XSD

What is XSD?

XSD or XML Schema Definition is another way, compared to DTD, to define legal building blocks within a XML document. Using XML, different groups can agree on a standardised XML format for sending and receiving data. The file extension for XSD is **.xsd**

The format stays very similar to XML.

Quick Example:

XML document.

```
<?xml version="1.0" ?>
<student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=" student.xsd">
     <firstName>John</firstName>
     <lastName>Smith</lastName>
     <email>jsmith@gmail.com</email>
     <mobile>0211223344</mobile>
</student>
```

XSD file.

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:element name="student">
      <xsd:complexType>
           <xsd:sequence>
                <xsd:element name="firstName" type="xsd:string"/>
                <xsd:element name="lastName" type="xsd:string"/>
                <xsd:element name="email" type="xsd:string"/>
                <xsd:element name="mobile" type="xsd:string"/>
           </xsd:sequence>
      </xsd:complexType>
      </xsd:element>
</xsd:schema>
```

XSD file explained:

Opening tag:
This must be at the start of each XSD document, this also contains a link to the schema that is being used. This link is stored as a variable and can be used throughout the file for.

Elements:
Elements must be defined in an element tag with a name, the type can also be listed if it is part of a sequence.

Complex vs Simple types:
After defining an element, you must state if it is a complexType or a simpleType.

- A complexType is an element that contains other elements or other attributes.
- A simpleType is an element that contains no attributes and no elements.

Sequence:
A sequence contains multiple simpleTypes that make up a parent element. This also must be defined before listing each individual element.

Element attributes

Here we will demonstrate an attribute within a complexType element.

```
<scan schedule="hourly">
     <start>2018-06-20T13:00:00</start>
     <finish>2018-06-20T13:01:47</finish>
     <virusFound>true</virusFound>
</scan>

<xsd:element name="scan">
 <xsd:complexType>
     <xsd:sequence>
         <xsd:element name="start" type="xsd:dateTime" />
         <xsd:element name="finish" type="xsd:dateTime" />
         <xsd:element name="virusFound" type="xsd:boolean" />
     </xsd:sequence>
     <xsd:attribute name="schedule" type="xsd:string" />
 </xsd:complexType>
</xsd:element>
```

Attribute Declaration:
The attribute must be declared after the sequence child elements of the parent have been defined. Within this tag we must note the attribute name and type.

Here we will demonstrate attributes within a simpleType

```
<price promotionCode="FAMILYDEAL">39.50</price>

<xsd:element name="price">
    <xsd:complexType>
       <xsd:simpleContent>
          <xsd:extension base="xsd:decimal">
             <xsd:attribute name="promotionCode" type="xsd:string" />
          </xsd:extension>
       </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
```

SimpleContent:
Notice that the simpleContent is defined within a complexType element. The content is then named within the simpleContent tag as extension, however it does not need a name as this is already defined.

Attribute:
The attribute is then defined after the contents of the element, making sure to define a name and type.

Simple type with a restriction

When using a restriction, the data contents can only have specific values. Here are some simple examples to look at.

```
<xsd:element name="grade">
     <xsd:simpleType>
         <xsd:restriction base="xsd:string">
             <xsd:enumeration value="A"/>
             <xsd:enumeration value="B"/>
             <xsd:enumeration value="C"/>
             <xsd:enumeration value="D"/>
         </xsd:restriction>
     </xsd:simpleType>
</xsd:element>
```

The element grade can only contain the values A,B,C,D

Numerical example:

```
<xsd:element name="mark">
     <xsd:simpleType>
         <xsd:restriction base="xsd:integer">
             <xsd:minInclusive value="0"/>
             <xsd:maxInclusive value="100"/>
         </xsd:restriction>
     </xsd:simpleType>
</xsd:element>
```

The element mark can only contain numbers 0 to 100.

Writing Complex XSD Documents

When writing complex XSD documents you can quickly feel overwhelmed and congested if you write the whole document start to finish. Instead, you should start with the parent element, making sure to close the tag, before heading over to child elements.

Here is a more complex example:

XML:

```
<?xml version="1.0" ?>
<studentList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="studentList.xsd">
     <student>
         <firstName>John</firstName>
         <lastName>Smith</lastName>
         <email>jsmith@gmail.com</email>
     </student>
     <student>
         <firstName>Mary</firstName>
         <lastName>Jane</lastName>
         <email>mjane@gmail.com</email>
         </student>
</studentList>
```

XSD:

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="studentList">
       <xsd:complexType>
          <xsd:sequence>
             <xsd:element name="student" minOccurs="0"
maxOccurs="unbounded">
                <xsd:complexType>
                   <xsd:sequence>
                      <xsd:element name="firstName" type="xsd:string"/>
                      <xsd:element name="lastName" type="xsd:string"/>
                      <xsd:element name="email" type="xsd:string"/>
                   </xsd:sequence>
                </xsd:complexType>
             </xsd:element>
          </xsd:sequence>
       </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

Note that when dealing with multiple elements of the same type it is smart to identify a minimum and maximum occurrence bound.

That's it ! Try it out and see how you go :).