

ՅՈՒՐԱԿԱՆ ԵՐԱՆԻՍՈՒԹՅԱՆ ԵՐԱՆՈՒՄ

ԵՐԱՆՈՒՄ, ԵՐԱՆՈՒՄ

ՄԱՍԻՆ ԵՐԱՆՈՒՄ

Otvoreno računarstvo

XML - DTD, XSD

- Document Type Definition (DTD)
 - Građevne komponente
 - Elementi, atributi, entiteti,...
 - Deklariranje tipa dokumenata
- XML Schema Definition (XSD)
 - Prilog: XSD

Mario Žagar





DTD

Document Type Definition

Tipovi dokumenata

- Dokument XML
 - označava sadržaj i određuje hijerarhijsku strukturu
- Bez propisanog tipa dokumenta
 - oznake i hijerarhijska struktura (gnježđenje) su potpuno nedefinirane i mogu se koristiti potpuno proizvoljno
- Definicija tipa dokumenta mora odrediti:
 - oznake koje se mogu koristiti (konačan broj)
 - sve moguće hijerarhijske strukture koje se u dokumentu mogu pojaviti (ne mora biti konačan broj)
- Tip dokumenta
 - eksplicitno ne određuje značenje oznaka i struktura, no podrazumijeva da to znanje dijele korisnici dokumenata

Valjani XML dokumenti

- Definicija tipa dokumenta efektivno definira jezik oznaka
 - riječi + tvorba rečenica + (implicitno) značenje
- DTD je jezik za definiranje opisnih jezika
 - meta-jezik
- Ako je XML dokumentu pridijeljen tip, parser:
 - koristi definiciju tipa za provjeru rječnika dokumenta, vrijednosti elemenata i atributa te strukture dokumenta
- **XML dokument koji zadovoljava provjeru tipa dokumenta je valjan** (eng. valid)
- Valjan XML dokument mora biti dobro oblikovan
- Dobro oblikovan XML dokument ne mora biti valjan

Zašto provjera valjanosti?

- Zašto uopće koristiti provjeru valjanosti dokumenta?
 - “provjera valjanosti je obična gnjavaža”
 - definiranje dobrog tipa dokumenta, koji će biti koristan dulje vrijeme, zahtijeva puno vremena i razmišljanja
- ali:
 - tjera na pridržavanje normi, preporuka, dogovora
 - sprječava velike probleme tijekom korištenja, u komunikaciji i suradnji različitih sustava, potencijalno iz različitih organizacija
 - identificira krivca za probleme u komunikaciji

Građevne komponente

- Građevne komponente XML dokumenta (po DTD-u):
 - Elementi
 - rječnik i struktura tipa dokumenta
 - Entiteti
 - građevne komponente XML dokumenta
 - Atributi
 - dodatni podaci o elementima
 - Parsirani znakovni podaci (PCDATA)
 - tip vrijednosti elementa, parser obrađuje sadržaj u potrazi za elementima i entitetima
 - Znakovni podaci (CDATA)
 - kao PCDATA, tip vrijednosti atributa



Elementi

Deklaracija elemenata

- Elementi se deklariraju ključnom riječi ELEMENT

<!ELEMENT ime-elementa (sadržaj)>

- Pet vrsta sadržaja:
 - prazan
 - jednostavan
 - slobodan
 - elementi s djecom
 - miješani

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Prazan sadržaj



<!ELEMENT ime-elementa EMPTY>

Primjer:

<!ELEMENT hLine EMPTY>

<hLine> </hLine> ili
<hLine/>

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Jednostavan sadržaj



- Sadržaj elementa čine isključivo parsirani znakovni podaci

<!ELEMENT ime-elementa (#PCDATA)>

- Primjer:

<!ELEMENT lastName (#PCDATA)>

<lastName>Ferković – Enter</lastName>

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Slobodan sadržaj

- Sadržaj elementa
 - čini bilo koja kombinacija podataka koja zadovoljava uvjete dobrog oblikovanja
 - služi za postavljanje točaka proširenja unutar tipa dokumenta

<!ELEMENT ime-elementa ANY>

- Primjer:

<!ELEMENT misc ANY>

<misc>

<NSK> <posudba><datum> ...

</NSK>

</misc>

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Elementi s djecom



<!ELEMENT element-roditelj (elementi-djeca)>

<!ELEMENT html (head, body)>

<html>

<head> ... </head>

<body> ... </body>

</html>

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

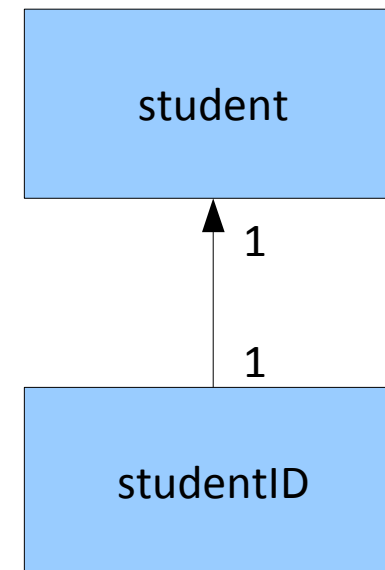
Jedan element dijete



<!ELEMENT element-roditelj (element-dijete)>

<!ELEMENT student (studentID)>

```
<student>
  <studentID>
    ...
  </studentID>
</student>
```

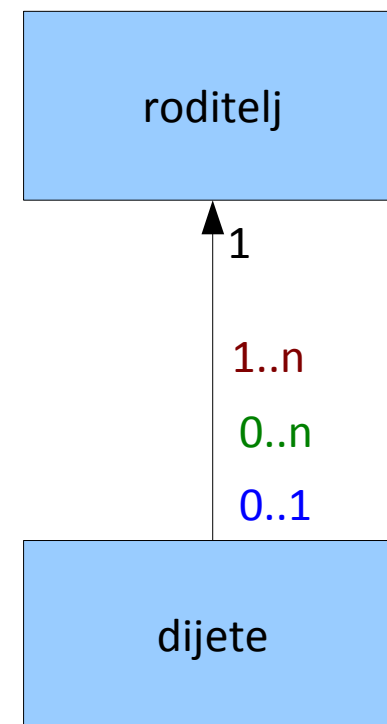


- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Indikatori kardinalnosti elementa



