZ SELLERS, ALEJANDRO
LENGUAJE DE MARCAS 1º DAW

Ejercicio 1. Realiza una DTD para describir un conjunto de artículos, sabiendo que hay que almacenar el nombre, código y fabricante.

```
<!DOCTYPE articulos [  
<!ELEMENT articulos (artículos+)>  
<!ELEMENT articulos (nombre, precio, código, fabricante)>  
<!ELEMENT nombre (#PCDATA)>  
<!ELEMENT precio (#PCDATA)>  
<!ELEMENT codigo (#PCDATA)>  
<!ELEMENT fabricante (#PCDATA)>  
<!ATTLIST codigo tipo (EAN | SKU | OTRO)"OTRO" >  
<!ATTLIST precio moneda CDATA #IMPLIED>
```

<!ELEMENT articulos (artículos+)> elemento raíz es artículos y debe contener al menos un elemento artículos, uno o mas indicado por el símbolo +.

<!ELEMENT articulos (nombre, precio, código, fabricante)> cada artículos debe contener los elementos nombre, precio, código y fabricante.

<!ELEMENT nombre (#PCDATA)>

<!ELEMENT precio (#PCDATA)>

<!ELEMENT codigo (#PCDATA)>

<!ELEMENT fabricante (#PCDATA)>

Con #PCDATA indica que solo contiene texto.

<!ATTLIST codigo tipo (EAN | SKU | OTRO) "OTRO"> el elemento código tiene un atributo tipo, que puede ser:

Ean código de barras.

Sku identificación interna de inventario.

Otro otro tipo de código, si no se especifica un valor, el predeterminado será otro.

<!ATTLIST precio moneda CDATA #IMPLIED> el elemento precio puede tener un atributo moneda con cualquier valor, el #IMPLIED que significa que este atributo es opcional.

Ejercicio 2. Obtén el DTD y el XSD del siguiente documento XML:

DTD DE VIVERO.

```
C: > Users > alexp > Desktop > página 194, del 1 al 9 > vivero.dtd > ...
1
2 <!ELEMENT vivero (especie+)>
3 <!ELEMENT especie (nombre, precio, variedad, origen, color_fruto+, otros_datos)>
4 <!ATTLIST especie siembra CDATA #REQUIRED>
5
6 <!ELEMENT nombre (#PCDATA)>
7 <!ELEMENT precio (#PCDATA)>
8 <!ATTLIST precio moneda CDATA #REQUIRED>
9 <!ELEMENT variedad (#PCDATA)>
10 <!ELEMENT origen (#PCDATA)>
11 <!ELEMENT color_fruto (#PCDATA)>
12
13 <!ELEMENT otros_datos (maduracion, riego)>
14 <!ELEMENT maduracion (#PCDATA)>
15 <!ELEMENT riego (#PCDATA)>
16
```

XSD VIVERO.

```
C: > Users > alexp > Desktop > página 194, del 1 al 9 > vivero.xsd > xs:schema
1 <?xml version="1.0" encoding="iso-8859-1"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
3   <xs:element name="vivero">
4     <xs:complexType>
5       <xs:sequence>
6         <xs:element name="especie" maxOccurs="unbounded">
7           <xs:complexType>
8             <xs:sequence>
9               <xs:element name="nombre" type="xs:string"/>
10              <xs:element name="precio">
11                <xs:complexType>
12                  <xs:simpleContent>
13                    <xs:extension base="xs:decimal">
14                      <xs:attribute name="moneda" type="xs:string" use="required"/>
15                    </xs:extension>
16                  </xs:simpleContent>
17                </xs:complexType>
18              </xs:element>
19              <xs:element name="variedad" type="xs:string"/>
20              <xs:element name="origen" type="xs:string"/>
21              <xs:element name="color_fruto" type="xs:string" maxOccurs="unbounded"/>
22              <xs:element name="otros_datos">
23                <xs:complexType>
24                  <xs:sequence>
25                    <xs:element name="maduración" type="xs:string"/>
26                    <xs:element name="riego" type="xs:string"/>
27                  </xs:sequence>
28                </xs:complexType>
29              </xs:element>
30            </xs:sequence>
31            <xs:attribute name="siembra" type="xs:gYear" use="required"/>
32          </xs:complexType>
33        </xs:element>
34      </xs:sequence>
35    </xs:complexType>
36  </xs:element>
```

