Actividad 12 - Explicación

XSD – Atributos

Voy a intentar explicar cuál sería el proceso para definir el esquema del ejercicio 2 de la Actividad 12. También es válido para el primer ejercicio, solo que en el primero no se define ningún tipo de datos personalizado (simpleType) y se usan solo datos predefinidos.
En este caso no os doy la solución completa, solo un esquema de cómo debe quedar y vosotros tenéis que completarlo con lo que ya sabéis de las actividades anteriores.

Declarar los tipos simples

Lo primero que debemos hacer en el esquema es **declarar los tipos simples** que vamos a utilizar, es decir, tipos de datos a usar ya sea en los elementos o atributos, y que se derivan de tipos primitivos.
En este caso tendremos tres *simpleType*:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:simpleType name="tipo_dni">
        <!-- Aquí definimos la expresión regular para validar un dni -->
        <!-- Como vimos en la actividad anterior -->
    </xs:simpleType>
    <xs:simpleType name="tipo_tipo_documento">
       <!-- Aquí definimos un tipo para los valores que puede tomar el atributo "tipo" dentro del
            elemento "documento". Será un lista de posibles valores o dni o pasaporte (enumeration).
            También como vimos en la actividad anterior -->
    </xs:simpleType>
    <xs:simpleType name="tipo_valor_estado">
        <!-- Aquí definimos un tipo para los valores que puede tomar el atributo "valor" dentro del
            elemento "estado_civil". Será un lista de posibles valores "soltero, casado, divorciado o viudo". (enumeration).
            También como vimos en la actividad anterior -->
    </xs:simpleType>
```

Declarar los tipos complejos.

En segundo lugar vamos a **declarar los tipos complejos** que aparecen en nuestro xml. Recordad que los tipos complejos hacen referencia a los elementos que tienen atributos y/o elementos hijos.
Estos tipos los podemos declarar aparte. Aunque hasta ahora los estábamos declarando de manera anónima al declarar un elemento (dentro de él), es recomendable declararlos aparte y darles un nombre. Es similar a las clases en programación, donde defino sus tipos de datos, atributos y elementos hijos aparte y luego al declarar un elemento le digo que es de ese tipo.
En nuestro esquema también tendremos tres *complexType*:

```
<xs:complexType name="tipo_documento">
    <!-- Aquí definimos un tipo complejo para el elemento "documento" y ¿por qué es complejo? porque tiene un atributo.
         Como tiene atributos y texto dentro se debe declarar como simpleContent -->
    <xs:simpleContent>
        <!-- Como todo simpleContent, quiere decir que la etiqueta va a tener un valor dentro y tenemos que decir
             con un "extension" de que tipo datos será ese valor. Podríamos pensar que es un string, pero ya hemos
             definido un tipo simple antes para el dni con la expresión regular. -->
        <xs:extension base="??????">
            <!-- Dentro del extension lo único que hay que declarar en el atributo que tiene este elemento.
                 En este caso es el atributo "tipo" y será del tipo de datos que hemos defindo antes para este
                 caso "tipo_tipo_documento" -->
            <xs:attribute ???????????/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
<xs:complexType name="tipo_estado_civil">
    <!-- Aquí definimos un tipo complejo para el elemento "estado_civil" y ¿por qué es complejo? porque tiene un atributo.
         Como tiene atributos pero NO texto dentro, NO se declara como simpleContent.
         Solo hay que declarar su atributo y será del tipo de datos que hemos defindo antes para este
         caso "tipo_valor_estado"  -->
    <xs:attribute ???????????/>
</xs:complexType>
<xs:complexType name="tipo_persona">
    <!-- Por último declaramos el tipo para el elemento raíz "persona", que lógicamente en complejo porque tiene atributos
         y además elementos hijos. Como no tiene texto NO se declara como simpleContent.
         Primero: Definimos una secuencia con los tres elementos hijos que tiene (esto está bastante trillado)
                  Cada elmento tendrá su tipo.
                  *nombre: este es de tipo básico string
                  *documento: será del tipo_documento que hemos definido antes
                  *estado_civil: será del tipo_estado_civil que hemos definido antes
         Segundo: Definimos los dos atributos que tiene
                  *codido: de tipo ID y requerido
                  *f_nacimiento: de tipo fecha y opcional -->
</xs:complexType>
```

1
2

1. Este es quizá el concepto más difícil de entender. Como se explica en el vídeo y en los apuntes. Los tipos complejos son aquellos que tienen atributos y/o hijos, es decir, o uno de los dos o ambos. Pues bien, **los tipos complejos que tienen atributos y texto dentro de la etiqueta pero NO tienen hijos se definen como simpleContent y esto hay que indicarlo en el esquema**. Este es el caso del elemento <documento>. Como ese elemento lleva un texto dentro de su etiqueta (en el ejercicio *78767611H*), hay que decir de qué tipo de datos base hereda y para eso se usa la etiqueta *extension* y puede ser de un tipo predefinido (string, integer, date…) o de un tipo que ya hayamos definido nosotros, como pasa aquí. Luego definimos el atributo que también puede ser de un tipo predefinido o de uno definido por nosotros.
2. Este es el caso opuesto al anterior. Los elementos <persona> y <estado_civil> son también complejos, porque tiene atributos y/o hijos, pero como NO tiene texto en la etiqueta en sí, se definen como *complexContent*, pero este caso está implícito, es decir, es el caso por defecto, y por tanto **estos tipos NO hay que declararlos como complexType** (no hay que poner la etiqueta), ya que se da por asumido.

Declarar los elementos de XML.

Por último hay que declarar el elemento raíz. Hasta ahora solo habíamos definido los tipos de datos que vamos a usar (simpleType) y los tipos de elementos complejos que tendrá nuestro xml (complexType), pero no hemos definido ningún elemento en sí.
Por tanto ahora definimos nuestro elemento base **persona**, que es de tipo “tipo_persona” y en ese tipo ya tiene definido todo lo demás.
Pensad que es parecido a lo que en programación orientada a objetos es definir una clase y luego declarar un objeto de esa clase. En este caso definimos un tipo complejo y luego declaramos un elemento que ese de ese tipo.

```
    <xs:element name="persona" type="tipo_persona" />
</xs:schema>
```

Y ya estaría completo nuestro esquema. Se que la primera vez puede parecer complicado, pero una vez que se hace bien y se repite un par de veces, es sencillo. Repasar de nuevo el vídeo 6 sobre atributos y leed bien este documento. Si aún os quedan dudas, me las preguntais sin problema.