Student Online Teaching Advice Notice

CSU22041: Information Management I

Document Type Definition (DTD)

… an **art of making information accessible**.

2020-2021

Gaye Stephens gaye.stephens@tcd.ie

XML Example

```
<?xml version='1.0' encoding='ISO-8859-1' standalone='no' ?>
<doc type="book" isbn="1-56592-796-9" xml:lang="en">
  <title>A Guide to XML</title>
  <author>Norman Walsh</author>
  <chapter>
    <title>What Do XML Documents Look Like?</title>
    <paragraph>If you are ...</paragraph>
      <item>
        <paragraph>The document begins ...</paragraph>
      </item>
      <item>
        <paragraph>Empty elements have ...</paragraph>
        <paragraph>In a very ..</paragraph>
      </item>
    <section>...</section>
    ...
  </chapter>
  <chapter>...</chapter>
</doc>
```

VALID XML

DOCUMENT TYPE DEFINITION (DTD)

What is a DTD?

- Document Type Definition,
- Defines structure/model of XML documents
  - Elements and Cardinality
  - Attributes
  - Aggregation
- Defines default ATTRIBUTE values
- Defines ENTITIES
- Stored in a plain text file and referenced by an XML document **(external)**
- Alternatively a DTD can be placed in the XML document itself **(internal)**

Example DTD

```
<?xml version="1.0" ?>
<database>
<person age='34'>
    <name>
        <title> Mr </title>
        <firstname> John </firstname>
        <firstname> Paul </firstname>
        <surname> Murphy </surname>
    </name>
    <hobby> Football </hobby>
    <hobby> Racing </hobby>
</person>

<person >
    <name>
        <firstname> Mary </firstname>
        <surname> Donnelly </surname>
    </name>
</person>
</database>
```

```
<!DOCTYPE database [

<!ELEMENT database (person*)>

<!ELEMENT person (name,hobby*)>
<!ATTLIST person age CDATA
   #IMPLIED>

<!ELEMENT name (title?, firstname+,
   surname)>

<!ELEMENT hobby (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT surname (#PCDATA)>

]>
```

- Syntax for occurrences of elements in DTDs
  - ?: zero-or-one
  - +: one-or-more
  - *: zero-or-more

Element Type Declaration

- Define occurrences of elements

  ?: zero-or-one

  +: one-or-more

  *: zero-or-more

```
<!ELEMENT doc
   (title, author+, editor?,
   chapter+, appendix*)>

<!ELEMENT chapter
   (title,
    (section+ | paragraph+))>

<!ELEMENT paragraph
   (#PCDATA)>

………
```

Entity Declaration

- Internal entities
  - Built-in
- External entities
  - References to a file (text, images etc.)
- Parameter entities
  - Used inside DTDs

```
<!ENTITY author
   "Norman Walsh, Sun Corp.">
```

```
<!ENTITY copyright
   SYSTEM "copyright.xml">
```

```
<!ENTITY % part
   "(title?, (paragraph |
   section)*)">
```

Attribute List Declaration

- Define type of attribute
  - CDATA
  - ID
  - ENTITY
  - ………
- Define default values of attributes
  - #REQUIRED
  - #IMPLIED
  - #FIXED
  - A list of values with default selection

```
<!ATTLIST person
   ssn ID #IMPLIED>


<!ATTLIST adult
   age CDATA #REQUIRED>


<!ATTLIST mml
   version CDATA #FIXED '1.0'>


<!ATTLIST person
    sex (m | f) #REQUIRED>


<!ATTLIST day
   temperature (l | m | h) "l">
```

Simple DTD Example

- Syntax for occurrences of elements in DTDs
  - ?: zero-or-one
  - +: one-or-more
  - *: zero-or-more

```
<!DOCTYPE doc[
<!ENTITY % part "(title?, (paragraph | section)*)">

<!ELEMENT doc (title, author+, chapter+, appendix*)>
<!ATTLIST doc type (book | article) "book“
              isbn CDATA #REQUIRED>

<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT chapter %part;>
<!ELEMENT appendix %part;>
<!ELEMENT section %part;>
<!ELEMENT paragraph (#PCDATA | url | ol)*>
<!ATTLIST paragraph type CDATA #IMPLIED>
<!ELEMENT ol (item+)>
<!ELEMENT item (sentence+)>
<!ELEMENT sentence (#PCDATA)>
<!ELEMENT url (#PCDATA)>
]>
```