Ejercicio 3. Quiere implementarse en un documento XML la estructura de las calificaciones de los alumnos de un centro de Málaga, entre los que hay que guardar: datos personales (nombre y apellidos), materia (OACE y EEE) con sus respectivas tareas y cuestionarios, tal y como muestra en la siguiente figura:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <calificaciones>
3      <alumno>
4          <datos>
5              <nombre>Felipe</nombre>
6              <apellidos>Pascual</apellidos>
7          </datos>
8          <materia>
9              <OACE>
10                 <cuestionario>4</cuestionario>
11                 <tareas>8</tareas>
12             </OACE>
13             <EEE>
14                 <cuestionario>7</cuestionario>
15                 <tareas>6</tareas>
16             </EEE>
17         </materia>
18     </alumno>
19
20     <alumno>
21         <datos>
22             <nombre>Maria</nombre>
23             <apellidos>Garcia</apellidos>
24         </datos>
25         <materia>
26             <OACE>
27                 <cuestionario>9</cuestionario>
28                 <tareas>2</tareas>
29             </OACE>
30             <EEE>
31                 <cuestionario>9</cuestionario>
32                 <tareas>6</tareas>
33             </EEE>
34         </materia>
35     </alumno>
36
```

Se añade otra asignatura, completando con los valores que se estiman oportunos.

```
38      <alumno>
39          <datos>
40              <nombre>Carlos</nombre>
41              <apellidos>Fernandez</apellidos>
42          </datos>
43          <materia>
44              <OACE>
45                  < cuestionario>5</ cuestionario>
46                  < tareas>7</ tareas>
47              </OACE>
48              <EEE>
49                  < cuestionario>6</ cuestionario>
50                  < tareas>5</ tareas>
51              </EEE>
52          </materia>
53      </alumno>
54  </calificaciones>
```

Ejercicio 4. Dado el siguiente documento XML, determina sus etiquetas, elementos y atributos.

Las etiquetas son los nombres que delimitan un elemento, mientras que un elemento incluye las etiquetas y el contenido dentro de ellas.

<Agenda> Es el elemento raíz

<contacto> Elemento hijo de <agenda>

<ciudad> Elemento hijo de <contacto> contiene el valor Málaga

<teléfono> Elemento hijo de <contacto> es un elemento vacío.

<email> elemento hijo de <contacto> contiene “666777888prueba.com”

Los atributos nos dan información adicional acerca del elemento:

Nombre en <contacto> alumno LM

N en <teléfono> 666777888

Encoding en la declaración XML “utf-8”

Por lo tanto, la estructura del código quedaría en:

Elemento raíz: <agenda>

Elementos hijos: <contacto>, <ciudad>, <teléfono>, <email>

Atributos: nombre en <contacto> y n en <teléfono>

Ejercicio 5. Dado el siguiente fichero DTD, construye un documento XML con algunos datos.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Moviles>
3      <Movil>
4          <Modelo>iPhone 15</Modelo>
5          <precio>1200</precio>
6          <Fabricante>Apple</Fabricante>
7          <color>Negro</color>
8      </Movil>
9
10     <Movil>
11         <Modelo>Galaxy S23</Modelo>
12         <precio>950</precio>
13         <Fabricante>Samsung</Fabricante>
14         <color>Azul</color>
15     </Movil>
16
17     <Movil>
18         <Modelo>Xiaomi 13 Pro</Modelo>
19         <precio>850</precio>
20         <Fabricante>Xiaomi</Fabricante>
21         <color>Blanco</color>
22     </Movil>
23 </Moviles>
24
```

El xml que he creado tiene, <móviles> que es el elemento raíz y contiene uno o mas <móvil>, cada <móvil> tiene los elementos modelo, precio, fabricante y color, en este mismo orden. Todos los valores de los elementos son #PCDATA, incluye tres registros de móviles, con diferentes valores.

Ejercicio 6. Quiere almacenarse la siguiente información en un xml, que contiene datos relativos a socios de un gimnasio sabiendo que genero e id son atributos.

This XML file does not appear to have any style information associated

```
▼<gimnasio>
  ▼<persona genero="Femenino" id="001">
    <nombre>Segundo Pascual</nombre>
    ▼<telefonos>
      <telefono>6XXXXXXXXX</telefono>
      <telefono>95XXXXXXXXX</telefono>
    </telefonos>
    ▼<emails>
      <email>User1@gmail.com</email>
      <email>User2@gmail.com</email>
      <email>User3@gmail.com</email>
    </emails>
    <socio>Si</socio>
  </persona>
  ▼<persona genero="Masculino" id="002">
    <nombre>Obdulia Toledo</nombre>
    <telefonos/>
    ▼<emails>
      <email>Carmen@uma.es</email>
    </emails>
    <socio>No</socio>
  </persona>
</gimnasio>
```

