M. Essert: WEB PROGRAMIRANJE

4. predavanje

XML Schema je na XML utemeljena alternativa za DTD.

XML Schema opisuje strukturu XML dokumenta.

XML Schema jezik se označuje kao XML Schema Definition (XSD).

XML Schema:

- definira elemente koji se mogu pojaviti u dokumentu
- definira atribute koji se mogu pojaviti u dokumentu
- definira koji elementi su djeca (child) elementi
- definira redoslijed djece (child) elemenata
- definira broj djece (child) elemenata
- definira je li element prazan ili uključuje tekst
- definira tipove podataka za elemente i atribute
- definira pretpostavljenu i fiksnu vrijednosti za elemente i atribute

---

XML Schema je nasljednik od DTD-a

XML Schema se koristi u većini Web aplikacija kao nadomjestak za DTD. Ovdje su neki razlozi za to:

- XML Schema su proširive za buduće dodatke
- XML Schema are bogatije i korisnije od DTD-a
- XML Schema se pišu u XML-u
- XML Schema podržavaju tipove podataka
- XML Schema podržavaju prostor imena (namespaces)

XML Schema je prvotno preporučio Microsoft, ali je postala službeni standard od strane W3C ureda u lipnju 2001. Specifikacija standarda je stabilna i nadgleda se od W3C članstva.

XML Schema podržava tipove podataka

Jedan od najvećih dosega XML Schema je podržavanje tipove podataka, čime se postiže:

- jednostavan opis dopustivog sadržaja dokumenta
- jednostavna provjera ispravnosti podataka
- jednostavan rad s podacima iz baza podataka (database)

- jednostavno je definirati obličja podataka (data facets) (restrikcije na podacima)
- jednostavno definiranje uzoraka, formata podataka (data patterns, data formats)
- jednostavnija pretvorba podataka između različitih tipova podataka

---

XML Schema-e koriste XML sintaksu

Drugi veliki doseg XML Schema je da su one napisane u XML-u, što znači:

- Ne mora se učiti neki drugi jezik
- Koristi se isti XML editor da se editiraju Schema datoteke
- Može se koristiti XML parser za parsiranje (izvlačenje podataka iz) schema
- Moguće je obraditi schemu sa XML DOM
- Može se transformirati schema sa XSLT

XML Schema-e daje sigurnost prijenosu podataka

Kada se podaci šalju od pošiljatelja primatelju, važno je da obje strane imaju ista „očekivanja" o sadržaju. Sa XML Schema-ma pošiljatelj može opisati podatke na način na koji će ih primatelj razumjeti.

Datum kao 1999-03-11 može se (u nekim zemljama) interpretirati kao 3. studenoga ili (kao u drugim zemljama) kao 11. ožujka, ali kao XML element s tipom podataka kao:

<date type="date">1999-03-11</date>

garantira međusobno razumijevanje sadržaja, jer XML tip podataka za datum zahtijeva format CCYY-MM-DD.

XML Schema su proširive

XML Schema su proširive, baš kao i XML, budući da su pisane u XML-u pa je moguće:

- Ponovno korištenje jedne schema-e u drugim schema-ma
- Stvoriti vlastite tipove podataka izvedene iz standardnih tipova
- Referencirati višestruke schema-e iz istog dokumenta

Dobro formatirano (Well-Formed) nije dovoljno

Dobro formatirani XML dokument je dokument koji prihvaćaju XML sinaksna pravila:

- Mora početi sa XML deklaracijom

- Mora imati jedan jedincati korijenski (root) element
- Svi početni tag-ovi moraju imati pridružene krajnje-tagove
- XML tag-ovi su osjetljivi na velika i mala slova
- svi elementi moraju biti zatvoreni
- svi elementi moraju biti ispravno ugnježđeni, bez međusobnog preklapanja
- svi atributne vrijednosti moraju biti unutar navodnika
- XML entities moraju se koristiti za specijalne znakove

Čak ako su dokumenti dobro formatirani oni još uvijek mogu sadržavati pogreške, oni mogu još uvijek sadržavati pogreške, koje mogu imati ozbiljne posljedice. Sa XML schema-ma, većinu ovih pogrešaka moguće je otkriti s pomoću software-a za provjeru valjanosti.

XML dokumenti mogu imati reference na DTD ili na XML schema-e.