Note on UML Attribute mapping

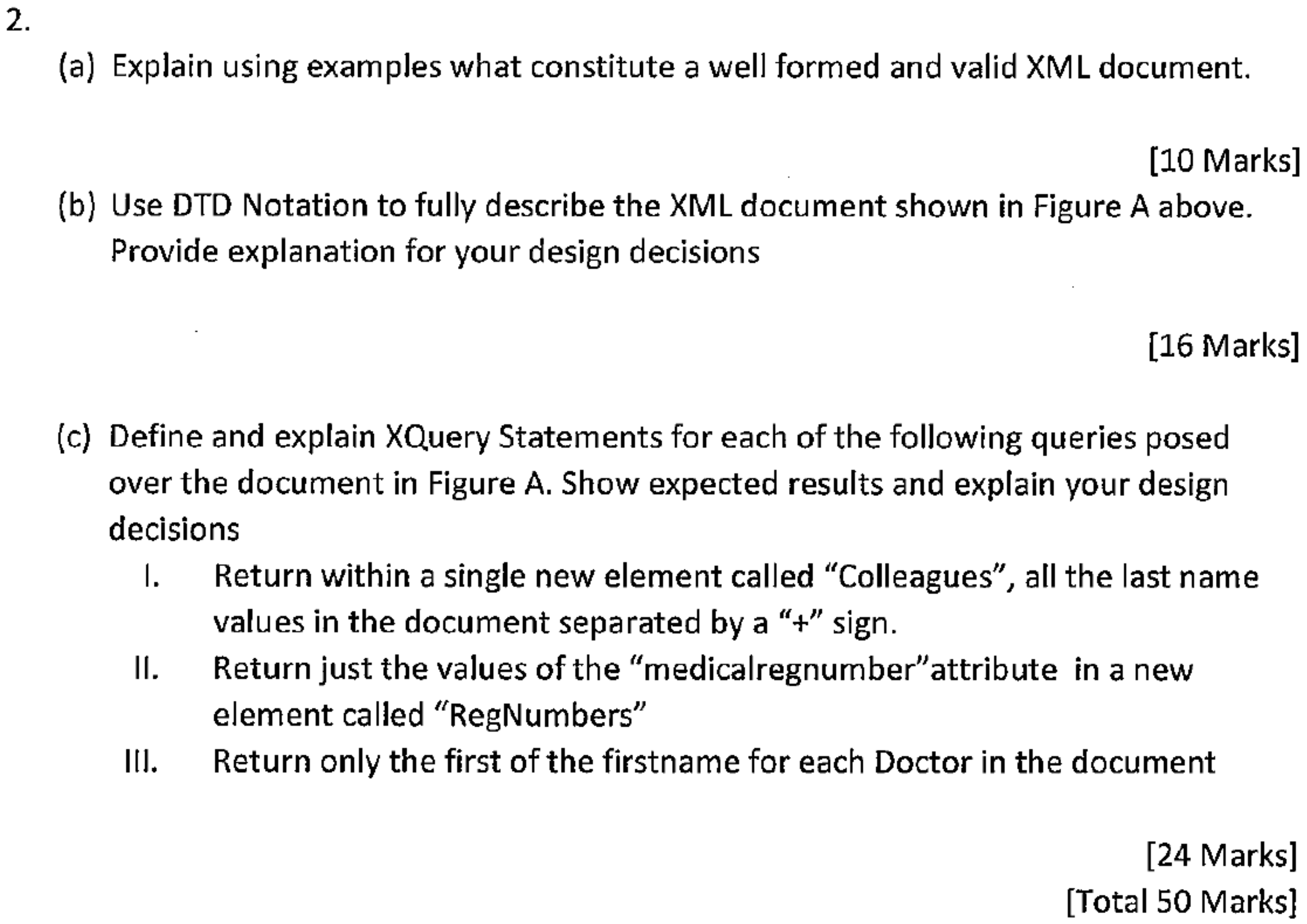
- UML attribute **can** be represented as XML attribute
  - Note can be inappropriate for String attributes which are either large or contain whitespaces, tabs (as these are stripped out of XML attributes by XML parsers)

```
<Product>
      <CatalogItem.name>
            Wizard 4000
       </CatalogItem.name>
       <CatalogItem.sku>
             p123
        </CatalogItem.sku>
        <CatalogItem.listPrice>
                <Money>
                <Money.currency>USD</Money.currency>
                <Money.amount>2000</Money.amount>
                 </Money> </Catalog.listPrice>
   </Product>
```

```
<Product sku="123" >
      <CatalogItem.name>
            Wizard 4000
       </CatalogItem.name>
        <CatalogItem.listPrice>
                <Money currency="USD" amount="2000"/>
         </Catalog.listPrice>
   </Product>
```

