XML Schema

Një XML Schema përshkruan strukturën e një XML dokumenti, ngjashëm si një DTD. XML Schema është një alternativë e bazuar në XML e DTD-së. Qëndron për **X**ML **S**chema **D**efinition.

```
<xs:element name="note">

<xs:complexType>
  <xs:sequence>
    <xs:element name="to" type="xs:string"/>
    <xs:element name="from" type="xs:string"/>
    <xs:element name="heading" type="xs:string"/>
    <xs:element name="body" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

</xs:element>
```

Schema e mësipërme interpretohet si vijon:

- <xs:element name = “note”> definon elementin e quajtur “note”
- <xs:complexType> elementi “note” është një lloj kompleks sepse përmban elemente tjera
- <xs:sequence> lloji kompleks është një sekuencë elementesh
- <xs:element name = “to” type = “xs:string”> elementi “to” është i llojit string (tekst)
- <xs:element name = “from” type = “xs:string”> elementi “from” është i llojit string
- <xs:element name = “heading” type = “xs:string”> elementi “heading” është i llojit string
- <xs:element name = “body” type = “xs:string”> elementi “body” është i llojit string

Qëllimi i një XML Schema është të definojë blloqet ndërtuese legale të një XML dokumenti.

- Elementet dhe atributet që mund të shfaqen në një dokument
- Numrin e (dhe rendin e) elementeve fëmijë
- Llojet e të dhënave për elementet dhe atributet
- Vlerat default dhe fikse për elementet dhe atributet

Referenca në një XML Schema

```
<?xml version="1.0"?>

<note
xmlns="https://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://www.w3schools.com/xml note.xsd">
  <to>Tove</to>
  <from>Jani</from>
```

```
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

XML Schemas janë më të fuqishme se DTD

- XML Schemas shkruhen në XML
- XML Schemas janë të zgjerueshme
- XML Schemas mbështesin llojet e të dhënave
- XML Schemas mbështesin namespaces

Pse përdoret një XML Schema?

Me XML Schema, fajllat në XML mund të bartin një përshkrim të formatit të tyre.
Me XML Schema, grupet e pavarura të njerëzve mund të merren vesh në një standard për shkëmbim të dhënash.
Me XML Schema, mund të verifikoni të dhëna.

Një nga fuqitë më të mëdha të XML Schemas është mbështetja për llojet e të dhënave:

- Është më e lehtë të përshkruajmë përmbajtjen e dokumentit
- Është më e lehtë të definojmë kufizime në të dhënat
- Është më e lehtë të validojmë korrektësinë e të dhënave
- Është më e lehtë të konvertojmë të dhënat ndërmjet llojeve të të dhënave

Një tjetër fuqi e XML Schemas është se ato shkruhen në XML.

XSD - Elementi <schema>

Elementi <schema> është elementi rrënjë i çdo XML Schema.

```
<?xml version="1.0"?>

<xs:schema>
...
...
</xs:schema>
```

Elementi schema mund të përmbajë disa atribute. Një deklarim i skemës shpesh duket si vijon:

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

Fragmenti i mësipërm tregon që elementet dhe llojet e të dhënave të përdorura në skemën vijnë nga namespace “http://www.w3.org/2001/XMLSchema”. Dhe poashtu tregon se elementet dhe llojet e të dhënave që vijnë nga ky namespace duhet të kenë prefiksin **xs:**

```
targetNamespace="https://www.w3schools.com"
```

Fragmenti i mësipërm tregon se elementet e definuara nga kjo skemë (note, to, from, heading, body.) vijnë nga namespace “https:www.w3schools.com”.

```
xmlns="https://www.w3schools.com"
```

Fragmenti i mësipërm tregon se namespace default është “https://www.w3schools.com”.