Jednostavni XML dokument

Pretpostavimo jednostavni XML dokument nazvan "note.xml":

```
<?xml version="1.0"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Jednostavni DTD

Ovo je jednostavna DTD datoteka nazvana "note.dtd" koja definira elemente od gornjeg XML dokumenta ("note.xml"):

```
<!ELEMENT note (to, from, heading, body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Linija 1 definira **note** element koji ima četiri elementa: "to, from, heading, body". Linija 2-5 definira **to** element da bude tipa "#PCDATA", **from** element da bude tipa "#PCDATA", i tako dalje ...

Jednostavna XML schema

Ovo je jednostavna XML schema datoteka nazvana "note.xsd" koja definira elemente od XML dokumenta iznad ("note.xml"):

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
<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>
</xs:schema>
```

note element se kaže da je složeni tip (**complex type**) budući da sadrži ostale druge elemente. Za ostale elemente (to, from, heading, body) kaže se da su jednostavni tipovi (**simple types**) budući da oni ne sadrže druge elemente.

Referenca na DTD

Ovaj XML dokument ima referencu na DTD:

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM
"http://www.w3schools.com/dtd/note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Referenca na XML schema-u

Ovaj XML dokument ima referencu na XML schema-u:

```
<?xml version="1.0"?>
<note
xmlns="http://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=
"http://www.w3schools.com/schema/note.xsd">

<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
```

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

<schema> element je korijen (root) element svake XML schema-e!

<schema> element

<schema> element je korijenski element svake XML schema-e:

```
<?xml version="1.0"?>
<xs:schema>
...
...
</xs:schema>
```

<schema> element može sadržavati neke atribute. Deklaracija schema-e često izgleda poput ove:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

Sljedeći fragment:

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

pokazuje da elementi i tipovi podataka korišteni u schemi (shema, element, complexType, sequence, string, boolean, itd.) dolaze iz "http://www.w3.org/2001/XMLSchema" prostora imena. Ono također određuje da elementi i tipovi podataka koji dolaze s adrese: "http://www.w3.org/2001/XMLSchema" prostora imena trebaju imati prefix od xs: !!

Ovaj fragment:

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

pokazuje da elementi definirani s ovom shem-om (note, to, from, heading, body.) dolaze iz "http://www.w3schools.com" prostora imena.

Ovaj fragment:

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

pokazuje da je pretpostavljeni prostor imena "http://www.w3schools.com".

Ovaj fragment:

```
elementFormDefault="qualified"
```

pokazuje da bilo koji element korišten sa XML instancom dokumenta koji je deklariran u ovoj schema-i mora biti označen, kvalificiran s prostorom imena.

Referenciranje schema-e u XML dokumentu

Ovaj XML dokument ima referencu na XML Schema:

```
<?xml version="1.0"?>
<note xmlns="http://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3schools.com note.xsd">
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Sljedeći fragment:

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

specificira pretpostavljenu deklaraciju prostora imena. Ova deklaracija govori programu za provjeru (schema-validator) da se svi elementi korišteni u ovom XML dokumentu deklariraju u "http://www.w3schools.com" prostoru imena.

U trenutku kad se ima XML schema instanca prostora imena:

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

može se koristiti schemaLocation atribut. This atribut ima dvije vrijednosti. Prva vrijednost je prostor imena koji se koristi, a druga vrijednost je lokacija XML schema-e koja se koristi za taj prostor imena:

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

XML Schema-e definiraju elemente od XML datoteka.

Jednostavni (simple) element

Jednostavni element je XML element koji sadrži samo tekst. On ne može sadržavati neki drugi element ili atribut.

Tekst pritom može imati više različitih tipova podataka. On može biti jedan od tipova podataka koji su uključeni u XML schema definiciju (boolean, string, date, itd.), ili može biti poseban tip podatka kojeg korisnik sam definira.

Moguće je također dodati restrikcije (facets) na tipove podataka da bi se ograničio njihov sadržaj ili zahtijevati podudaranje podatka s definiranim uzorkom.

Sintaksa za definiranje jednostavnog elementa je:

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

gdje je xxx ime elementa, a yyy je tip podatka tog elementa.