https://www.w3schools.com/xml/xml_dtd_el_vs_attr.asp

Past XML Exam Question



2.

(a) Explain using examples what constitute a well formed and valid XML document.

[10 Marks]

(b) Use DTD Notation to fully describe the XML document shown in Figure A above. Provide explanation for your design decisions

[16 Marks]

(c) Define and explain XQuery Statements for each of the following queries posed over the document in Figure A. Show expected results and explain your design decisions

I. Return within a single new element called "Colleagues", all the last name values in the document separated by a "+" sign.

II. Return just the values of the "medicalregnumber"attribute in a new element called "RegNumbers"

III. Return only the first of the firstname for each Doctor in the document

[24 Marks]
[Total 50 Marks]

2017

For information purposes- not included in exam or assignment

XML NAMESPACES & XML SCHEMA

What are XML Schemas?

- W3C Recommendation, Part 0: Primer
  - Part 1: Structures
  - Part 2: Datatypes
- DTDs use a non-XML syntax and have a number of limitations
  - no namespace support
  - lack of data-types
- XML Schemas are an alternative to DTDs
- Supports simple/complex data-types
- https://www.w3schools.com/xml/xml_schema.asp

Why use XML Schemas?

- Uses an XML syntax
- Supports simple and complex data-types such as user-defined types
- An XML document and its contents can be validated against a Schema
- Can validate documents containing multiple namespaces
- Schemas are more powerful than DTDs and will eventually replace DTDs

Named Types – simple (can contain “only text”)

DTD

```
<!ELEMENT birthday(#PCDATA)>
```

XML Schema

```
<xsd:element name=” birthday" type="xsd:date"/>
```

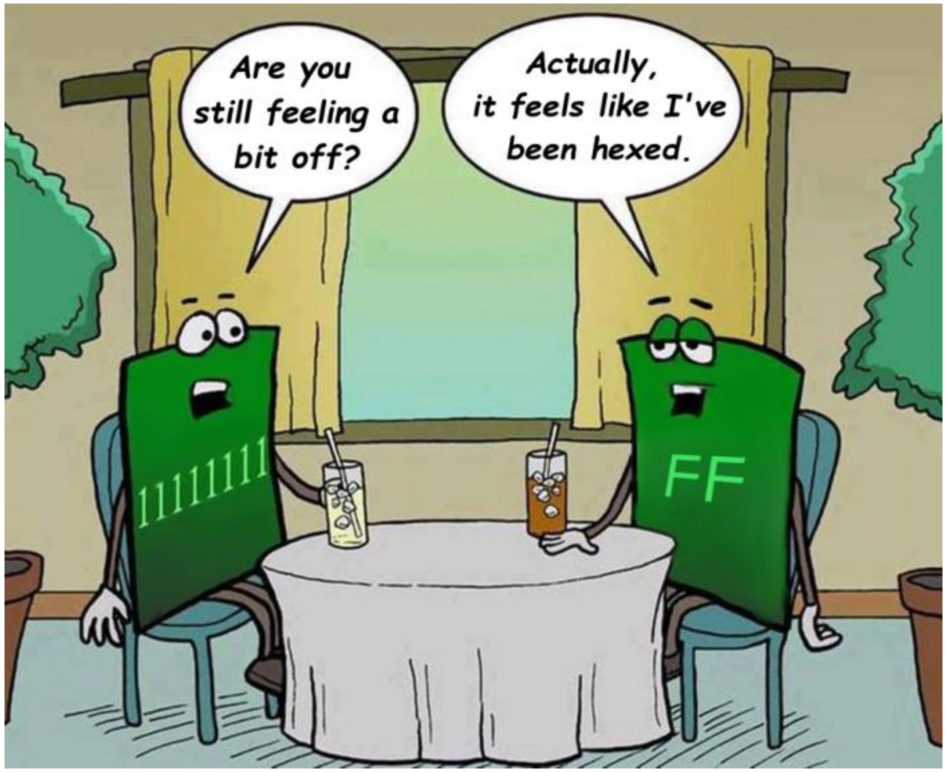
XML doc. Instance

```
<birthday>01 March 2001</birthday>
```