```
elementFormDefault="qualified"
```

Fragmenti i mësipërm tregon se ndonjë element i përdorur nga XML dokumenti që janë deklaruar në këtë skemë duhet të jetë namespace i kualifikuar.

Referencimi i një skemë në një XML Dokument

Ky XML Dokument ka një referencë në një XML Schema:

```
<?xml version="1.0"?>

<note xmlns="https://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://www.w3schools.com note.xsd">

<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

```
xmlns="https://www.w3schools.com"
```

Fragmenti i mësipërm specifikon deklarimin e namespace default. Ky deklarim i tregon schema-validatorit se të gjitha elementet që përdoren në këtë XML Dokument janë të deklaruar në namespace “https://www.w3schools.com”.

Kur e kemi në dispozicion XML Schema Instance namespace:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

mund të përdorim atributin schemaLocation. Ky atribut ka dy vlera, të ndarë nga një hapësirë. Vlera e parë është namespace për tu përdorur. Vlera e dytë është lokacioni i XML Schema për ta përdorur për këtë namespace.:

```
xsi:schemaLocation="https://www.w3schools.com note.xsd"
```

XSD Elementet e thjeshta

XML Schemas definojnë elementet e XML fajllave tuaj. Një element i thjeshtë është një XML element që përmban vetëm tekst. Ai nuk mund të përmbajë ndonjë element tjetër ose atribute.

Teksti mund të jetë i llojeve të ndryshme . Mund të jetë vetëm i llojeve të përfshirë në XML Schema Definition (boolean, string, date, etj.), ose mund të jetë një lloj që mund ta krijoni vetë.

Ju mund gjithashtu të shtoni detyrime (kufizime) në një lloj të dhënash në mënyrë që të kufizoni përmbajtjen e tij, ose ju mund të kërkoni që të dhënat të përputhen me një model specifik.

Definimi i elementit të thjeshtë

```
<xs:element name="xxx" type="yyy"/>
```

ku xxx është emri i elementit, ndërsa yyy është lloji i të dhënave.

XML Schema ka disa lloje të të dhënave. Më të zakonshmet janë:

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

Shembull, XML elemente:

```
<lastname>Refsnes</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

Në vazhdim janë definimet korresponduese të elementeve të thjeshta:

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

Vlerat default dhe fikse për elementet e thjeshta

Elementet e thjeshta mund të kenë një vlerë default ose vlerë fikse të specifikuar. Një vlerë default është automatikisht e caktuar në elementin kur nuk specifikohet ndonjë vlerë tjetër.

```
<xs:element name="color" type="xs:string" default="red"/>
```

Një vlerë fikse është gjithashtu e caktuar automatikisht në elementin, dhe nuk mund të specifikoni ndonjë vlerë tjetër.

```
<xs:element name="color" type="xs:string" fixed="red"/>
```

XSD Atributet

Të gjithë atributet deklarohen si lloje të thjeshta. Elementet e thjeshta nuk kanë atribute. Nëse një element ka atribute, ai konsiderohet të jetë një lloj kompleks. Por vetë atributi deklarohet si një lloj i thjeshtë.

Definimi i një atributi

```
<xs:attribute name="xxx" type="yyy"/>
```

ku xxx është emri i atributit dhe yyy specifikon llojin e të dhënave të atributit përkatës. Llojet e të dhënave janë të ngjashme si te elementet.

Shembull: xml element me një atribut

```
<lastname lang="EN">Smith</lastname>
```

Dhe në vazhdim është definimi korrespondues i atributir.

```
<xs:attribute name="lang" type="xs:string"/>
```

Vlerat default dhe fikse për atributet

Atributet mund të kenë të specifikuar vlera default ose fikse.

Vlera default:

```
<xs:attribute name="lang" type="xs:string" default="EN"/>
```

Vlera fikse:

```
<xs:attribute name="lang" type="xs:string" fixed="EN"/>
```

Atributet opsionale dhe required

Atributet janë opsionale by default. Për të specifikuar që atributi është required, përdorim atributin “use”:

```
<xs:attribute name="lang" type="xs:string" use="required"/>
```

Kufizimet në përmbajtjen

Kur një XML element ose atribut ka të definuar një lloj të dhënash, ajo vendos kufizime në përmbajtjen e elementit ose atributit. Nëse një XML element është i llojit “xs:date” dhe përmban një string si “Hello World”, elementi nuk do validohet. Me XML Schemas, mund gjithashtu të shtoni kufizimet e juaja për XML elementet ose atributet e juaja. Këto kufizime (detyrime) quhen facets. I shohim në vazhdim.

XSD Kufizimet/Facets

Kufizimet (detyrimet) përdoren për të definuar vlerat e pranueshme për XML elemenetet ose atributet. Kufizimet në XML elementete quhen facets.

Kufizimet në vlerat

Shembulli në vazhdim definon një element të quajtur “age” me një kufizim. Vlera e moshës nuk mund të jetë më e vogël se 0 ose më e madhe se 120:

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="120"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Kufizimet në një bashkësi vlerash

Për të kufizuar përmbajtjen e një XML elementi në një bashkësi të vlerave të pranueshme, duhet të përdorim enumeration constraint.
Shembulli i mëposhtëm definon një element të quajtur “car” me një kufizim. Vlerat e vetme të pranueshme janë: Audi, Golf, BMW:

```
<xs:element name="car">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Golf"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

ose

```
<xs:element name="car" type="carType"/>

<xs:simpleType name="carType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Audi"/>
    <xs:enumeration value="Golf"/>
    <xs:enumeration value="BMW"/>
  </xs:restriction>
</xs:simpleType>
```

Në këtë rast lloji “carType” mund të përdoret nga elementet tjera sepse nuk është pjesë e elementit “car”.

Kufizimet në një seri vlerash

Për të kufizuar përmbajtjen e një XML elementi që të definojë një seri numrash ose shkronjash që mund të përdoren, mund të përdorim pattern constraint. Shembulli i mëposhtëm definon një element të quajtur “letter” me një kufizim. Vlera e vetme e pranueshme është një nga shkronjat e vogla nga a në z:

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim definon një element të quajtur “initials” me një kufizim. Vlera e vetme e pranueshme është tre nga shkronjat e mëdha nga A në Z:

```
<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim gjithashtu definon një element të quajtur “initials” me një kufizim. Vlera e vetme e pranueshme është tre nga shkronjat e mëdha ose të vogla nga a në z:

```
<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
    </xs:restriction>
```

```
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim definon një element të quajtur “choice” me një kufizim. Vlera e vetme e pranueshme është një nga shkronjat x, y ose z:

```
<xs:element name="choice">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[xyz]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim definon një element të quajtur “prodid” me një kufizim. Vlera e vetme e pranueshme është pesë shifra në një sekuencë, dhe secila shifër duhet të jetë në rangun nga 0 deri në 9:

```
<xs:element name="prodid">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Kufizime tjera në një seri vlerash

Shembulli në vazhdim definon një element të quajtur “letter” me një kufizim. Vlera e pranueshme është zero ose më shumë paraqitje të shkronjave të vogla nga a deri në z:

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="([a-z])*"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim gjithashtu definon një element të quajtur “letter” me një kufizim. Vlera e pranueshme është një ose më shumë çifte të shkronjave, secili çift përbëhet nga një shkronjë e vogël e ndjekur nga një shkronjë e madhe. Për shembull, “sToP” do të validohet nga ky pattern, por jo “Stop” ose “STOP” ose “stop”:

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
```

```
      <xs:pattern value="([a-z][A-Z])+"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim defionon një element të quajtur “gender” me një kufizim. Vlera e vetme e pranueshme është male ose female:

```
<xs:element name="gender">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="male|female"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Shembulli në vazhdim definon një element të quajtur “password” me një kufizim. Duhet të jenë patjetër saktësisht tetë karaktere në një rresht dhe këto karaktere duhet të jenë me shkronja të vogla ose të mëdha nga a deri në z, ose një numër nga 0 deri në 9:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9]{8}"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Kufizimet në karakteret hapësirëa

Për të specifikuar se si karakteret hapësiart duhet të trajtohen, duhet të përdorim whiteSpace constraint.

Ky shembull definon një element të quajtur “address” me një kufizim. Kufizimi “whiteSpace” vendoset në “preserve”, që nënkupton se XML procesori nuk do të largojë ndonjë karakter hapësira të bardha:

```
<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="preserve"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Ky shembull gjithashtu definon një element të quajtur “address” me një kufizim. Kufizimi whiteSpace është vendosur në “replace”, që nënkupton që XML procesori do të zëvendësojë të gjitha hapësirat e bardha (rreshtat e rinjë, tabet, hapësirat) me hapësira:

```
<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="replace"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Ky shembull gjithashtu definon një element të quajtur “address” me një kufizim. Kufizimi whiteSpace është vendosur në “collapse”, që nënkupton që XML procesori do të largojë të gjitha hapësirat e bardha (rreshtat e rinjë, tabet, hapësirat largohen, dhe hapësirat e shumta reduktohen në një hapësirë të vetme):

```
<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Kufizimet në gjatësinë

Për të kufizuar gjatësinë e një vlere në një element, ne mund të përdorim detyrimet length, maxLength, dhe minLength.
Ky shembull definon një element të quajtur “password” me një kufizim. Vlera duhet të jetë saktësisht tetë karaktere:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Ky shembull definon një element tjetër të quajtur “password” me një kufizim. Vlera duhet të jetë minimun pesë karaktere dhe maksimum tetë karaktere:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="5"/>
```

```
      <xs:maxLength value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Kufizimet për llojet (tipet) e të dhënave

Detyrimi	**Përshkrimi**
enumeration	Definon një listë të vlerave të pranueshme.
fractionDigits	Specifikon numrin maksimal të pjesëve decimale të lejuara. Duhet të jetë i barabartë ose më i madh se zero.
length	Specifikon numrin fiks të karaktereve ose elementet e listës që lejohen. Duhet të jetë i barabartë ose më i madh se zero.
maxExclusive	Specifikon kufirin e sipërm për vlerat numerike (vlera duhet të jetë më e vogël se kjo vlerë).
maxInclusive	Specifikon kufirin e sipërm për vlerat numerike (vlera duhet të jetë më e vogël ose e barabartë me këtë vlerë).
maxLength	Specifikon numrin maksimal të karaktereve ose elementeve të listës, që lejohen. Duhet të jetë e barabartë ose më e madhe se zero.
minExclusive	Specifikon kufirin e poshtëm për vlerat numerike (vlera duhet të jetë më e madhe se kjo vlerë).
minInclusive	Specifikon kufirin e poshtëm për vlerat numerike (vlera duhet të jetë më e madhe ose e barabartë me këtë vlerë).
minLength	Specifikon numrin minimal të karaktereve ose elementeve të listës, që lejohen. Duhet të jetë e barabartë ose më e madhe se zero.
pattern	Definon sekuencën fikse të karaktereve që janë të pranueshme.
totalDigits	Specifikon numrin fiks të shifrave të numrit, që lejohen. Duhet të jetë më e madhe se zero.
whiteSpace	Specifikon se si trajtohen hapësirat e bardha (rreshtat e rinjë, hapësirat, tabet).

XSD Elementet Komplekse

Një element kompleks përmban elemente tjera dhe/ose atribute. Janë katër lloje të elementeve komplekse:

- elementet e zbrazëta
- elementet që përmbajnë vetëm elemente tjera
- elementet që përmbajnë vetëm tekst
- elementet që përmbajnë edhe elemente edhe tekst

Secili prej këtyre elementeve mund të përmbajnë edhe atribute.!

