

1 ++++++Defining Complex Type Schema Lesson 6+++++

2

3

4

5 1.) Deriving "Anonymous" Complex Types

6

7

```
8 <xs:element name="year_built">
9   <xs:complexType>
10    <xs:simpleContent><!--no children, just data-->
11    <xs:extension base="xs:positiveInteger"><!--must be a positive integer-->
12      <!--need complex type to define attribute-->
13      <xs:attribute name="era" type="xs:string"/>
14    </xs:extension>
15  </xs:simpleContent>
16 </xs:complexType>
17 </xs:element>
```

18

19 **Example of use in XML document**

20

```
21 <year_built era="BC">282</year_built>
```

22

23

24 2.) Deriving **Named** Complex Types

25

26 **Create the definition**

```
27 <!--I can reuse this definition by calling the name-->
```

```
28 <xs:complexType name="yearType">
29   <xs:simpleContent>
30     <xs:extension base="xs:positiveInteger">
31       <xs:attribute name="era" type="xs:string"/>
32     </xs:extension>
33   </xs:simpleContent>
34 </xs:complexType>
```

35

36 **Reference the definition**

```
37 <xs:complexType name="historyType">
38   <xs:sequence>
39     <xs:element name="year_built" type="yearType"/>
40     <!--"yearType" references the above definition-->
41     <xs:element name="year_destroyed" type="yearType"/>
42   </xs:sequence>
43 </xs:complexType>
```

44

45 **Here is how the Derived "named" definition looks in XML**

```
46 <year_built era="BC">282</year_built>
47 <year_destroyed era="BC">226</year_destroyed>
```

48

49

50

51 3.) Anonymous Complex Types with one Child

52

```
53 <xs:element name="ancient_wonders">
54   <xs:complexType>
```

```

55         <xs:complexContent>
56             <xs:restriction base="xs:anyType">
57                 <xs:sequence>
58                     <xs:element name="wonder" type="wonderType"/>
59                 </xs:sequence>
60             </xs:restriction>
61         </xs:complexContent>
62     </xs:complexType>
63 </xs:element>

```

Same thing, but without restriction and base which are not needed (even with one child, you must use <xs:sequence>)

Preferred:

```

68 <xs:element name="ancient_wonders">
69     <xs:complexType>
70         <xs:sequence>
71             <xs:element name="wonder" type="wonderType"/>
72         </xs:sequence>
73     </xs:complexType>
74 </xs:element>

```

4.) Deriving Named Complex Types ****Sequence**** of Children

```

78 <xs:complexType name="wonderType">
79     <xs:sequence>
80         <xs:element name="name" type="nameType"/>
81         <xs:element name="location" type="xs:string"/>
82         <xs:element name="height" type="heightType"/>
83         <xs:element name="history" type="historyType"/>
84         <xs:element name="main_image" type="imageType"/>
85         <xs:element name="source" type="sourceType"/>
86     </xs:sequence>
87 </xs:complexType>

```

<!--gets referenced by element "name"-->

```

90 <xs:complexType name="nameType">
91     <xs:simpleContent>
92         <xs:extension base="xs:string">
93             <xs:attribute name="language" type="xs:string"/>
94         </xs:extension>
95     </xs:simpleContent>
96 </xs:complexType>

```

<!--gets referenced by element "height"-->

```

99 <xs:complexType name="heightType">
100     <xs:simpleContent>
101         <xs:extension base="xs:nonNegativeInteger">
102             <xs:attribute name="units" type="xs:string"/>
103         </xs:extension>
104     </xs:simpleContent>
105 </xs:complexType>

```

5.) Allowing Child Elements to Appear in ****Any**** Order

```

109 <xs:complexType name="historyType">
110   <!--use xs:all to allow any order-->
111   <xs:all>
112     <xs:element name="year_built" type="yearType"/>
113     <xs:element name="year_destroyed" type="yearType"/>
114     <xs:element name="how_destroyed" type="destrType"/>
115     <xs:element name="story" type="storyType"/>
116   </xs:all>
117 </xs:complexType>

```

118
119

6.) Creating Child Elements as a set of ****Choices****

```

121 <!--use xs:choice to allow any order-->
122 <xs:complexType name="wonderType">
123   <xs:sequence>
124     <xs:element name="name"
125       type="nameType"/>
126     **THIS **
127     <xs:choice>
128       <xs:element name="location"
129         type="xs:string"/>
130     <xs:sequence>
131       **OR THIS**
132       <xs:element name="city"
133         type="xs:string"/>
134       <xs:element name="country"
135         type="xs:string"/>
136     </xs:sequence>
137   </xs:choice>
138   ...
139 </xs:sequence>
140 </xs:complexType>

```

141
142

XML Example

```

143 <wonder>
144   <name language="English"> Colossus of Rhodes</name>
145   <location>Rhodes, Greece</location>
146   ...
147 </wonder>
148
149
150 <wonder>
151   <name language="English">Colossus of Rhodes</name>
152   <city>Rhodes</city>
153   <country>Greece</country>
154   ...
155 </wonder>

