Validacija
XML Schema

Šta je XML Schema?

XML Schema predstavlja (XML) dokument koji sadrži opis strukture i pravila koja čine validan XML dokument.
XML dokument ako zadovoljava sva ograničenja zadata šemom, je validan za tu šemu.

XML Schema dokumenti imaju ekstenziju **.xsd** - pišu se u XSD (*XML Schema Definition*) jeziku.

XML Schema:

- omogućava validaciju XML dokumenta prema detaljnoj specifikaciji
- podržava sintaksu XML-a
- omogućava tipizaciju podataka i prikaz ograničenja
- koristi prostor imena (namespace)
- prezentuje veze koje postoje između elemenata

XML Schema podržava tipove podataka

- Jedna od najvažnijih osobina je podrška tipovima podataka.

- XSD omogućava lakšu primenu za:
- opis dozvoljenog sadržaja tj tipova
- validnost podataka
- rad sa podacima iz baza
- definisanje ograničenja za podatke
- definisanje složenih uzoraka podataka
- konverzije između tipova podataka

XML Schema daje veću sigurnost

Sadrži ekplicitan opis.

- Na primer datum: "03-11-2004" će u nekim zemljama biti prihvaćen kao 3.Novembar a u drugim 11.Mart.
- Medjutim, XML element sa tipom podataka “date”:
- <date type="date">2004-03-11</date>
- može da zahteva format "YYYY-MM-DD“ , i na taj način se isključuju mogući problemi

DTD i XML Schema

DTD deklaracija elementa *količina*

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

Deklaracija elementa *količina* u XML Schema

```
<xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
    <xsd:element name='količina' type=“xsd:nonNegativeInteger"/>
</xsd:schema>
```

Na osnovu DTD deklaracije validan je XML kod:

<količina>3</količina>
<količina>-12</količina>
<količina>malo</količina>

Na osnovu XML Scheme validan je samo prvi primer.

Struktura opisana XML Schemom

- <?xml version="1.0"?>

<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema >
...
...
</xs:schema>

Atributi elementa ***schema***

- <?xml version="1.0"?>
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
- targetNamespace=" http://www.mytestschema.rs "
- xmlns="http://www.mytestschema.rs"
- elementFormDefault="qualified">
- </xs:schema>

- Označava da su elementi i tipovi podataka korišćeni u ovoj šemi iz prostora imena "http://www.w3.org/2001/XMLSchema" namespace, za koji će se koristiti prefiks **xs**
- Označava da su elementi definisani sa ovom šemom (note, to, from, heading, body.) iz prostora imena "http://www.mytestschema.rs".
- Označava da je podrazumevani prostor imena "http://www.mytestschema.rs".
- Označava da elementi korišćeni od XML dokumenata moraju biti kvalifikovani.

Pridruživanje šeme XML dokumentu

1. način. Atribut **xsi:schemaLocation** sadrzi dva tokena. Prvi token sadrži URI odredišnog prostora imena. Drugi je fizička lokacija.

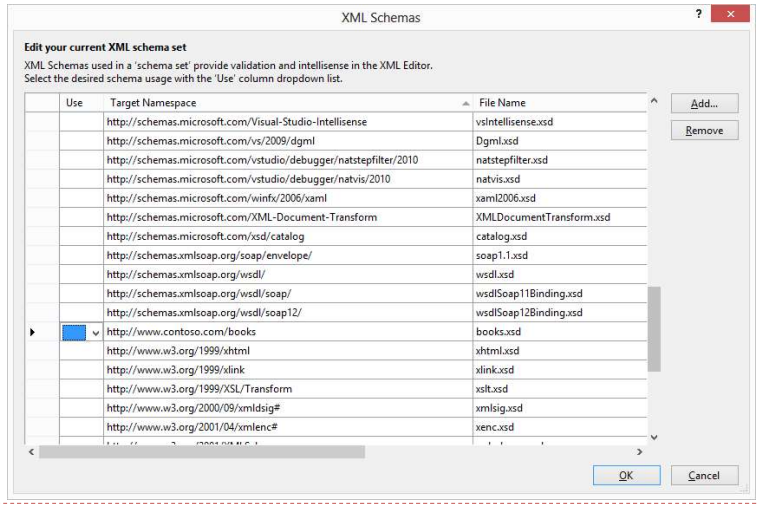
```
<stylesheet xmlns="http://www.w3.org/1999/XSL/Transform"
xmlns:html="http://www.w3.org/1999/xhtml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3.org/1999/xhtml http://www.w3.org/1999/xhtml.xsd" >
```