Shembuj të elementeve komplekse

Një XML element kompleks, “product”, që është i zbrazët:

```
<product pid="1345"/>
```

Një XML element kompleks, “employee”, që përmban vetëm elemente tjera:

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

Një XML element kompleks, “food”, që përmban vetëm tekst:

```
<food type="dessert">Ice cream</food>
```

Një XML element kompleks, “description”, që përmban edhe tekst edhe elemente:

```
<description>
It happened on <date lang="norwegian">03.03.99</date> ....
</description>
```

Si të definojmë një element kompleks

Shikoni në XML elementin kompleks “employee”, që përmban vetëm elemente tjera:

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

Mund të definojmë një element kompleks në një XML Schema në dy mënyra të ndryshme:

1. Elementi “employee” mund të deklarohet direkt duke emëruar elementin, si vijon:

```
<xs:element name="employee">
```

```
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Nëse përdorim metodën e përshkruar më herët, vetëm elementi “employee” mund të përdorë llojin kompleks të specifikuar. Vini re se elementet fëmijë, “firstname” dhe “lastname”, janë të rrethuar nga treguesi <sequence>. Kjo nënkupton se elementet fëmijë duhet të shfaqen në të njëjtin rend siç janë deklaruar. Do të mësojmë në lidhje me treguesit më vonë.

2. Elementi “employee” mund të ketë një atribut type që referon në emrin e llojit kompleks që përdoret:

```
<xs:element name="employee" type="personinfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Nëse përdorni metodën e përshkruar më lartë, elemente të ndryshme mund të referojnë në llojin e njëjtë kompleks, si vijon:

```
<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Ju mund gjithashtu ta bazoni një element kompleks në një element ekzsitues kompleks dhe të shtoni elemente tjera, si vijon:

```
<xs:element name="employee" type="fullpersoninfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
```

```
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XSD elementet e zbrazëta

Një element kompleks i zbrazët nuk mund të ketë përmbajtje, vetëm atribute.
Një XML element i zbrazët:

```
<product prodid="1345" />
```

Elementi “product” nuk ka përmbajtje. Për të definuar një lloj pa përmbajtje, ne duhet të definojmë një lloj që lejon elementet në përmbajtjen e tij, por aktualisht nuk deklarojmë ndonjë element, si vijon:

```
<xs:element name="product">
  <xs:complexType>
    <xs:complexContent>
      <xs:restriction base="xs:integer">
        <xs:attribute name="prodid" type="xs:positiveInteger"/>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

Në shembullin e mësipërm, definojmë një lloj kompleks me një përmbajtje komplekse. Elementi complexContent sinjalizon që ne kemi qëllim të kufizojmë ose zgjerojmë modelin e përmbajtjes së një lloji kompleks, dhe kufizimi i numrave të plotë deklaron një atribut, por nuk përfshin ndonjë përmbajtje të elementit.

Megjithatë, është e mundshme të deklarojmë elementin “product” më shkurtimisht, si vijon:

```
<xs:element name="product">
  <xs:complexType>
    <xs:attribute name="prodid" type="xs:positiveInteger"/>
```

```
  </xs:complexType>
</xs:element>
```

Ose mund ti jepni elementint complexType një emër, dhe të lejoni që elementi “product” të ketë një atribut type që referon që emrin e complexType (nëse përdorni këtë metodë, disa elemente mund të referojnë në të njëjtin lloj kompleks):

```
<xs:element name="product" type="prodtype"/>

<xs:complexType name="prodtype">
  <xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
```

Llojet komplekse që përmbajnë vetëm elemente

```
<person>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</person>
```

Ju mund të definoni elementin “person” në një skemë, si vijon:

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Vini re etiketën <xs:sequence>. Kjo nënkupton se elementet e definuara (“firstname” dhe “lastname”) duhet të shfaqen në atë rend brenda elementit “person”.

Ose mund ti jepni elementit complexType një emër, dhe të lejoni që elementi “person” të ketë një atribut type që referon në emrin e complexType (nëse përdorni këtë metodë, disa elemente mund të referojnë në të njëjtin lloj komples):

```
<xs:element name="person" type="persontype"/>

<xs:complexType name="persontype">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Elementet komplekse që përmbajnë vetëm tekst

Ky lloj përmban vetëm përmbajtje të thjeshtë (tekst dhe atribute), prandaj ne shtojmë një element simpleContent përreth përmbajtjes. Kur përdorim përmbajtje të thjeshtë, ju duhet të definoni një ekstension ose një kufizim brenda elementit sipmleContent, si vijon:

```
<xs:element name="somename">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="basetype">
      ....
      ....
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

OR

```
<xs:element name="somename">
  <xs:complexType>
    <xs:simpleContent>
      <xs:restriction base="basetype">
      ....
      ....
      </xs:restriction>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

Përdorim elementin extension/restriction për të zgjeruar ose kufizuar llojin e thjeshtë base për elementin.