- Točno jedan element – ne navodi se indikator (prethodni primjer)
- Najmanje jedan element – indikator **+**
`<!ELEMENT roditelj (dijete+)>`
- Nijedan ili više elemenata – indikator *****
`<!ELEMENT roditelj (dijete*)>`
- Nijedan ili jedan element – indikator **?**
`<!ELEMENT roditelj (dijete?)>`



- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Kombiniranje više elemenata djece



- Operator slijeda elemenata ','
 - u dokumentu moraju biti navedeni u definiranom slijedu:
`<!ELEMENT roditelj (dijete1, dijete2, dijete3)>`
- Operator alternative (XOR) '|'
 - ili jedan ili drugi element
`<!ELEMENT roditelj (dijete1 | dijete3)>`
- Operator grupiranja ()
 - niz elemenata unutar zagrada tretira se kao jedan element kod primjene ostalih operacija i indikatora
`<!ELEMENT roditelj ((dijete1 | dijete2)+, dijete3)>`

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Primjeri (I)

- Podaci o studentu

`<!ELEMENT student (studentID, firstName, lastName)>`

- Adresa je opcionalan element

`<!ELEMENT student (studentID, firstName, lastName, address?)>`

- Nebitan poredak imena i prezimena

`<!ELEMENT student (studentID, ((firstName, lastName) | (lastName, firstName)), address?)>`

- Upisani predmeti

`<!ELEMENT student (studentID, ((firstName, lastName) | (lastName, firstName)), address?, courses*)>`

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Primjeri (II)

- Element **y** može se unutar elementa **x** pojaviti paran broj puta...

`<!ELEMENT x ((y,y)*)>`

- ... ili neparan broj puta...

`<!ELEMENT x (y,(y,y)*)>`

- ... ili 3 ili 5 puta...

`<!ELEMENT x (y,y,y,(y,y)?)>`

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Miješani sadržaj



`<!ELEMENT element(#PCDATA|element1|element2|..)*>`

- Sadržaj elementa mogu činiti tekst (jednostavan sadržaj) ili nabrojani elementi
- Ne može se definirati redoslijed navođenja elemenata, kao ni broj pojavljivanja elemenata
- Izbjegavati korištenje ovog tipa sadržaja
 - Elementi
 - Atributi
 - Entiteti
 - PCDATA
 - CDATA

Hijerarhija elemenata

- Oznakom **<!ELEMENT >** deklariraju se dozvoljena imena i hijerarhija elemenata u tipu dokumenta
 - Unutar jednog DTD-a može biti definirano više neovisnih hijerarhija
 - Definirana hijerarhijska struktura mora slijediti strukturu stabla:
 - korijenski element (nema čvora roditelja)
 - do svih ostalih elemenata stabla postoji put od korijenskog elementa
 - Elementi unutar DTD-a mogu biti navođeni proizvoljnim redoslijedom
- Elementi
 - Atributi
 - Entiteti
 - PCDATA
 - CDATA



Entiteti

Entiteti

- Unutarnji opći entiteti
`<!ENTITY ime "vrijednost">`
`<!ENTITY fer "Fakultet elektronike i računovodstva">`
`<inst>&fer;</inst>`
- Unutarnji opći ugrađeni entiteti
`$4 > 2$`

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Vrste entiteta

- **Opći** (referencirani iz XML dokumenta) ili
 - **Parametarski** (referencirani iz DTD-a)

 - **Unutarnji** (definirani unutar DTD dokumenta) ili
 - **Vanjski** (definirani u zasebnoj datoteci)

 - **Parsirani** – tekstualni, prolaze parsiranje ili
 - **Neparsirani** – binarni, tekstualni – ne prolaze parsiranje
- Elementi
 - Atributi
 - Entiteti
 - PCDATA
 - CDATA



Atributi

Atributi



```
<!ATTLIST element atribut tip default-vrijednost>
```

```
<!ATTLIST element atribut1 tip default-vrijednost  
          atribut2 tip default-vrijednost  
          ...  
          atributN tip default-vrijednost >
```

```
<!ATTLIST course courseID CDATA "-">
```

```
<course courseID="479"> ... </course>
```

- Element čiji se atribut definira mora biti definiran
- Ime atributa mora biti jedinstveno za pojedini element

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Tipovi atributa



<!ATTLIST element atribut tip default-vrijednost>

CDATA	neparsirani znakovni podaci
(en1 en2 en3...)	jedna od vrijednosti iz liste (enumeracija)
ID	jedinstveni identifikator
IDREF	vrijednost postojećeg jedinstvenog identifikatora
IDREFS	lista vrijednosti postojećih jedinstvenih identifikatora
NMTOKEN	valjano ime u XML-u
NMTOKENS	lista valjanih imena u XML-u
ENTITY	entitet
ENTITIES	lista entiteta

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Podrazumijevane vrijednosti



<!ATTLIST element atribut tip default-vrijednost>

<i>vrijednost</i>	podrazumijevana vrijednost atributa
#REQUIRED	obavezna vrijednost atributa
#IMPLIED	neobavezna vrijednost atributa
#FIXED <i>vrijednost</i>	nepromjenljiva vrijednost

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

CDATA tip atributa

- neparsirani niz znakova

```
<!ELEMENT course (title+, lecturer, contents?)>
<!ATTLIST course status CDATA "undergrad">
```

- Vrijednost atributa status navedena:

```
<course status="module"> ... </course>
```