Ovdje su neki XML elementi:

```
<lastname>Refsnes</lastname>
<age>34</age>
<dateborn>1968-03-27</dateborn>
```

A ovdje su odgovarajuće definicije jednostavnih elemenata:

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

XML schema ima puno ugrađenih (built-in) tipove podataka. Ovdje je lista najčešćih tipova:

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

Deklaracija pretpostavljenih i fiksnih vrijednosti za jednostavne elemente

Pretpostavljenu vrijednost se automatski pridružuje elementu kad se ne specificira druga vrijednost. U sljedećem primjeru pretpostavljena vrijednost je "red":

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

Fiksne vrijednosti se također automatski pridružuju elementu. Nije moguće specificirati neku drugu vrijednost. U sljedećem primjeru fiksna vrijednost je "red":

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

Atribut

Jednostavni elementi ne mogu imati atribute. Ako element ima atribute, onda se on promatra kao složeni tip (complex type). Pritom se atribut sam za sebe uvijek deklarira kao jednostavni tip..

Sintaksa za definiranje atributa je:

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

gdje je xxx ime atributa, a yyy je tip podatka tog atributa.

Ovdje je jedan XML element s atributom:

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

A ovdje odgovarajuća jednostavna definicija atributa:

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

Deklaracija pretpostavljenih i fiksnih vrijednosti za atribute

Atributi mogu imati pretpostavljenu ili fiksnu vrijednost.

Pretpostavljenu vrijednost se automatski pridružuje atributu kad se ne specificira druga vrijednost. U sljedećem primjeru pretpostavljena vrijednost je "EN":

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

Fiksne vrijednosti se također automatski pridružuju atributu. Nije moguće specificirati neku drugu vrijednost. U sljedećem primjeru fiksna vrijednost je "EN":

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

Pretpostavlja se da su svi atributi proizvoljni. Da se eksplicitno izrazi kako je atribut proizvoljan (optional), koristi se "use" atribut:

```
<xs:atribut name="lang" type="xs:string" use="optional"/>
```

A ako se atribut zahtijeva piše se „required“:

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

Restrikcije na sadržaj

Kad XML element ili atribut ima definiran tip, onda je automatski postavljena restrikcija na elementni ili atributni sadržaj. Ako je XML element tipa "xs:date" i sadrži string poput "Hello Mother", onda element neće proći provjeru valjanosti (validation).

Osim toga moguć je XML schema-ma dodati vlastite restrikcije na XML elemente i atribute. Ove restrikcije se zovu obličja (facets).

Restrikcije na vrijednosti

Ovaj primjer definira element nazvan "age" sa restrikcijom. Vrijednost elementa „age" ne može biti manje od 0 ili veće od 100:

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

Restrikcije na skup vrijednosti

Ograničiti sadržaj XML element na skup prihvatljivih vrijednosti, moguće je upotrebom „enumeration" ograničenja.

Ovaj primjer definira element nazvan "car":

```
<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>
```

Element "car" je jednostavni tip s restrikcijom. Dopuštene vrijednosti su: Audi, Golf, BMW.

Gornji primjer može se također napisati ovako:

```
<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>
```

Primjedba: U ovom slučaju tip "carType" može se koristiti i od strane drugog elemente budući da nije dio "car" elementa.

Restrikcije na niz vrijednosti

Ograničiti sadržaj XML elementa da smije definirati samo niz brojeva ili slova, postiže se preko uzorka s ograničenjem.

Ovaj primjer definira element nazvan "letter":

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

Element "letter" je jednostavan tip sa restrikcijom. Jedina dopustiva vrijednost je JEDAN od MALIH slova iz popisa a do z.

Sljedeći primjer definira element nazvan "initials":

```
<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>
```

Element "initials" je jednostavan tip sa restrikcijom. Jedina dopustiva vrijednost su TRI VELIKA slova iz popisa A do Z.

Sljedeći primjer definira element nazvan "initials":

```
<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>
```

Element "initials" je jednostavan tip sa restrikcijom. Jedina dopustiva vrijednost su TRI MALA ili VELIKA slova iz popisa a do z, odnosno A do Z.

Sljedeći primjer definira element nazvan "choice":

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

Element "choice" je jednostavan tip sa restrikcijom. Jedina dopustiva vrijednost JEDAN od sljedećih slova: x, y, ili z.