1. korišćenjem atributa **xsi:noNamespaceSchemaLocation.** *Sadrži adresu šeme koju treba upotrebiti za proveru validnosti elemenata koji ne pripadaju nijednom prostoru imena.*

2. način. **Izdavanjem komande nekom procesoru** za proveru validnosti da proveri validnost datog dokumenta u odnosu na eksplicitno zadatu šemu (a pri tome da zanemari sva uputstva u samom dokumentu)

9 6.11 po podne

- 2. nacin (MS VS) šeme uključene u proveru validnosti tekućeg xml fajla



10 6.11 po podne

Primer povezivanja (1)

```
<?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>
```

Označava podrazumevani prostor imena.
Kada je xmlns:xsi na raspolaganju može se koristiti atribut *schemaLocation*.
Prva vrednost ovog atributa je korišćeni prostor imena. Druga vrednost je lokacija XML scheme korišćene u tom prostoru imena.

Primer povezivanja (2)

```
<?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>
```

Nad elementom note vazi:

1. podrazumevani prostor imena
2. **nad svim podelementima i nad elementom koji su u naznacenom prostoru imena primenjuje se navedena schema**

Primer 1. Element koji ne pripada im. prostoru

- Dobro formiran XML dokument. *fullName* ne pripada imenskom prostoru.
- <?xml version="1.0"?> <fullName>Scott Means</fullName>

Ako "fullName" može da sadrži jednostavan znakovni niz, šema će izgledati:
adr-schema.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="fullName" type="xs:string"/>
</xs:schema>
```

PRIDRUŽIVANJE!
fullName ne pripada nekom prostoru imena: atribut noNamespaceSchemaLocation se primenjuje:

```
<?xml version="1.0"?>
<fullName xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="adr-schema.xsd">Scott Means</fullName>
```


Primer 2. opis *note* elementa: DTD vs XSD

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

note.xsd

```
<?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>
```

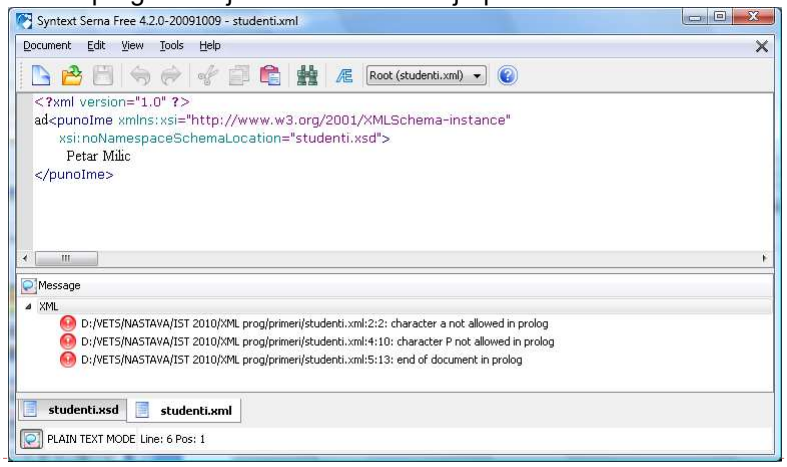
Primena na XML dokument: DTD vs. XSD