- Vrijednost atributa nije navedena, nakon parsiranja atributu će biti pridjeljena vrijednost 'undergrad'

```
<course> ... </course>
```

- nakon parsiranja:

```
<course status="undergrad"> ... </course>
```

- Elementi
- Atributi
- Entiteti
- PCDATA
- CDATA

Enumerirani tip atributa

- Atribut može poprimiti samo jednu od vrijednosti navedenih u deklaraciji enumeriranog tipa

`<!ELEMENT course (title+, lecturer, contents?)>`

`<!ATTLIST course status (undergrad|orientation|graduate|postgrad|module|elective|external) "undergrad">`

Vrijednost atributa status navedena:

`<course status="modul"> ... </course>`

Vrijednost atributa nije navedena, nakon parsiranja atributu će biti pridjeljena vrijednost 'undergrad'

`<course> ... </course>`

NMTOKEN, NMTOKENS

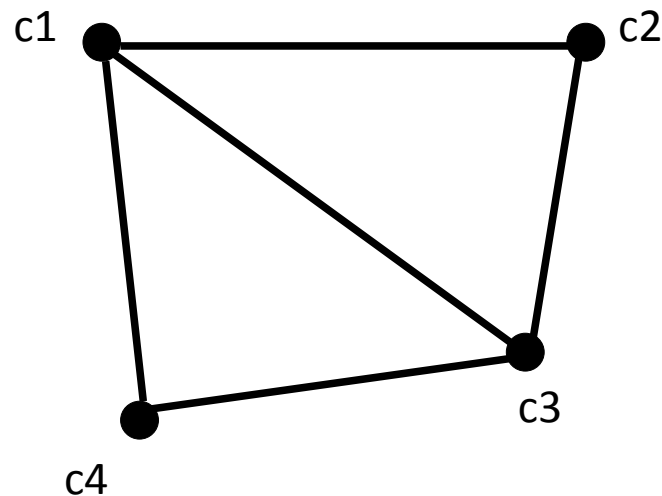
- **NMTOKEN**
 - valjano ime elementa ili atributa
 - **U stvaranju vrijednosti ovih atributa**
 - vrijede ista pravila kao i za stvaranje imena elemenata i atributa (npr. ime se može sastojati samo od jedne riječi, mora početi sa slovom ...)
 - **NMTOKENS**
 - jedan ili više NMTOKEN vrijednosti odvojenih razmakom
- <!ELEMENT title (#PCDATA)>
- <!ATTLIST title lang NMTOKEN #REQUIRED>
- <title lang="hr"> Otvoreno računarstvo</title>

ID, IDREF, IDREFS

- Vrijednosti kao NMTOKEN, NMTOKENS
- U terminologiji baza podataka:
 - ID = primarni ključ, IDREF = strani ključ, IDREFS = niz stranih ključeva odvojenih razmakom
- Parser provjerava
 - jedinstvenost vrijednosti u atributima tipa ID na razini dokumenta
 - postojanje vrijednosti atributa ID navedenih u atributima IDREF i IDREFS, na razini dokumenta
- Način stvaranja i očuvanja integriteta logičke strukture podataka unutar XML dokumenta
- Element može imati samo jedan atribut tipa ID
 - uobičajeno se koristi podrazumijevana vrijednost #REQUIRED

DTD za opis grafa

- Ne možemo koristiti strukturu XML dokumenta za opis strukture grafa (stablo je podvrsta grafa)



Primjer ID, IDREFS

```
<!ELEMENT graf (cvor)+>
<!ELEMENT cvor EMPTY>
<!ATTLIST cvor id ID #REQUIRED>
<!ATTLIST cvor susjedi IDREFS #IMPLIED>
```

```
<graf>
  <cvor id="c1" susjedi="c2 c3 c4" />
  <cvor id="c2" susjedi="c1 c3" />
  <cvor id="c3" susjedi="c1 c2 c4" />
  <cvor id="c4" susjedi="c1 c3" />
</graf>
```

primarni ključevi

strani ključevi

Podrazumijevana vrijednost

- Ako je vrijednost atributa navedena u XML dokumentu, nakon parsiranja atribut poprima tu, navedenu vrijednost
- Ako vrijednost atributa nije navedena, nakon parsiranja atributu će biti pridjeljena podrazumijevana vrijednost

```
<!ELEMENT course (title+, lecturer, contents?)>
<!ATTLIST course status CDATA 'opći'>
```

```
<course> ... </course>
```

```
<course status="module"> ... </course>
```

Obavezna vrijednost

- Vrijednost atributa se mora navesti u XML dokumentu
- Izostavljanje vrijednosti rezultira greškom prilikom parsiranja dokumenta

```
<!ELEMENT course (title+, lecturer, contents?)>
<!ATTLIST course status CDATA #REQUIRED>
```

```
<course> ... </course>      greška/grješka :-) !
```

```
<course status="undergrad"> ... </course>
```

Neobavezna vrijednost

- Vrijednost atributa se može navesti u XML dokumentu
- Izostavljanje vrijednosti rezultira nepostojanjem tog atributa nakon parsiranja

```
<!ELEMENT course (title+, lecturer, contents?)>
<!ATTLIST course status CDATA #IMPLIED>
```


```
<course> ... </course>
<course status="module"> ... </course>
```

Nepromjenljiva vrijednost

- Vrijednost atributa se može navesti u dokumentu
- Ako se navede, mora se koristiti podrazumijevana vrijednost navedena u deklaraciji atributa
 - navođenje različite vrijednosti - greška u parsiranju
 - nakon parsiranja atribut uvijek ima podrazumijevanu vrijednost

```
<!ELEMENT course (title+, lecturer, contents?)>  
<!ATTLIST course status CDATA #FIXED "undergrad">
```

```
<course> ... </course>  
<course status="undergrad"> ... </course>  
<course status="module"> ... </course> greška!
```