Sljedeći primjer definira element nazvan "prodid":

```
<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>
```

Element "prodid" je jednostavan tip sa restrikcijom. Jedine dopustive vrijednosti su PET znamenki u nizu, a svaka znamenka mora biti u području od 0 do 9.

Ostale restrikcije na niz vrijednosti

Ovaj primjer definira element nazvan "letter":

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

Element "letter" je jednostavan tip sa restrikcijom. Dopustive vrijednosti su nula ili više pojavaka malih slova od a do z.

Ovaj primjer također definira element nazvan "letter":

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

Element "letter" je jednostavan tip sa restrikcijom. Dopustive vrijednosti su jedan ili više pojavaka parova malog slova nakon kojeg slijedi veliko, iz skupa slova od a do z, odnosno, A do Z.

Sljedeći primjer definira element nazvan "gender":

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

Element "gender" je jednostavan tip sa restrikcijom. Jedine dopustive vrijednosti su male ILI female.

Sljedeći primjer definira element nazvan "password":

```
<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>
```

Element "password" je jednostavan tip sa restrikcijom. Mora biti točno 8 znakova u retku i ti znakovi moraju biti mala ili velika slova od a do z ili znamenke od 0 do 9.

Restrikcije na znakove praznina

Da bi se specificiralo kako će se obraditi praznine, potrebno je koristiti whiteSpace ograničenje.

Sljedeći primjer definira element nazvan "address":

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

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

Element "address" je jednostavan tip s restrikcijom. Ograničenje whiteSpace je postavljeno na "preserve", što znači da XML procesor NEĆE maknuti bilo koju prazninu ili prazni redak.

Ovaj primjer takodjer definira element nazvan address":

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

Element "address" je jednostavan tip s restrikcijom. Ograničenje whiteSpace je postavljeno na "replace", što znači da će XML procesor zamijeniti bilo koje znakove praznina (line feeds, tabs, spaces, i carriage returns) s prazninama, razmacima.

Ovaj primjer takodjer definira element nazvan address":

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

Element "address" je jednostavan tip s restrikcijom. Ograničenje whiteSpace je postavljeno na "collapse", što znači da ĆE XML procesor izbaciti sve znakove praznina (line feeds, tabs, spaces, i carriage returns) i reducirati ih na jedan razmak. Razmaci iza ili ispred znakova (leading i trailing spaces) će jednostavno nestati.

Restrikcije na duljinu

Ograničenje duljine neke vrijednosti elementa, upravlja se ograničenjima: length, maxLength i minLength.

Ovaj primjer definira element nazvan "password":

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

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

Element "password" je jednostavan tip s restrikcijom. Vrijednost mora biti točno 8 znakova duga.

Ovaj primjer definira još jedan element nazvan "password":

```
<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>
```

Ovaj element "password" je jednostavan tip s restrikcijom. Vrijednost mora biti najmanje 5, a najviše 8 znakova duga.

Restrikcije za tipove podataka

Ograničenje	Opis
enumeration	Definira popis dopuštenih vrijednosti
fractionDigits	Specificira maksimalan broj dopuštenih decimalnih mjesta. Mora biti jednak ili veći od nule.
length	Specificira točan broj dopuštenih znakova ili članova popisa. Mora biti jednak ili veći od nule.
maxExclusive	Specificira gornju među za numeričke vrijednosti (vrijednost mora biti manja od ove vrijednosti)
maxInclusive	Specificira gornju među za numeričke vrijednosti (vrijednost mora biti manja ili jednaka od ove vrijednosti)
maxLength	Specificira maksimalan broj dopuštenih članova popisa. Mora biti jednak ili veći od nule.
minExclusive	Specificira donju među za numeričke vrijednosti (vrijednost mora biti veća od ove vrijednosti)
minInclusive	Specificira donju među za numeričke vrijednosti (vrijednost mora biti veća ili jednaka od ove vrijednosti)
minLength	Specificira maksimalan broj dopuštenih članova popisa. Mora biti jednak ili veći od nule.
pattern	Definira točan broj dopuštenih znakova.
totalDigits	Specificira točan broj dopuštenih znamenki. Mora biti veći od nule.
whiteSpace	Specificira kako će se dohvaćati prazna mjesta (line feeds, tabs, spaces, i carriage returns)

Složeni element

Složeni element je XML element koji sadrži elemente i/ili atribute.