```
<?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>
```

note.xml

```
<?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>
```


Za proveru validnosti XML dokumenta u odnosu na šemu, potreban je procesor ili program koji će obaviti validaciju po osnovu šeme.



XSD
ograničena i implementacija

Deklaracije elemenata

Deklaracija elementa u šemi:

<xs:element name="ime" type="xs:tip">

Sadržaj elementa može biti:
1. prost
2. složen.

Prost sadržaj se sastoji od teksta bez ugnježdenih elemenata. U tabeli su navedeni neki najčešće korišćeni prosti tipovi definisani u W3C specifikaciji:

anyURI	URI identifikator resursa		duration	Vremenski period
boolean	true ili false		integer	Ceo broj
dateTime	datum i vreme zajedno		string	Unicode znakovni niz
date time	Datum Vreme		decimal	Decimalna vrednost

Ako je nešto prostog tipa ne može imati atribute. Ima samo ime i tip!

Deklaracije atributa

Atributi moraju biti deklarisani kao prosti tipovi.
Elementi koji sadrže atribute mogu biti složenog tipa.

Atributi se deklarišu pomoću elementa:

<xs:attribute name="ime" type="xs:tip">

Primer:

```
<xs:attribute name="id" type="xs:integer"/>     - definisanje atributa
<proizvod id="135">stolica</proizvod>           - upotreba atributa
```

Atributi su podrazumevano opcioni. Ako je obavezan: **use="required"**

Primer:

```
<xs:attribute name=”id” type=“xs:integer” use=“optional”/>
<xs:attribute name=”id” type=“xs:integer” use=“required”/>
```


- Atributi se deklarišu:
  - globalno – unutar elementa najvišeg nivoa i tada mogu biti referencirani na bilo kom mestu u šemi
  - lokalno – kao deo definicije složenog tipa koja je pridružena određenom elementu

Ograničenja

Kontrolišu moguće vrednosti prostih tipova podataka. Koristi se sintaksa:

xs:restriction

Jedno ograničene se iskazuje kao podelement unutar ograničenja. Više ograničenja se može kombinovati da bi se dodatno ograničile moguće vrednosti prostog tipa.

Šeme koriste sledeće tipove *faseta*:

length (ili minLength i maxLenght)	određuju dužinu podataka
pattern	zadaje sofisticirana ograničenja formata stringa
enumeration	ograničavaju moguće vrednosti na članove unapred definisane liste
whiteSpace	određuje način na koji se obrađuju beline unutar podataka
maxInclusive i maxExclusive	određuju najveće vrednosti stavki
minInclusive i minExclusive	određuju najmanje vrednosti stavki
totalDigits	ograničava ukupan broj cifara u decimalnom broju
fractionDigits	zadaje obavezan broj cifara u broju desno od decimalne tačke

***xs:minInclusive i xs:minExclusive*,**
xs:maxInclusive i xs:maxExclusive

Definišu najmanje, odnosno najveće vrednosti podataka. Razlika je u tome da li se data vrednost smatra delom skupa dozvoljenih vrednosti ili ne.

Primer1:

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

Primer2:Ovo su ekvivalentne deklaracije.

```
<xs:maxInclusive value="0"/>
<xs:maxExclusive value="1"/>
```

xs:enumeration

Ograničava moguće vrednosti elemenata i atributa na članove unapred definisane liste.

Primer:

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


xs:pattern

Ograničenje uzorka koristi regularni izraz za definisanje mogućih vrednosti.

Primer1:

```
<xs:simpleType name="bso">
    <xs:restriction base="xs:string">
        <xs:pattern value="\d\d\d-\d\d-\d\d\d\d"/>
    </xs:restriction>
</xs:simpleType>
```

Definiše da broj socijalnog osiguranja mora sadržati tri cifre, zatim crticu, dve cifre, novu crticu i još četiri cifre.

Primer2:

```
<xs:element name="pol">
<xs:simpleType>
    <xs:restriction base="xs:string">
        <xs:pattern value="muski|zenski">
    </xs:restriction>
</xs:simpleType>
</xs:element>
```

Definiše da pol može biti muški ili ženski

Regularni izrazi

.	Odgovara jednom, bilo kom, karakteru. Ako se koristi u izrazima u zagradma, tačka odgovara literalnoj tački.Na primer, a.c odgovara "*abc*", etc., ali [a.c] odgovara samo "*a*", ".", or "*c*".
[ ]	Odgovara karakteru koji je sadržan u zagradama. Na primer, [abc] odgovara "*a*", "*b*", or "*c*". [a-z] specificira sve karaktere od "*a*" do "*z*". Ovi oblici se mogu mešati: [abcx-z] odgovara "*a*", "*b*", "*c*", "*x*", "*y*", or "*z*", kao i [a-cx-z]. Karakter – se tretira kao literalni karakter ako je na poslednji ili prvi (posle ^) karakter unutar zagrada: [abc-], [-abc]. Zapazite da backslash escapes nisu dozvoljeni. Karakter] može biti uključen u izraz u zagradama ako je prvi karakter (posle ^): []abc].
[^]	Odgovara jednom karakteru koji nije sadrzan unutar zagrada. Na primer, [^abc] odgovara bilo kom karakteru drugačije od "*a*", "*b*", ili "*c*". [^a-z] odgovara bilo kom karakteru koji nije malo slovo u opsegu od "*a*" do "*z*". Kao i gore, literalni karakteri i opsezi mogu se mešati.
^	Odgovara početnoj poziciji u stringu. U line-based alatkama, odgovara startnoj poziciji u svakoj liniji.
$	Odgovara krajnjoj poziciji stringa ili poziciji pre završetka nove linije. U line-based alatkama odgovara krajnjoj poziciji u bilo kojoj liniji.
\d	Odgovara jednoj cifri
*	Prethodni element se pojavljuje nula ili više puta. Na primer, ab*c odgovara "*ac*", "*abc*", "*abbbc*", itd. [xyz]* odgovara "", "*x*", "*y*", "*z*", "*zx*", "*zyx*", "*xyzzy*", i tako dalje. (ab)* odgovara "", "*ab*", "*abab*", "*ababab*", itd.
{m,n}	Odgovara pojavljivanju prethodnog elementa bar *m* puta i ne više od *n* puta. Na primer a{3,5} odgovara samo "*aaa*", "*aaaa*", and "*aaaaa*".


```
/(Mary)( )(had)/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

```
/[a-z]a/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

```
/[^a-z]a/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

```
/A+B*C?D/

AAAD
ABBBBCD
BBBCD
ABCCD
AAABBBC
```

```
/a{5} b{,6} c{4,8}/

aaaaa bbbbb ccccc
aaa bbb ccc
aaaaa bbbbbbbbbbbbb ccccc

/a+ b{3,} c?/

aaaaa bbbbb ccccc
aaa bbb ccc
aaaaa bbbbbbbbbbbbb ccccc

/a{5} b{6,} c{4,8}/

aaaaa bbbbb ccccc
aaa bbb ccc
aaaaa bbbbbbbbbbbbb ccccc
```

```
/.*/

Special characters must be escaped.*

/\.\*/
Special characters must be escaped.*
```

```
/a/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.

/Mary/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

```
/^Mary/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.

/Mary$/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

```
/.a/

Mary had a little lamb.
And everywhere that Mary
went, the lamb was sure
to go.
```

.at	odgovara bilo kom stringu od tri karaktera koji se završava sa "at", na primer "hat", "cat", i "bat".
[hc]at	odgovara "hat" i "cat".
[^b]at	odgovara svim stringovima kao i .at sa izuzetkkom "bat".
[^hc]at	odgovara svim stringovima kao .at ali drugačijim od "hat" i "cat".
^[hc]at	odgovara "hat" i "cat", ali samo na početku nekog stringa ili linije.
[hc]at$	odgovara "hat" i "cat", ali samo na kraju stringa ili linije.
\[.\]	odgovara jednom karakteru koji je okružen sa "[" i "]", na primer: "[a]" i "[b]".

xs:length - Ograničavanje dužine. Definiše/zadaje tačno dužinu.

xs:minLength, xs:maxLength - Zadaju opseg dužina (najmanju i najveću) za vrednosti datog tipa.