Shembull i një XML elementi, “shoesize”, që përmban vetëm tekst:

```
<shoesize country="france">35</shoesize>
```

Shembulli i mëposhtëm deklaron një complexType, “shoesize”. Përmbajtja është definuar si një vlerë integer, dhe elementi “shoesize” gjithashtu përmban një atribut të quajtur “country”:

```
<xs:element name="shoesize">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:integer">
        <xs:attribute name="country" type="xs:string" />
      </xs:extension>
    </xs:simpleContent>
```

```
  </xs:complexType>
</xs:element>
```

Dhe poashtu të krijojmë vetë llojin “showsize”:

```
<xs:element name="shoesize" type="shoetype"/>

<xs:complexType name="shoetype">
  <xs:simpleContent>
    <xs:extension base="xs:integer">
      <xs:attribute name="country" type="xs:string" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

Llojet komplekse me përmbajtje të përzier

Një element me llojin kompleks të përzier mund të përmbajë atribute, elemente dhe tekst.

Një XML element “letter”, që përmban edhe tekst edhe elemente tjera:

```
<letter>
  Dear Mr. <name>John Smith</name>.
  Your order <orderid>1032</orderid>
  will be shipped on <shipdate>2001-07-13</shipdate>.
</letter>
```

Skema vijuese deklaron elementin “letter”:

```
<xs:element name="letter">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="orderid" type="xs:positiveInteger"/>
      <xs:element name="shipdate" type="xs:date"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Për të mundësuar që karaktere (tekst) të shfaqen ndërmjet elementeve fëmijë të “letter”, atributi mixed duhet të vendoset në “true”. Etiketa <xs:sequence> nënkupton që elementet e definuara (name, orderid dhe shipdate) duhet të shfaqen në atë rend në brenda elementit “letter”.

Ose poashtu mund të deklarojmë llojin kompleks duke i dhënë një emër dhe ta përdorim në elementet tjera përmes atributit type:

```
<xs:element name="letter" type="lettertype"/>
```

```
<xs:complexType name="lettertype" mixed="true">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="orderid" type="xs:positiveInteger"/>
    <xs:element name="shipdate" type="xs:date"/>
  </xs:sequence>
</xs:complexType>
```

XSD Indicators

Mund të kontrollojmë se si do të përdoren elementet në dokumentet, me indikatorët (treguesit).

Janë shtatë indikatorë:

Indikatorët e rendit:
- All
- Choice
- Sequence

Indikatorët e paraqitjes:
- maxOccurs
- minOccurs

Indikatorët gruporë
- Group name
- atributteGroup name

Order Indicators

Përdoren për të definuar rendin e elementeve.

All Indicator

Indikatori <all> specifikon që elementet fëmijë mund të shfaqen në çfardo rendi, dhe që secili element fëmijë duhet të paraqitet vetëm njëherë:

```
<xs:element name="person">
  <xs:complexType>
    <xs:all>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:all>
  </xs:complexType>
</xs:element>
```

Kur përdorim indikatorin <all> mund të vendosim indikatorin <minOccurs> në 0 ose 1 dhe indikatorin <maxOccurs> mund të vendoset vetëm në 1.

Choice Indicator

Indikatori <choice> specifikon që njëri ose tjetri element fëmijë mund të paraqitet (shfaqet):

```
<xs:element name="person">
  <xs:complexType>
    <xs:choice>
      <xs:element name="employee" type="employee"/>
      <xs:element name="member" type="member"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

Sequence Indicator

Indikatori <sequence> specifikon që elementet fëmijë duhet të shfaqen në një rend specifik:

```
<xs:element name="person">
   <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Occurrence Indicators

Indikatorët e paraqitjes (shfaqjes) përdoren për të definuar se sa shpesh paraqitet (shfaqet) një element.