Summary

- XML Vocabularies are defined using
  - DTD
  - XSD
- DTDs/XSDs used to validate XML documents
- XSD – more powerful than DTDs
  - Supports simple and complex data-types such as user-defined types
  - Can validate documents containing multiple namespaces

That's All Folks Thank You for Listening


Are you still feeling a bit off?
Actually, it feels like I've been hexed.
11111111
FF

For information purposes- not included in exam or assignment

XML NAMESPACES & XML SCHEMA

Named Types – simple (can contain "only text")

DTD

```
<!ELEMENT birthday(#PCDATA)>
```

XML Schema

```
<xsd:element name=" birthday" type="xsd:date"/>
```

XML doc. Instance

```
<birthday>01 March 2001</birthday>
```

Named Types – complex(can contain elements and attributes)

DTD

```
<!ELEMENT student_name (firstname, lastname)>
```

XML Schema

```
<xsd:complexType name="namePerson">
   <xsd:sequence>
     <xsd:element name="firstname" type="xsd:string"/>
     <xsd:element name="lastname" type="xsd:string/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name = "student_name" type="namePerson"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Michael</firstname>
   <lastname>Porter</lastname>
</student_name>
```

Simple Type - Restriction

XML Schema

```
<simpleType name='celsiusBodyTemp'>
  <restriction base='decimal'>
    <totalDigits value='4'/>
    <fractionDigits value='1'/>
    <minInclusive value='36.4'/>
    <maxInclusive value='40.5'/>
  </restriction>
</simpleType>
<xsd:element name="temp" type="celsiusBodyTemp"/>
```

XML doc. Instance

```
<temp>37.2</temp>
```

Simple Type - Enumeration

XML Schema

```
<xsd:simpleType name="weekday">
   <xsd:restriction base="xsd:string">
     <xsd:enumeration value="Sunday"/>
     <xsd:enumeration value="Monday"/>
     <xsd:enumeration value="Tuesday"/>
     [...]
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="delivery" type="weekday"/>
```

XML doc. Instance

```
<delivery>Tuesday</delivery>
```

Complex Type - Cardinalities

DTD

```
<!ENTITY % fullname "title?, firstname*, lastname">
<!ELEMENT student_name (%fullname;)>
```

XML Schema

```
<xsd:complexType name="fullname">
  <xsd:sequence>
    <xsd:element name="title" minOccurs="0"/>
    <xsd:element name="firstname" minOccurs="0"
                 maxOccurs="unbounded"/>
    <xsd:element name="lastname"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name=” student_name" type="fullname"/>
```

XML doc. Instance

```
<student_name>
  <firstname>Michael</firstname>
  <firstname>Jason</firstname>
  <lastname>Porter</lastname>
</student_name>
```

Complex Type – Derived Type by extension

DTD

```
<!ENTITY % name "title?, firstname*, lastname">
<!ELEMENT student_name (%name;, maidenname?)>
```