Primer1:

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

Primer2:

```
<xs:element name="password">
        <xs:simleType>
      <xs:restriction base="xs:string">
        <xs:minLength value="5"/>
        <xs:maxLength value="8"/>
      </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Složeni tipovi

Šema pridružuje tip svakom elementu i atributu koji deklariše.
Elementi koji definišu složene tipove mogu imati atribute i mogu sadržati ugnježdene elemente. Samo elementi mogu biti složenog tipa. Tipovi atributa su uvek prosti.

Deklaracija složenog tipa: **xs:complexType**

XML kod

```
<zaposleni>
  <ime>Pera</ime>
  <prezime>Miric</prezme>
</zaposleni>
```

XML Schema kod

```
<xs:element name="zaposleni">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ime" type="xs:string"/>
      <xs:element name="prezime" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```


Element može sadržati i **referencu** na složeni tip (ili ga sadržati).

```
<xs:element name="zaposleni" type="osobainfo"/>

<xs:complexType name="osobainfo"/>
  <xs:sequence>
    <xs:element name="ime" type="xs:string"/>
    <xs:element name="prezime" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Na ovaj način se može lako definisati više elemenata istog složenog tipa.

```
<xs:element name="zaposleni" type="osobainfo"/>
<xs:element name="student" type="osobainfo"/>
<xs:element name="clan" type="osobainfo"/>

<xs:complexType name="osobainfo"/>
  <xs:sequence>
    <xs:element name="ime" type="xs:string"/>
    <xs:element name="prezime" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Složeni element može proširiti neki postojeći složeni element novim elementima.

```
<xs:element name="zaposleni" type="dopunjeninfo"/>
<xs:complexType name="osobainfo"/>
  <xs:sequence>
    <xs:element name="ime" type="xs:string"/>
    <xs:element name="prezime" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

```
<xs:complexType name="dopunjeninfo"/>
  <xs:complexContent>
    <xs:extension base="osobainfo">
      <xs:sequence>
        <xs:element name="ulica" type="xs:string"/>
        <xs:element name="grad" type="xs:string"/>
        <xs:element name="drzava" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```


Referisanje na element

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

<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>
```

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

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
    <xs:element ref="to"/>
    <xs:element ref="from"/>
    <xs:element ref="heading"/>
    <xs:element ref="body"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<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:schema>
```

Ograničenja broja pojavljivanja

Eksplicitnog zadavanja najmanjeg i najvećeg broja pojavljivanja nekog elementa na nekom mestu u dokumentu.

Koriste se sledeći atributi elementa xs:element:

1. **minOccurs** – definiše minimalan broj pojavljivanja elementa
2. **maxOccurs** – definiše maksimalan broj pojavljivanja elementa

Podrazumevan vrednost za oba atributa je 1.

```
<xs:element name="student">
  <xs:complexType>
    <xs:sequence>
      <xs:element name= "brIndeks" type="xs:string" />
      <xs:element name= "telefon" type="xs:string"
      minOccurs= "1"  maxOccurs= "10" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```


Mešoviti (*mixed*) sadržaj elemenata

Mešoviti sadržaj imaju elementi koji sadrže stringove, tj. tekst ali i druge elemente.

XML Schema omogućava definisanje ove funkcionalnosti kao i precizno definisanje broja i redosleda elemenata unutar znakovnih podataka.

Atribut **mixed** elementa tipa **complexType** određuje da li se znakovni podaci smeju pojaviti unutar tela elementa.