Cada persona tiene un atributo genero y id, un usuario puede tener múltiples teléfonos o ninguno (teléfono*), se permite mas de un email por usuario(email+) y es posible registrar múltiples usuarios (persona+)

```
C: > Users > alexp > Desktop > página 194, del 1 al 9 > ≡ gimnasio.dtd > ...
1  <!ELEMENT gimnasio (persona+)>
2  <!ELEMENT persona (nombre, telefonos, emails, socio)>
3  <!ATTLIST persona genero CDATA #REQUIRED id CDATA #REQUIRED>
4  <!ELEMENT nombre (#PCDATA)>
5  <!ELEMENT telefonos (telefono*)>
6  <!ELEMENT telefono (#PCDATA)>
7  <!ELEMENT emails (email+)>
8  <!ELEMENT email (#PCDATA)>
9  <!ELEMENT socio (#PCDATA)>
10
```

El elemento raiz es gimnasio y contiene una o mas personas (persona+), cada persona tiene cuatro subelementos obligatorias: nombre, teléfonos, emails y socio, los atributos de persona tienen dos obligatorios genero e id.

En cuanto a los subelementos tenemos:

<!ELEMENT NOMBRE (#PCDATA)>: El nombre de la persona es texto plano.

<!ELEMENT TELEFONOS (TELEFONO*) : La lista de teléfonos puede contener ninguno o más elementos teléfono.

<!ELEMENT EMAIL (email+)>: La lista de correo electrónico debe de contener por lo menos uno

<!ELEMENT email (#PCDATA)>: Cada dirección de correo electrónico es texto.

<!ELEMENT socio (#PCDATA)>: Indica si la persona es socia del gimnasio o no, también como texto.

Ejercicio 7. Dado el siguiente XML:

Completa el siguiente fichero DTD:

<!ELEMENT agenda (contacto+)>

<!ELEMENT contacto (nombre_apellidos, telefono+, edad, email+, direccion)>

<!ELEMENT nombre_apellidos (#PCDATA)>

<!ELEMENT telefono (#PCDATA)>

<!ELEMENT edad (#PCDATA)>

<!ELEMENT email (#PCDATA)>

<!ELEMENT direccion (calle, cod_post, provincia)>

<!ELEMENT calle (#PCDATA)>

<!ELEMENT cod_post (#PCDATA)>

<!ELEMENT provincia (#PCDATA)>

Ejercicio 8. Responde a las siguientes preguntas, argumentando el resultado:

a) ¿Podrías añadir 7 contactos más a la agenda?

Si, se pueden añadir 7 contactos mas, ya que el elemento contacto tiene el atributo maxOccurs="unbounded", lo que significa que no hay un limite en la cantidad de contactos que se pueden agregar en la agenda.

b) Si un usuario deja el campo teléfono sin rellenar, ¿Estaría bien?

Si, estaría bien. El campo teléfono tiene el atributo minOccurs="0", lo que indica que es opcional y puede omitirse sin que el XML sea invalido.

c) El usuario Ataulfo dispone de dos teléfonos móviles y uno fijo, ¿Podrían añadirse todos en el XML?

No, no se podrían añadir los tres números de teléfono, ya que el campo teléfono tiene el atributo maxOccurs="2", lo que quiere decir que solo puede haber hasta dos números de teléfono por contacto.

d) ¿Cuánto correos electrónicos, con máximo y mínimo, hay que introducir?

El campo email tiene minOccurs="1" y maxOccurs="3", lo que significa que cada contacto debe tener al menos 1 correo electrónico y puede registrar hasta 3 correos electrónicos.

e) ¿De que tipo es el campo edad?

El campo edad es de tipo xs:unsignedbyte, lo que significa que solo puede contener valores numéricos entre positivos entre 0 y 255

Ejercicio 9. Enumera las principales diferencias entre DTD y XML

1. Propósito

DTD: Es un esquema que define la estructura y reglas que debe seguir un documento XML.

XML: Es un lenguaje de marcado utilizado para almacenar y transportar datos de forma estructurada.

2. Sintaxis

Dtd: Usa una sintaxis propia y no sigue la estructura de xml.

XML: Sigue una sintaxis basada en etiquetas.

3. Uso

DTD: Se utiliza para validar documentos XML, asegurando que sigan una estructura predefinida.

XML: Se usa para representar datos en una estructura jerárquica y legible tanto por humanos como por máquinas.

4. Validación

DTD: Permite definir que elementos y atributos puede contener un documento XML y en que orden.

XML: Por si solo no dicta reglas de estructura, pero puede validarse contra un DTD o un XSD.

5. Expresividad

DTD: Es mas limitado en cuanto a definir tipos de datos y restricciones.

XML: Permite almacenar y estructurar datos sin restricciones previstas, hasta que se valida con un esquema.