Deklariranje tipa XML dokumenta

Deklaracija tipa dokumenta

- XML dokument
 - može i ne mora deklarirati svoj tip unutar prologa dokumenta
- Ako je tip deklariran
 - (validirajući) parser prilikom učitavanja provjerava i dobru oblikovanost i valjanost, javlja greške
- Unutarnja, vanjska i kombinirana definicija:


```
<!DOCTYPE korijenski-element [DTD definicija]>
```

```
<!DOCTYPE korijenski-element SYSTEM URI>
```

```
<!DOCTYPE korijenski-element PUBLIC FPI URI>
```

```
<!DOCTYPE korijenski-element SYSTEM URI [DTD definicija]>
```

```
<!DOCTYPE korijenski-element PUBLIC FPI URI [DTD definicija]>
```

Korijenski-element

 - bilo koji element definiran unutar DTD-a

Unutarnja definicija tipa

- DTD se nalazi unutar XML dokumenta

`<!DOCTYPE korijenski-element [DTD definicija]>`

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>

<!DOCTYPE korijen [
    <!ELEMENT korijen (dijete1, dijete2)>
    ...
]>
<korijen>
    <dijete1>
        ...
    </dijete1>
    ...
</korijen>
```


Vanjske definicije tipova

- DTD se nalazi u zasebnoj datoteci
 - ili u skupu datoteka ako se u definiciji tipa koriste vanjski parametarski entiteti)
- Atribut **standalone** XML deklaracije treba biti postavljen u “**no**”
- Privatna vanjska definicija – koristi se u užem krugu (osobe, organizacija), nije za javnu distribuciju
<!DOCTYPE korijenski-element SYSTEM URI>
- Javna vanjska definicija – namijenjena javnom korištenju (de facto / de iure standard?)
<!DOCTYPE korijenski-element PUBLIC FPI URI>

Formal Public Identifier



standard//odgovoran//tip-dokumenta//jezik

standard:

- (nije standard)
 - + (odobrilo nestandardno tijelo)
- ime-standarda

odgovoran: ime grupe odgovorne za održavanje DTD-a

tip-dokumenta: što definira, verzija dokumenta

jezik: ISO 639 (dva ili tri slova identificiraju jezik)

-//UNIZG-FER-ZARI-RASIP//eIndex Ver. 1.0//HR

-//W3C//DTD HTML 4.0 Transitional//EN

Vanjske definicije tipa

- DTD se nalazi u zasebnoj datoteci
 - za provedbu provjere valjanosti, parser mora dohvatiti datoteku s definicijom

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE korijen SYSTEM "dtd/primjer1.dtd">
...
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE korijen PUBLIC "-//RASIP//eIndex Ver. 1.0//HR"
  "http://www.fer.hr/rasip/dtd/eindex-v1.0.dtd">
...
```

Kombinirana definicija tipa

```
<!DOCTYPE korijenski-element SYSTEM URI [DTD definicija]>
```

```
<!DOCTYPE korijenski-element PUBLIC FPI URI [DTD  
  definicija]>
```

- Dio DTD-a u vanjskoj datoteci, dio unutar XML dokumenta
- Kolizija definicije elemenata:
 - u teoriji: unutarnje definirani elementi imaju prednost (omogućava redefiniciju tipa)
 - u praksi: greška prilikom parsiranja

Usporedba deklaracija

- Unutarnja deklaracija:
 - DTD uvijek dostupan, nije ga potrebno dodatno dohvaćati
- Vanjska deklaracija
 - manje datoteke dokumenta (parser može držati DTD dokumente u priručnoj memoriji)
 - dijeljenje standardne definicije tipa između više korisnika

eIndex DTD (I)

```
<!-- eIndex DTD Definition -->
<!ELEMENT eindex (student, courses, attachments?)>
<!ATTLIST eindex version CDATA #FIXED "1.0">
<!ELEMENT student (studentID, firstName, lastName,
                    POB, DOB, country)>
<!ATTLIST student status (regular|switched|exchange|
                           external) "regular">
<!ELEMENT studentID (#PCDATA)>
<!ELEMENT firstName (#PCDATA)>
<!ELEMENT lastName (#PCDATA)>
<!ELEMENT POB (#PCDATA)>
<!ELEMENT DOB (#PCDATA)>
<!ELEMENT country (#PCDATA)>
<!ELEMENT courses (course*)>
```

eIndex DTD (II)

```

<!ELEMENT course (courseID, title+, ects, lecturer,
  enrollment+)>
<!ATTLIST course status (undergrad|orientation|graduate|
  postgrad|module|elective|external) #REQUIRED>
<!ELEMENT courseID (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ATTLIST title lang NMTOKEN #REQUIRED>
<!ELEMENT ects (#PCDATA)>
<!ELEMENT lecturer (#PCDATA)>
<!ELEMENT enrollment (school-year, successPct)>
<!ATTLIST enrollment semester (winter|summer) #IMPLIED>
<!ATTLIST enrollment grade (A|B|C|D|E|F) #REQUIRED>
<!ELEMENT school-year (#PCDATA)>
<!ELEMENT successPct (#PCDATA)>
<!ELEMENT attachments ANY>

```

Nedostaci DTD-a

- Nema provjere valjanosti podataka kod jednostavnih sadržaja !
- Nužnost provjere nameće korištenje atributa umjesto elemenata za pohranu podataka !
- Slaba provjera tipova podataka kod atributa
- Nespretno riješeno definiranje kardinalnosti
- Nepostojeća podrška za prostore imena
- DTD nije XML (nije dobro oblikovan dokument)

Provjera valjanosti dokumenta

- PHP skripta za provjeru valjanosti dokumenta na osnovu pridruženog DTD-a:

```
$doc = new DOMDocument;  
$doc->Load($argv[1]);  
if( $doc->validate() == TRUE )  
    echo "$argv[1] is a valid xml document\n";
```

```
> php validateDTD.php graf.xml  
graf.xml is a valid xml document
```

(skripta, xml i dtd datoteke moraju biti u istom direktoriju)

XML Schema Definition (XSD)

XSD

- W3C preporuka donesena 2001. godine
- Namjena kao i DTD, ispravlja brojne nedostatke kao što su:
 - definiranje različitih tipova sadržaja elemenata i atributa
 - jednostavnije i preciznije mogućnosti definiranja strukture
 - definiranje novih tipova i formata zapisa podataka
 - podrška prostorima imena
 - Schema je XML, DTD nije
 - ...