XML Schema

```
<xsd:complexType name="fullnameExt">
   <xsd:complexContent>
     <xsd:extension base="fullname">
       <xsd:sequence>
         <xsd:element name="maidenname" minOccurs="0"/>
       </xsd:sequence>
     </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:element name=" student_name" type="fullnameExt"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Jane</firstname>
   <lastname>Porter</lastname>
   <maidenname>Hughes</maidenname>
</student_name>
```

Complex Type – Derived Type by Restriction

XML Schema

```
<xsd:complexType name="simpleName">
   <xsd:complexContent>
     <xsd:restriction base="fullname">
       <xsd:sequence>
         <xsd:element name="title" maxOccurs="0"/>
         <xsd:element name="firstname" minOccurs="1"/>
         <xsd:element name="lastname"/>
       </xsd:sequence>
     </xsd:restriction>
   </xsd:complexContent>
</xsd:complexType>
```

XML doc. Instance

```
<name>
  <firstname>Jane</firstname>
  <lastname>Porter</lastname>
</name>
```

Structure - Sequence

DTD

```
<!ELEMENT student_name (title?, firstname*, lastname)>
```

XML Schema

```
<xsd:complexType name="fullname">
   <xsd:sequence>
     <xsd:element name="title" minOccurs="0"/>
     <xsd:element name="firstname" minOccurs="0"
                  maxOccurs="unbounded"/>
     <xsd:element name="lastname"/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name=” student_name" type="fullname"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Michael</firstname>
   <firstname>Jason</firstname>
   <lastname>Porter</lastname>
</student_name>
```

Structure - Choice

DTD

```
<!ELEMENT pay (product, number, (cash | cheque))>
```

XML Schema

```
<xsd:complexType name="payment">
   <xsd:sequence>
     <xsd:element ref="product"/>
     <xsd:element ref="number"/>
     <xsd:choice>
       <xsd:element ref="cash"/>
       <xsd:element ref="cheque"/>
     </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name="pay" type="payment"/>
```

XML doc. Inst.

```
<pay>
   <product>Ericsson Telefon MD110</product>
   <number>1544-198-J</number>
   <cash>EUR150</cash>
</pay>
```

Attributes-

DTD

```
<!ELEMENT greeting (#PCDATA)>
<!ATTLIST greeting language CDATA "English">
```

XML Schema

```
<xsd:element name="greeting">
   <xsd:complexType>
     <xsd:simpleContent>
       <xsd:extension base="xsd:string">
         <xsd:attribute name="language" type="xsd:string"/>
       </xsd:extension>
     </xsd:simpleContent>
   </xsd:complexType>
</xsd:element>
```

XML doc. Instance

```
<greeting language="German">Hello!</greeting>
```

Attribute Groups

DTD

```
<!ELEMENT img EMPTY>
<!ATTLIST img src CDATA #REQUIRED
          width CDATA #IMPLIED
          height CDATA #IMPLIED>
```

XML Schema

```
<xsd:attributeGroup name="imgAttributes">
   <xsd:attribute name="src" type="xsd:string" use="required"/>
   <xsd:attribute name="width" type="xsd:integer"/>
   <xsd:attribute name="height" type="xsd:integer"/>
</xsd:attributeGroup>


<xsd:element name="img">
   <xsd:complexType>
     <xsd:attributeGroup ref="imgAttributes"/>
   <xsd:complexType>
</xsd:element>
```

XML Inst.

```
<img src="XMLmanager.gif" width="60"/>
```

Mixed Content

DTD

```
<!ELEMENT p (#PCDATA | b | i)*>
<!ELEMENT b (#PCDATA)>
```

XML Schema

```
<xsd:complexType name="bolditalicText" mixed="true">
   <xsd:choice minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="b" />
     <xsd:element ref="i" />
   </xsd:choice>
</xsd:complexType>


<xsd:element name="p" type="bolditalicText"/>
```