Postoje četiri vrste složenih elemenata:

- prazni elementi
- elementi koji sadrže samo druge elemente
- elementi koji sadrže samo tekst
- elemente koji sadrže i druge elemente i tekst

Primjedba: Svaki od ovih elemenata mogu također sadržavati atribute!

Primjeri složenih XML elemenata

Složeni XML element, "product", koji je prazan:

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

Složeni XML element, "employee", koji sadrži samo druge elemente:

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

Složeni XML element, "food", koji sadrži samo tekst:

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

Složeni XML element, "description", koji sadrže i druge elemente i tekst:

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

Definiranje složenih elemenata

Neka se promotri složeni XML element, "employee", koji sadrži samo druge elemente:

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

U XML schema-i složeni elementi mogu se definirati na različite načine:

1. "employee" element može se deklarirati direktno imenujući element, kao na ovaj način:

```
<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>
```

Ako se koristi gore opisana metoda, onda samo "employee" element može koristiti specificirani složeni type. Treba primijetiti da su djeca elementi (child elements), "firstname" i "lastname", su okruženi sa <sequence> indikatorom. To znači da se djeca elementi moraju pojaviti u istom redoslijedu kako su deklarirani: prvo "firstname", a onda "lastname".

2. Element "employee" može imati tipski atribut koji referencira ime složenog tipa koji se koristi:

```
<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>
```

Ako se koristi gore opisana metoda, onda se nekoliko elemenata može referencirati na isti složeni tip, na primjer ovako:

```
<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>
```

Moguće je također utemeljiti složeni tipski element na postojećem složenom tipu i dodati neke elemente, kao na primjer:

```
<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>
```

Definiranje složenih tipova za prazne elemente

Prazni složeni element može sadržavati atribut; ali on ne može imati neki sadržaj između otvorenih i zatvorenih tag-ova.

Prazan XML element:

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

Element "product", definiran poviše, uopće nema sadržaj. Definirati tip bez sadržaja, mora se s tipom koji dopušta samo elemente u svom sadržaju, ali bez stvarnog deklariranja bilo kojeg elementa, kao na primjer:

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

U gornjem primjeru definiran je complexType koji ima complexContent, tj. samo elemente. Element complexContent element poručuje da se namjerava sažeti ili proširiti sadržaj modela složenog tipa i da restrikcija cijelog broja deklarira jedan atribut, ali ne uvodi sadržaj nekog elementa.

Međutim, moguće je deklarirati element 'product' zgusnutije, poput:

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

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

Moguće je dati i complexType element ime, i pustiti da "product" element ima tip atributa koji se referira na ime od complexType (ako se koristi ova metoda, onda nekoliko elemenata može referencirati isti složeni tip):

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

Definirati složeni Tip samo s elementom

Jedan "samo element" složeni tip sadrži element koji u sebi ima samo druge elemente.

XML element "person" sadrži samo druge elemente:

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

To se može definirati kao "person" element u schema-i na ovaj način:

```
<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>
```

Primijetite <xs:sequence> tag. On znači da se definirani elementi ("firstname" i "lastname") moraju pojaviti u takvom redoslijedu unutar "person" elementa.

Moguć je zadati complexType elementu ime i načiniti da "person" element ima tip atributa koji se referencira na complexType ime (ako se koristi ova metoda, onda nekoliko elemenata može referencirati isti složeni tip):

```
<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>
```

Definirati složeni Text-Only element (element sa samim tekstom)

To je složeni tekst element koji može sadržavati i atribute i tekst.

Ovaj tip sadrži samo jednostavni sadržaj (tekst i atribute), pa stoga dodajemo simpleContent element oko sadržaja. Kad se koristi jednostavni sadržaj, mora se definirati proširenje ILI unutar simpleContent elementa, kao ovdje:

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

ILI

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

Preporuka: Treba koristiti proširujući element za proširenje osnovnog jednostavnog tipa, a onda koristiti restriktivni element da se ograniči osnovni tip za element.