XSD struktura



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

    ...

</xsd:schema>
```

- XML Schema dokument mora biti dobro oblikovan i valjan.
- Koriijenski element XSD dokumenta je **schema**.
- Dobra praksa je deklarirati prostor imena za XSD elemente (**xsd**, **xs**, ...) kako se bi došlo do konflikta s imenima elemenata, tipova, atributa itd. jezika kojeg se definira.

Komponente XSD

- Osnovne komponente definicije tipa dokumenta u DTD-u:

elementi `<!ELEMENT name (content)>`

atributi `<!ATTLIST element name type default-value>`

- XSD koristi tri osnovne komponente:

elementi `<xsd:element name="ime" type="tip" ...>`

atributi `<xsd:attribute name="ime" type="tip" ...>`

tipovi `<complexType name="ime" ...>` i

`<simpleType name="ime" ...>`

Tipovi u XSD

- Sadržaj elemenata i atributa u XSD definiran s njihovim tipom
 - tip je ravnopravna komponenta sheme!
 - u DTD-u jednostavni tipovi podržani samo kod atributa
- Općenito, tip je definiran vrstama podataka koje može sadržavati...
 - jednostavan sadržaj u obliku niza znakova
 - elemente (koji su pak određenog tipa...)
 - attribute (također određenog tipa...)

Vrste tipova u XSD

- XSD definira dvije vrste tipova:
 - jednostavne i složene
- Jednostavni tipovi:
 - jednostavan sadržaj, npr. string, broj, URI, datum ...
 - ne sadrže ni elemente ni attribute
 - provjera tipa sadržaja kod parsiranja (npr. element tipa integer ne može imati vrijednost “534ž45”)
 - DTD: element s (#PCDATA) sadržajem, bez atributa
- Složeni tipovi:
 - sadrže elemente i/ili attribute
 - sličnost s ostalim vrstama sadržaja elemenata iz DTD-a

Definiranje tipova u XSD

- Ugrađeni jednostavni tipovi (>40)
- Definiranje novih jednostavnih tipova ograničavanjem (ugrađenih ili novo-definiranih) jednostavnih tipova
- Ne postoje ugrađeni složeni tipovi, moraju se definirati
- Novi složeni tipovi definiraju se:
 - dodavanjem atributa jednostavnim tipovima
 - navođenjem elemenata (i atributa) koji čine složeni tip
 - proširenjem ili ograničenjem definiranih složenih tipova

Povezivanje dokumenta i sheme



- DTD: eksplicitna deklaracija tipa dokumenta
 - `<!DOCTYPE ...`
- Schema ne zahtijeva eksplicitnu deklaraciju:
 - parser koristi deklaracije prostora imena u XML dokumentu
 - XML dokument može koristiti više schema za definiranje sadržaja
- Parser nastoji locirati definiciju sheme (prostora imena)
 - na koji god način umije (kontaktira URI prostor imena, posjeduje lokalnu bazu definicija shema ...)
- Autor dokumenta može dati naputak parseru
 - gdje se nalazi definicija sheme korištenjem atributa `xsi:schemaLocation`, parser nije dužan slijediti taj naputak
- *`xsi:schemaLocation`* navodi parove
 - (URI prostora imena, URI dokumenta def. prostora imena)

Primjer: *schemaLocation* atribut



`<ex:eIndeks`

`xmlns:ex="http://www.fer.hr/eIndeks-Ver1.0"`

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

`xsi:schemaLocation="http://www.fer.hr/eIndeks-Ver1.0`

`http://www.fer.hr/rasip/xsd/eIndex1.0.xsd">`

...

URL eIndex schema dokumenta

URI prostora imena

SchemaInstance prostor
imena u kojem je deklariran
atribut *schemaLocation*

Kada DTD, a kada XML Schema?



- DTD
 - za jednostavnije, kraće dokumente
 - bitna je provjera strukture, same vrijednosti elemenata i atributa nisu toliko bitne
 - mogu se uređivati “ručno”, korištenjem običnih uređivača teksta
- XML Schema
 - za složene dokumente
 - bitna je provjera i strukture i tipova vrijednosti elemenata i atributa
 - definicija duža u odnosu na ekvivalentnu definiciju u DTD-u, prvenstveno namijenjeno uređivanju s pomoću namijenskih alata (grafička sučelja, manipuliranje simbolima umjesto tekстом ...)

Provjera valjanosti dokumenta

- PHP skripta za provjeru valjanosti dokumenta na osnovu pridružene sheme

```
$doc = new DOMDocument();  
$doc->load($argv[1]);  
if( $doc->schemaValidate($argv[2]) == TRUE )  
    echo "$argv[1] is a valid xml document\n";
```