XML doc. Instance

```
<p>This is <b>bold</b> and <i>italic</i> text</p>
```

Empty Element

DTD

```
<!ELEMENT img EMPTY>
<!ATTLIST src CDATA #REQUIRED>
```

XML Schema

```
<xsd:element name="img">
   <xsd:complexType>
     <xsd:attribute name="src" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>
```

XML doc. Instance

```
<img src="XMLmanager.gif"/>
```

XML Schema Example

```
<?xml version="1.0" encoding="utf-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
  <xsd:element name="book">

    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="title" type="xsd:string"/>
        <xsd:element name="author" type="xsd:string"/>
        <xsd:element name="character" type="xsd:string"
                     minOccurs="0" maxOccurs="unbounded">
        </xsd:element>
      </xsd:sequence>

      <xsd:attribute name="isbn" type="xsd:string"/>
    </xsd:complexType>

  </xsd:element>
</xsd:schema>
```

Named Types – complex(can contain elements and attributes)

DTD

```
<!ELEMENT student_name (firstname, lastname)>
```

XML Schema

```
<xsd:complexType name="namePerson">
   <xsd:sequence>
     <xsd:element name="firstname" type="xsd:string"/>
     <xsd:element name="lastname" type="xsd:string/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name = "student_name" type="namePerson"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Michael</firstname>
   <lastname>Porter</lastname>
</student_name>
```

Simple Type - Restriction

XML Schema

```
<simpleType name='celsiusBodyTemp'>
  <restriction base='decimal'>
    <totalDigits value='4'/>
    <fractionDigits value='1'/>
    <minInclusive value='36.4'/>
    <maxInclusive value='40.5'/>
  </restriction>
</simpleType>
<xsd:element name="temp" type="celsiusBodyTemp"/>
```

XML doc. Instance

```
<temp>37.2</temp>
```

Simple Type - Enumeration

XML Schema

```
<xsd:simpleType name="weekday">
   <xsd:restriction base="xsd:string">
     <xsd:enumeration value="Sunday"/>
     <xsd:enumeration value="Monday"/>
     <xsd:enumeration value="Tuesday"/>
     [...]
   </xsd:restriction>
</xsd:simpleType>
<xsd:element name="delivery" type="weekday"/>
```

XML doc. Instance

```
<delivery>Tuesday</delivery>
```

Complex Type - Cardinalities

DTD

```
<!ENTITY % fullname "title?, firstname*, lastname">
<!ELEMENT student_name (%fullname;)>
```

XML Schema

```
<xsd:complexType name="fullname">
  <xsd:sequence>
    <xsd:element name="title" minOccurs="0"/>
    <xsd:element name="firstname" minOccurs="0"
                 maxOccurs="unbounded"/>
    <xsd:element name="lastname"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name=” student_name" type="fullname"/>
```

XML doc. Instance

```
<student_name>
  <firstname>Michael</firstname>
  <firstname>Jason</firstname>
  <lastname>Porter</lastname>
</student_name>
```

Complex Type – Derived Type by extension

DTD

```
<!ENTITY % name "title?, firstname*, lastname">
<!ELEMENT student_name (%name;, maidenname?)>
```

XML Schema

```
<xsd:complexType name="fullnameExt">
   <xsd:complexContent>
     <xsd:extension base="fullname">
       <xsd:sequence>
         <xsd:element name="maidenname" minOccurs="0"/>
       </xsd:sequence>
     </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:element name=” student_name" type="fullnameExt"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Jane</firstname>
   <lastname>Porter</lastname>
   <maidenname>Hughes</maidenname>
</student_name>
```

Complex Type – Derived Type by Restriction

XML Schema

```
<xsd:complexType name="simpleName">
   <xsd:complexContent>
     <xsd:restriction base="fullname">
       <xsd:sequence>
         <xsd:element name="title" maxOccurs="0"/>
         <xsd:element name="firstname" minOccurs="1"/>
         <xsd:element name="lastname"/>
       </xsd:sequence>
     </xsd:restriction>
   </xsd:complexContent>
</xsd:complexType>
```