```

156
157

7.) Defining Elements to Contain *****Text***** Only

159

```

160 <xs:complexType name="yearType">
161   <xs:simpleContent>
162     <xs:extension base="xs:positiveInteger">

```

```

163         <xs:attribute name="era" type="xs:string"/>
164     </xs:extension>
165 </xs:simpleContent>
166 </xs:complexType>
167
168 <xs:complexType name="historyType">
169     <xs:sequence>
170         <xs:element name="year_built" type="yearType"/>
171         <xs:element name="year_destroyed" type="yearType"/>
172         <xs:element name="how_destroyed" type="destrType"/>
173         <xs:element name="story" type="storyType"/>
174     </xs:sequence>
175 </xs:complexType>
176
177     Example XML
178     <year_built era="BC">282</year_built>     ***VALID***
179
180     <year_built era="BC">long ago</year_built> ***INVALID***
181

```

182 8.) Defining Elements to be ****EMPTY****

```

183 <xs:complexType name="sourceType">
184     <xs:attribute name="sectionid" type="xs:positiveInteger"/>
185     <xs:attribute name="newspaperid" type="xs:positiveInteger"/>
186 </xs:complexType>
187

```

188 **Example XML**

```

189 <source sectionid="101" newspaperid="21"/>
190
191
192

```

193 9.) Defining Elements to be have *****MIXED Content*****

```

194 <xs:complexType name="story" mixed="true">
195     <xs:sequence>
196         <xs:element name="para" maxOccurs="unbounded">
197             <xs:complexType/>
198         </xs:element>
199     </xs:sequence>
200 </xs:complexType>
201

```

202 *****The XML Schema attribute maxOccurs="unbounded" allows for unlimited number**
203 **of para elements within the story element.*****

204 **Example XML**

```

205 <story>
206 In 294 BC, the people of the island
207 of Rhodes began building a colossal
208 statue of the sun god Helios. They
209 believed that it was because of his
210 blessings that they were able to
211 withstand a long siege on the
212 island and emerge victorious.
213 <para/>
214 The Colossus was built with bronze,
215 reinforced with iron, and weighted
216 with stones. While it is often

```

217 depicted straddling Mandrákion
 218 harbor, this is now considered
 219 technically impossible; and
 220 therefore, it likely stood beside
 221 the harbor.

222 <para/>

223 The statue was toppled by an
 224 earthquake in 226 BC. ...

225 </story>

226

227 9a.) Another example of mixed content with an attribute that indicates the
 number of times it can occur

228 <!--mixed content named complex type-->

229 <xs:complexType name="notesType" mixed="true">

230 <xs:sequence minOccurs="1">

231 <xs:element name="major" type="xs:string">

232 </xs:element>

233 </xs:sequence>

234 </xs:complexType>

235

236

237 10.) Deriving Complex Types from Existing Complex Types

238

239 <xs:complexType name="wonderType">

240 <xs:sequence>

241 <xs:element name="name" type="nameType"/>

242 <xs:element name="location" type="xs:string"/>

243 <xs:element name="height" type="heightType"/>

244 <xs:element name="history" type="historyType"/> <!--reference-->

245 <xs:element name="main_image" type="imageType"/>

246 <xs:element name="source" type="sourceType"/>

247 </xs:sequence>

248 </xs:complexType>

249

250 <xs:complexType name="historyType">

251 <xs:sequence>

252 <!--all of these get added as children to the element "history" above-->

253 <xs:element name="year_built" type="yearType"/>

254 <xs:element name="year_destroyed" type="yearType"/>

255 <xs:element name="how_destroyed" type="destrType"/>

256 <xs:element name="story" type="storyType"/>

257 </xs:sequence>

258 </xs:complexType>

259

260 11.) Defining 3 Attributes, one with restrictions

261 <xs:complexType name="sourceType">

262 <xs:attribute name="sectionid" type="xs:positiveInteger"/>

263 <xs:attribute name="newspaperid">

264 <xs:simpleType><!--just data-->

265 <xs:restriction base="xs:positiveInteger">

266 <!--exactly 4 digit pattern using regular expressions-->

267 <xs:pattern value="\d{4}"/>

268 </xs:restriction>

269 </xs:simpleType>

270 </xs:attribute>

271 </xs:complexType>

272

273 **Example of ***attribute definition*** in XML**

274 **Invalid:**

275 <source sectionid="101" newspaperid="21"/>

276 **Valid:**

277 <source sectionid="101" newspaperid="1321"/>

278

279 **12.) Defining Attributes, and making them required**

280 <xs:complexType name="nameType">

281 <xs:simpleContent>

282 <xs:extension base="xs:string">

283 <xs:attribute name="id" type="xs:integer" use="required"/>

284 </xs:extension>

285 </xs:simpleContent>

286 </xs:complexType>

287

288

289

290

291

292