Për të gjithë indikatorët “Order” dhe “Group” (any, all, choice, sequence, group name, dhe group reference) vlera default për maxOccurs dhe minOccurs është 1.

maxOccurs Indicator

Indikatori <maxOccurs> specifikon numrin maksimal të herëve që një element mund të paraqitet (shfaqet):

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string" maxOccurs="10"/>
```

```
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Shembulli mësipër tregon që elementi “child_name” mund të paraqitet minimum një herë (vlera default e minOccurs është 1) dhe maksimum dhjetë herë në elementin “person”.

minOccurs Indicator

Indikatori <minOccurs> specifikon numrin minimal të herëve që një element mund të paraqitet (shfaqet):

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string"
     maxOccurs="10" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Shembulli i mësipërm tregon se elementi “child_name” mund të paraqitet minimum zero herë dhe maksimum dhjetë herë në elementin “person”.

Për të lejuar që një element të shfaqet (paraqitet) në një numër të palimituar të herëve, përdorim deklarimin maxOccurs = “unbounded”.

Group Indicators

Përdoren për të definuar bashkësitë e lidhura të elementeve.

Element Groups

Definohen me deklarimin group, si vijon:

```
<xs:group name="groupname">
...
</xs:group>
```

Duhet të definojmë një element all, choice, ose sequence brenda deklarimit group. Shembulli vijues definon një grup të emëruar “persongroup”, që definon një grup elementesh që duhet të paraqiten (shfaqen) në rend fiks:

```
<xs:group name="persongroup">
  <xs:sequence>
```

```
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:element name="birthday" type="xs:date"/>
  </xs:sequence>
</xs:group>
```

Pasi të keni definuar një grup, ju mund të referenconi atë në një tjetër definimin, si vijon:

```
<xs:group name="persongroup">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:element name="birthday" type="xs:date"/>
  </xs:sequence>
</xs:group>

<xs:element name="person" type="personinfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:group ref="persongroup"/>
    <xs:element name="country" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Attribute Groups

Definohen me deklarimin attributeGroup, si vijon:

```
<xs:attributeGroup name="groupname">
...
</xs:attributeGroup>
```

Shembulli vijues definon një grup atributesh të emëruar “personattrgroup”:

```
<xs:attributeGroup name="personattrgroup">
  <xs:attribute name="firstname" type="xs:string"/>
  <xs:attribute name="lastname" type="xs:string"/>
  <xs:attribute name="birthday" type="xs:date"/>
</xs:attributeGroup>
```

Pasi të keni definuar një grup atributesh, mund ta referenconi në një tjetër definimin, si vijon:

```
<xs:attributeGroup name="personattrgroup">
  <xs:attribute name="firstname" type="xs:string"/>
  <xs:attribute name="lastname" type="xs:string"/>
  <xs:attribute name="birthday" type="xs:date"/>
```

```
</xs:attributeGroup>

<xs:element name="person">
  <xs:complexType>
    <xs:attributeGroup ref="personattrgroup"/>
  </xs:complexType>
</xs:element>
```

Elementi <any>

Na lejon që të zgjerojmë XML dokumentet me elementet që nuk janë të specifikuar nga skema!

Shembulli i mëposhtëm tregon një deklarim të elementit “person”. Duke përdorur elementin <any> ne mund ta zgjerojmë përmbajtjen e “person” (pas <lastname>) me ndonjë element:

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:any minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Elementi <anyAttribute>

Na lejon që të zgjerojmë XML dokumentet me atributete që nuk janë specifikuar me skemën!

Shembulli i mëposhtëm tregon një deklarim për elementin “person”. Duke përdorur elementin <anyAttribute> ne mund të shtojmë ndonjë atribut në elementin “person”

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
    <xs:anyAttribute/>
  </xs:complexType>
</xs:element>
```