Ovdje je primjer XML elementa, "shoesize", koji sadrži text-only element:

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

Sljedeći primjer deklarira complexType, "shoesize". Sadržaj se definira kao cjelobrojni tip podatka i "shoesize" element koji također sadrži element nazvan "country":

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

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

Moguće je također complexType elementu dati ime, neka to bude "shoesize" element koji ima tipni atribut koji se referencira na ime complexType elementa (ako se koristi ova metoda,onda nekoliko elemenata može referencirati isti složeni tip):

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

Definirati složene tipove s mješovitim sadržajem

Mješoviti složeni tipni elementi mogu sadržavati atribute, elemente i tekst.

XML element "letter", sadrži i elemente i tekst:

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

Primijetite da se tekst pojavljuje između elemenata. "name", "orderid", i "shipdate" su sve djeca od elementa letter. Sljedeće schema deklarira "letter" element:

```
<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>
```

Primjedba: Omogućiti znakovne podatke da se pojavljuju između elemenata djece od elementa letter, znači da mixed atribut treba bit postavljen u istinu (true). Tag

<xs:sequence> znači da se elementi (name, orderid i shipdate) moraju pojavljivati u tom redoslijedu unutar "letter" elementa.

Može se također dati complexType elementu ime i načiniti da "letter" element ima atribut koji referencira ime od complexType (ako se koristi ova metoda,onda nekoliko elemenata može referencirati isti složeni tip):

```
<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>
```

Indikatori

S indikatorima se upravlja KAKO se elementi koriste u dokumentima.

Ima nekoliko vrsta indikatora:

Indikatori poretka:

- All
- Choice
- Sequence

Indikatori pojavka:

- maxOccurs
- minOccurs

Indikatori skupine:

- Group name
- atributGroup name

Indikatori poretka

Indikatori poretka koriste se za definiranje kako se elementi trebaju pojavljivati.

„All" Indikator

<all> indicator specificira pretpostavljenu vrijednost da elementi djece mogu se pojavljivati u bilo kojem rasporedu i da svaki element djeteta može se pojaviti jednom i samo jednom:

```
<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>
```

Primjedba: kad se koristi<all> indikator može se postaviti <minOccurs> indikator na 0 ili 1, a <maxOccurs> indikator može se postaviti samo na 1

„Choice" indikator

<choice> indikator određuje hoće li se jedan ili drugi element djeteta pojaviti:

```
<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" indikator

<sequence> indikator specificira da se elementi djeteta mogu pojaviti u zadanom redoslijedu:

```
<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" indikatori

Indikatori poretka se koristi za definiranje kako često se element pojavljuje.

Primjedba: Za sve "Order" i "Group" indikatore (any, all, choice, sequence, group name i group reference) pretpostavljena vrijednost za maxOccurs i minOccurs je 1!!!!!

„maxOccurs" indikator

<maxOccurs> indikator specificira maximalni broj puta pojavljivanja elementa:

```
<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>
```

Gornji primjer pokazuje da se "child_name" element može najmanje jednom pojaviti (pretpostavljena vrijednost za minOccurs je 1), a najviše deset puta pojaviti u elementu "person".

„minOccurs" indikator

<minOccurs> indikator specificira minimalni broj pojavaka nekog elementa:

```
<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>
```

Gornji primjer pokazuje da se "child_name" element može najmanje niti jednom pojaviti, a najviše deset puta pojaviti u elementu "person".

Da se dopusti nekom elementu pojavak neograničen broj puta, treba koristiti naredbu maxOccurs="unbounded".

Radni primjer:

XML datoteka nazvana "Myfamily.xml":

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<persons xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="family.xsd">
<person>
<full_name>Hege Refsnes</full_name>
<child_name>Cecilie</child_name>
</person>
<person>
<full_name>Tove Refsnes</full_name>
<child_name>Hege</child_name>
<child_name>Stale</child_name>
<child_name>Jim</child_name>
<child_name>Borge</child_name>
</person>
<person>
<full_name>Stale Refsnes</full_name>
</person>
</persons>
```

Gornja XML datoteka sadrži korjenski element nazvan "persons". Unutar ovog korjenskog elementa definirano je nekoliko "person" elemenata. Svaki "person" element mora sadržavati "full_name" element djeteta i može sadržavati do pet "child_name" elemenata djeteta.

Ovdje je datoteka schema-e "family.xsd":

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
<xs:element name="persons">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="person" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="full_name" type="xs:string"/>
            <xs:element name="child_name" type="xs:string"
            minOccurs="0" maxOccurs="5"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```