```
>php validateXSL.php eIndex.xml eIndex.xsd  
eIndex.xml is a valid xml document
```

(skripta, xml i xsd datoteke moraju biti u istom direktoriju)



Pitanja?

Prilozi uz XSD:

- Komentar: materijali u prilogu nisu ispitno gradivo



XSD

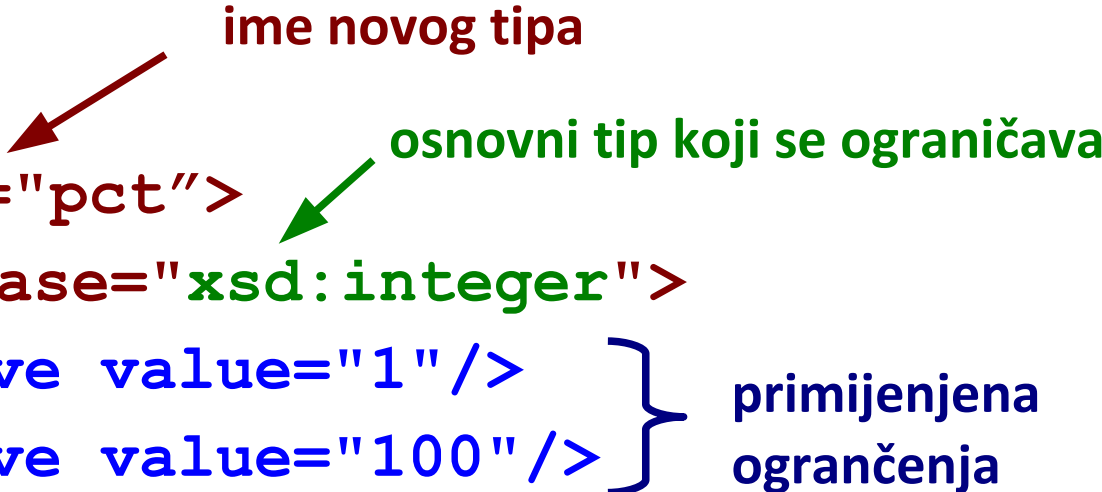
Jednostavni tipovi podataka

Ugrađeni tipovi podataka

- xsd:string – niz znakova (neograničena duljina)
- xsd:integer – cjelobrojna vrijednost
- xsd:decimal – broj u formatu pomičnog zareza
- xsd:boolean – true/false
- xsd:date – datum u formatu “YYYY-MM-DD”
- xsd:time – vrijeme u formatu “hh:mm:ss”
- xsd:anyURI – Uniform Resource Identifier
- xsd:hexBinary – binarni podaci u heksadecimalnom zapisu
- ... > 40

Ograničavanje jednostavnih tipova podataka

- Ograničavanjem jednostavnog tipa podataka definira se novi jednostavni tip podataka
- Ograničavaju se vrijednosti koje novi tip podatka može sadržavati, u ovisnosti o tipu podataka koji se ograničava



```
<xsd:simpleType name="pct">  
  <xsd:restriction base="xsd:integer">  
    <xsd:minInclusive value="1"/>  
    <xsd:maxInclusive value="100"/>  
  </xsd:restriction>  
</xsd:simpleType>
```

primijenjena ograničenja

Ograničenje raspona vrijednosti



```
<xsd:minInclusive value="value"/>
```

```
<xsd:maxInclusive value="value"/>
```

```
<xsd:minExclusive value="value"/>
```

```
<xsd:maxExclusive value="value"/>
```

Tipovi podataka:

- integer
- decimal
- date
- time

```
<xsd:simpleType name="pct">
```

```
  <xsd:restriction base="xsd:integer">
```

```
    <xsd:minInclusive value="0"/>
```

```
    <xsd:maxInclusive value="100"/>
```

```
  </xsd:restriction>
```

```
</xsd:simpleType>
```

Ograničenje skupa vrijednosti

```
<xs:enumeration value="value1"/>
...
<xsd:simpleType name="grade">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="A"/>
    <xsd:enumeration value="B"/>
    <xsd:enumeration value="C"/>
    <xsd:enumeration value="D"/>
    <xsd:enumeration value="E"/>
    <xsd:enumeration value="F"/>
  </xsd:restriction>
</xsd:simpleType>
```

Tipovi podataka:

- string
- date
- time
- integer
- decimal
- anyURI
- hexBinary

Ograničenje obrascem

```
<xsd:pattern value="pattern" />
```

pattern: regularni izraz

Tipovi podataka:

- integer
- decimal
- date
- time
- string
- anyURI
- hexBinary
- boolean

```
<xsd:simpleType name="pin">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[0-9][0-9][0-9][0-9]" />
  </xsd:restriction>
</xsd:simpleType>
```

Ograničenja duljine

```
<xsd:minLength value="value"/>
<xsd:maxLength value="value"/>
<xsd:length value="value"/>
```

Tipovi podataka:

- string
- anyURI
- hexBinary

```
<xsd:simpleType name="password">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="8"/>
    <xsd:maxLength value="20"/>
  </xsd:restriction>
</xsd:simpleType>
```

Ograničenje za *whitespace*

`<xsd:whiteSpace value="ws-value"/>`

ws-value

preserve – ne mijenja ws znakove

replace – sve ws znakove svodi na ' '

collapse – replace + briše vodeće i završne ws znakove, više uzastopnih ws znakova pretvara u jedan
(' white space ' -> 'white space')

Tipovi podataka:

- integer
- decimal
- date
- time
- string
- anyURI
- hexBinary
- boolean

```
<xsd:simpleType
  name="wsIgnore_integer">
  <xsd:restriction base="xsd:integer">
    <xsd:whiteSpace value="collapse"/>
  </xsd:restriction>
</xsd:simpleType>
```

Ograničenja za realne brojeve



```
<xsd:totalDigits value="value"/>  
<xsd:fractionDigits value="value"/>
```

Tipovi podataka:

- decimal

```
<xsd:simpleType name="price">  
  <xsd:restriction base="xsd:decimal">  
    <xsd:totalDigits value="7"/>  
    <xsd:fractionDigits value="2"/>  
  </xsd:restriction>  
</xsd:simpleType>
```

Liste

- Atomični tipovi podataka: nedjeljivi prema XML parseru
- Primjeri ugrađenih atomičnih tipova podataka:
 - number, NMTOKEN, ID, date, boolean, (string*), ...
 - *- string može sadržavati prazne znakove (“Marko Ferković”)
- Liste – sadrže dva ili više atomičnih podataka odvojenih praznim znakovima
 - “A B C D E F”
- Ugrađeni tipovi liste: NMTOKENS, IDREFS, ENTITIES...

Definiranje liste