XML doc. Instance

```
<name>
   <firstname>Jane</firstname>
   <lastname>Porter</lastname>
</name>
```

Structure - Sequence

DTD

```
<!ELEMENT student_name (title?, firstname*, lastname)>
```

XML Schema

```
<xsd:complexType name="fullname">
   <xsd:sequence>
      <xsd:element name="title" minOccurs="0"/>
      <xsd:element name="firstname" minOccurs="0"
                   maxOccurs="unbounded"/>
      <xsd:element name="lastname"/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name=" student_name" type="fullname"/>
```

XML doc. Instance

```
<student_name>
   <firstname>Michael</firstname>
   <firstname>Jason</firstname>
   <lastname>Porter</lastname>
</student_name>
```

Structure - Choice

DTD

```
<!ELEMENT pay (product, number, (cash | cheque))>
```

XML Schema

```
<xsd:complexType name="payment">
   <xsd:sequence>
     <xsd:element ref="product"/>
     <xsd:element ref="number"/>
     <xsd:choice>
       <xsd:element ref="cash"/>
       <xsd:element ref="cheque"/>
     </xsd:choice>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name="pay" type="payment"/>
```

XML doc. Inst.

```
<pay>
   <product>Ericsson Telefon MD110</product>
   <number>1544-198-J</number>
   <cash>EUR150</cash>
</pay>
```

Attributes-

DTD

```
<!ELEMENT greeting (#PCDATA)>
<!ATTLIST greeting language CDATA "English">
```

XML Schema

```
<xsd:element name="greeting">
   <xsd:complexType>
     <xsd:simpleContent>
       <xsd:extension base="xsd:string">
         <xsd:attribute name="language" type="xsd:string"/>
       </xsd:extension>
     </xsd:simpleContent>
   </xsd:complexType>
</xsd:element>
```

XML doc. Instance

```
<greeting language="German">Hello!</greeting>
```

Attribute Groups

DTD

```
<!ELEMENT img EMPTY>
<!ATTLIST img src CDATA #REQUIRED
              width CDATA #IMPLIED
              height CDATA #IMPLIED>
```

XML Schema

```
<xsd:attributeGroup name="imgAttributes">
   <xsd:attribute name="src" type="xsd:string" use="required"/>
   <xsd:attribute name="width" type="xsd:integer"/>
   <xsd:attribute name="height" type="xsd:integer"/>
</xsd:attributeGroup>


<xsd:element name="img">
   <xsd:complexType>
     <xsd:attributeGroup ref="imgAttributes"/>
   <xsd:complexType>
</xsd:element>
```

XML Inst.

```
<img src="XMLmanager.gif" width="60"/>
```

Mixed Content

DTD

```
<!ELEMENT p (#PCDATA | b | i)*>
<!ELEMENT b (#PCDATA)>
```

XML Schema

```
<xsd:complexType name="bolditalicText" mixed="true">
   <xsd:choice minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element ref="b" />
     <xsd:element ref="i" />
   </xsd:choice>
</xsd:complexType>


<xsd:element name="p" type="bolditalicText"/>
```

XML doc. Instance

```
<p>This is <b>bold</b> and <i>italic</i> text</p>
```

Empty Element

DTD

```
<!ELEMENT img EMPTY>
<!ATTLIST src CDATA #REQUIRED>
```

XML Schema

```
<xsd:element name="img">
   <xsd:complexType>
     <xsd:attribute name="src" type="xsd:string"/>
   </xsd:complexType>
</xsd:element>
```

XML doc. Instance

```
<img src="XMLmanager.gif"/>
```

XML Schema Example

```
<?xml version="1.0" encoding="utf-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
  <xsd:element name="book">

    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="title" type="xsd:string"/>
        <xsd:element name="author" type="xsd:string"/>
        <xsd:element name="character” type="xsd:string"
                     minOccurs="0" maxOccurs="unbounded">
        </xsd:element>
      </xsd:sequence>

      <xsd:attribute name="isbn" type="xsd:string"/>
    </xsd:complexType>

  </xsd:element>
</xsd:schema>
```