```
<pismo>
 Dragi<ime> Pero</ime>
 Vasa narudzbina <narid>1432 </narid>
 ce stici<nardatum> 2007-06-18</nardatum>
</pismo>
```

```
<xs:element name="pismo" type="pismoTip"/>
<xs:complexType name="pismoTip" mixed="true">
  <xs:sequence>
    <xs:element name="ime" type="xs:string"/>
    <xs:element name="narid" type="xs:positiveInteger"/>
    <xs:element name="nardatum" type="xs:date"/>
  </xs:sequence>
</xs:complexType>
```

Zadavanje redosleda elemenata

Za definisanje redosleda elemenata, XML šema nudi:

1. **xs:sequence**
2. **xs:choice**
3. **xs:all**

Ove mogućnosti se mogu ugnježdavati.

xs:sequence

Ovaj element definiše koje elemente sadrži složeni element i tačan redosled.

Pismo mora sadržati element *pozdrav*, element *telo* i element *završetak*, tim redom.

```
<xs:element name="pismo">
  <xs:complexType  mixed="true">
    <xs:sequence>
      <xs:element name="pozdrav" />
      <xs:element name="telo" />
      <xs:element name="zavrsetak" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

xs:choice

Definiše pojavljivanje samo jednog elementa iz niza mogućnosti.

Pozdrav mora da sadrži samo jednu iz liste dozvoljenih pozdravnih reči (*zdravo, cao ili draga*).

```
<xs:element name="pozdrav">
  <xs:complexType  mixed="true">
    <xs:choice>
      <xs:element name="zdravo" />
      <xs:element name="cao" />
      <xs:element name="draga" />
    </xs:choice>
  </xs:complexType>
</xs:element>
```


xs:all

Definiše da se svaki od elemenata mora pojaviti jednom. Redosled pojavljivanja nije važan.

cirpismo.xsd

```
<xs:element name="telo">
  <xs:complexType  mixed="true">
    <xs:all>
      <xs:element name="stavka"/>
      <xs:element name="datumPrispeca"/>
      <xs:element name="cena"/>
    </xs:all>
  </xs:complexType>
</xs:element>
```

Redosled pojavljivanja elemenata stavka, cena i datumPrispeca nije bitan, a sprečeno je i pojavljivanje više referenci iste vrednosti.

cirpismo.xlm

```
<pismo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="cirpismo.xsd">
   <pozdrav> <zdravo/>Bobo! </pozdrav>
   <telo>
      Hvala Vam sto se narucili <stavka/>
      (<cena/>). Trebalo bi da stigne do <datumPrispeca/>
   </telo>
   <zavrsetak/>
</pismo>
```

using System.Xml;
using System.Xml.Schema;

```
{
    // Create the XmlSchemaSet class.
    XmlSchemaSet sc = new XmlSchemaSet();
    // Add the schema to the collection.
    sc.Add("urn:bookstore-schema", "books.xsd");
    // Set the validation settings.
    XmlReaderSettings settings = new XmlReaderSettings();
    settings.ValidationType = ValidationType.Schema;
    settings.Schemas = sc;
    settings.ValidationEventHandler += new ValidationEventHandler (ValidationCallBack);
    // Create the XmlReader object.
    XmlReader reader = XmlReader.Create("booksSchemaFail.xml", settings); // Parse the file.
    while (reader.Read());
}
// Display any validation errors.
private static void ValidationCallBack(object sender, ValidationEventArgs e)
{ Console.WriteLine("Validation Error: {0}", e.Message); }
```


Inline Schema

```
public static void Main() {
    // Set the validation settings.
    XmlReaderSettings settings = new XmlReaderSettings();
    settings.ValidationType = ValidationType.Schema;
    settings.ValidationFlags |= XmlSchemaValidationFlags.ProcessInlineSchema;
    settings.ValidationFlags |= XmlSchemaValidationFlags.ReportValidationWarnings;
    settings.ValidationEventHandler += new ValidationEventHandler (ValidationCallBack);
    // Create the XmlReader object.
    XmlReader reader = XmlReader.Create("inlineSchema.xml", settings);
    // Parse the file.
     while (reader.Read());
}
private static void ValidationCallBack (object sender, ValidationEventArgs args)
{
    if (args.Severity==XmlSeverityType.Warning)  Console.WriteLine("\tWarning: Matching schema not found. No validation
    occurred." + args.Message);
     else Console.WriteLine("\tValidation error: " + args.Message);
}
```