```
<xsd:simpleType name="ime-liste">
  <xsd:list itemType="tip-atomičnog-elementa-liste"/>
</xsd:simpleType>
```

- Iz tipova listi moguće stvoriti nove tipove korištenjem ograničenja:

```
<xsd:simpleType name="ime-nove-liste">
  <xsd:restriction base="ime-bazne-liste">
    <xsd:length value="..." />    <- ograničenje...
  </xsd:restriction>
</xsd:simpleType>
```

- Ograničenja na liste:
 - minLength, maxLength, length, pattern, enumeration

Primjer:

- Tip liste atomičnih podataka tipa pct


```
<xsd:simpleType name="pct-list">
  <xsd:list itemType="pct"/>
</xsd:simpleType>
```
- Tip liste atomičnih podataka tipa pct, 2..5 elemenata u listi

```
<xsd:simpleType name="pct2-5-list">
  <xsd:restriction base="pct-list">
    <xsd:minLength value="2"/>
    <xsd:maxLength value="5"/>
  </xsd:restriction>
</xsd:simpleType>
```

XSD Elementi

- Deklariranje elementa u XSD dokumentu:
`<xsd:element name="element-name" type="type-name"/>`
- U XSD elementi i tipovi definiraju strukturu XML dokumenta
- DTD:
`<!ELEMENT name (content)>`
- XSD dijeli elemente u dvije vrste, u ovisnosti o njihovom tipu
 - **jednostavni** elementi su jednostavnog tipa
 - **složeni** elementi su složenog tipa

Primjer deklaracije i korištenja

- Korištenje ugrađenih tipova podataka

```
<xsd:element name="ime" type="xsd:string"/>
<ime>Marko Ferković - Enter</ime>
<xsd:element name="studenata-u-grupi"
  type="xsd:integer"/>
<studenata-u-grupi>15</studenata-u-grupi>
<xsd:element name="datum-upisa" type="xsd:date"/>
<datum-upisa>2006-07-22</datum-upisa>
```

- Korištenje naknadno definiranih jednostavnih tipova podataka

```
<xsd:element name="ocjena" type="grade"/>
<ocjena>B</ocjena>
<xsd:element name="uspješnost" type="pct"/>
<uspješnost>76</uspješnost>
```

Default i fixed svojstva elementa



- Podrazumijevana vrijednost
 - parser nadopunjuje vrijednost samo ako je element naveden unutar XML dokumenta ali je sadržaj ostavljen prazan
 - ako element nije naveden, parser ga neće dodati u dokument

<xsd:element ... default="default-value"/>

- Nepromjenljiva vrijednost
 - ako je element naveden, vrijednost mora biti “fixed-value”
 - ako je sadržaj prazan, parser će dopuniti vrijednost na “fixed-value”
 - ako element nije naveden, parser ga neće dodati u dokument

<xsd:element ... fixed="fixed-value"/>

XSD Atributi

- Deklariranje atributa u XSD dokumentu:

```
<xsd:attribute name="attr-name" type="type-name"/>
```

- Atributi su isključivo jednostavnog tipa

- vrlo su slični jednostavnim elementima, sastavni su dio složenih tipova

- DTD:

```
<!ATTLIST element attr-name type default-value>
```

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

```
<xsd:attribute name="version" type="xsd:decimal"/>
```

```
<xsd:attribute name="kamata" type="pct"/>
```

Modifikatori atributa

- Podrazumijevana vrijednost (DTD: “default-value”):
`<xsd:attribute ... default="default-value"/>`
- Nepromjenljiva vrijednost (DTD: #FIXED “fixed-value”):
`<xsd:attribute ... fixed="fixed-value"/>`
- Navođenje vrijednosti atributa u XML dokumentu (DTD: #REQUIRED | #IMPLIED) definirano atributom `use` (optional|required|prohibited), podrazumijevana vrijednost “optional”:

`<xsd:attribute ... use="use-type"/>`

Anonimni i imenovani tipovi (I)



Anonimni tipovi

```
<xsd:element name="bodova">  
  ■ <xsd:simpleType>  
    <xsd:restriction base="xsd:integer">  
      <xsd:minInclusive value="0"/>  
      <xsd:maxInclusive value="100"/>  
    </xsd:restriction>  
  ■ </xsd:simpleType>  
</xsd:element>
```

Imenovani tipovi

```
<xsd:element name="bodova" type="pct"/>  
  
<xsd:simpleType name="pct">  
  <xsd:restriction base="xsd:integer">  
    <xsd:minInclusive value="0"/>  
    <xsd:maxInclusive value="100"/>  
  </xsd:restriction>  
</xsd:simpleType>
```


Anonimni i imenovani tipovi (II)



- **Anonimni tipovi:**
 - definirani unutar deklaracije elementa ili atributa
 - vrijede samo za element ili atribut u kojem su ugnježdjeni
- **Imenovani tipovi:**
 - definirani na razini XSD dokumenta
 - mogu se koristiti u deklaracijama više elemenata i atributa



XSD

Složeni tipovi podataka

Složeni tipovi podataka

```
<xsd:complexType name="type-name">  
...  
</xsd:complexType>
```

- Vrste složenih tipova podataka:
 - elementi i atributi u sadržaju
 - prazan sadržaj s atributima
 - jednostavan sadržaj s atributima
 - miješani sadržaj

Elementi i atributi u sadržaju

```
<xsd:complexType name="upis">
  <xsd:sequence>
    <xsd:element name="šk-god" type="xsd:string"/>
    <xsd:element name="uspješnost" type="pct"/>
  </xsd:sequence>
  <xsd:attribute name="ocjena" type="grade"
    use="required"/>
</xsd:complexType>
```

Indikator poretka elemenata

Deklaracija elemenata

Deklaracija atributa

```
<element name="upis-ocjene" type="upis"/>
```

```
<upis-ocjene ocjena='A'>
  <šk_god>2007-2008</šk_god>
  <uspješnost>85</uspješnost>
</upis-ocjene>
```

Prazan sadržaj s atributima

```
<xsd:complexType name="registracija">  
  <xsd:attribute name="oznaka" type="xs:string"/>  
</xsd:complexType>
```

```
<element name="reg" type="registracija" />  
<reg oznaka='ZG VOZIM-FER'/>
```

- U definiciji tipa ne navodi se niti jedan element, samo jedan ili više atributa
- DTD: <!ELEMENT eName EMPTY>
 <!ATTLIST eName attr1 ...>

Jednostavan sadržaj s atributima



- Stvara se proširenjem ili ograničenjem jednostavnog sadržaja - dodavanjem atributa jednostavnom tipu

```
<xsd:complexType name="naziv-predmeta">  
  <xsd:simpleContent>  
    <xsd:extension base="xsd:string">  
      <xsd:attribute name="lang" type="xsd:string"/>  
    </xsd:extension>  
  </xsd:simpleContent>  
</xsd:complexType>
```

jednostavan tip koji proširujemo
s jednim ili više atributa

- DTD:
 <!ELEMENT eName (#PCDATA)>
 <!ATTLIST eName attr1 ...>

```
<element name="naziv" type="naziv-predmeta" />
```

```
<naziv lang="hr">Otvoreno računarstvo</naziv>
```

Miješani sadržaj

```
<xsd:complexType name="opis-predmeta" mixed="true">
  <xsd:sequence>
    <xsd:element name="ime" type="xsd:string"/>
    <xsd:element name="opis" type="xsd:string"/>
    <xsd:element name="kompetencije" type="compList"/>
  </xsd:sequence>
  <xsd:attribute name="lang" type="xsd:string"/>
</xsd:complexType>
```

- DTD:


```
<!ELEMENT predmet-desc (#PCDATA|ime|opis|ciljevi)*>
```
- Za razliku od DTD-a, parser provjerava redoslijed i brojnost elemenata unutar miješanog sadržaja

Indikatori

- **Indikatori brojnosti**

- određuju broj ponavljanja elementa ili grupe elemenata u XML dokumentu koji shema definira

- **Indikatori poretka**

- određuju strukturu dokumenta propisujući poredak elemenata u XML dokumentu

- **Indikatori grupe**

- definiraju grupe elemenata ili atributa

- Daju iste mogućnosti kombiniranja elemenata kao i DTD

Indikatori brojnosti

- *maxOccurs* i *minOccurs* neobavezni atributi elemenata, indikatora poretka i indikatora grupa
- *maxOccurs* – najveći dopušten broj pojavljivanja elementa...
 - vrijednosti: 1.. (cjelobrojna vrijednost) + "unbounded"
- *minOccurs* – najmanji dopušten broj pojavljivanja elementa...
 - vrijednosti: 0.. (cjelobrojna vrijednost)
- Podrazumijevane vrijednosti oba atributa su 1, tj. ako nisu navedeni element se mora pojaviti točno jednom

Primjeri korištenja indikatora brojnosti



```
<xsd:complexType name="compList">
  <xsd:element name="kompetencija" minOccurs="1"
maxOccurs="10">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:maxLength="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:complexType>
```

Najmanje jedna, a najviše deset kompetencija po predmetu

Indikatori poretka

```
<xsd:indikator-poretka>
```

```
  <element ... />
```

```
  <element ... />
```

```
</xsd:indikator-poretka>
```

} elementi na koje se indikator odnosi

- **Sequence:**

- elementi na koje se indikator odnosi moraju se navesti u definiranom redosljedu (DTD: ,)

- **All:**

- elementi na koje se indikator odnosi moraju se navesti svi, svaki element samo jednom, redoslijed navođenja nije bitan (DTD: nema ekvivalenta)

- **Choice:**

- smije se navesti samo jedan od elemenata na koje se indikator odnosi (DTD: |)

Indikator poretka *sequence*

```
<xsd:complexType name="upis">
  <xsd:sequence>
    <element name="šk-god" type="xsd:string"/>
    <element name="uspješnost" type="pct"/>
  </xsd:sequence>
  <xsd:attribute name="ocjena" type="grade"
    use="required"/>
</xsd:complexType>
```

```
<upis-ocjene ocjena='A'>
  <šk_god>2007-2008</šk_god>
  <uspješnost>85</uspješnost>
</upis-ocjene>
```

Indikator poretka *all*

```
<xsd:complexType name="upis">  
  <xsd:all>  
    <element name="šk-god" type="xsd:string"/>  
    <element name="uspješnost" type="pct"/>  
  </xsd:all>  
  <xsd:attribute name="ocjena" type="grade"/>  
</xsd:complexType>
```

```
<upis-ocjene>  
  <uspješnost>85</uspješnost>  
  <šk_god>2007-2008</šk_god>  
</upis-ocjene>
```

Indikator poretka *choice*



```
<xsd:complexType name="jendvatri">
  <xsd:choice>
    <element name="prvi" type="xsd:string"/>
    <element name="drugi" type="xsd:integer"/>
    <element name="treći" type="xsd:date"/>
  </xsd:choice>
</xsd:complexType>

<odlučiSe>
  <treći>2007-02-29</treći>
</odlučiSe>
```

Indikator grupe elemenata

- Definira imenovanu ili anonimnu grupu elemenata
- DTD: operator ()
- Elementi članovi grupe obavezno se navode unutar indikatora poretka (sequence, choice, all)

```

<xs:group name="grupa">
  <xs:sequence>
    <xs:element name="prvi" type="xs:string"/>
    <xs:element name="drugi" type="xs:string"/>
    <xs:element name="treći" type="xs:string"/>
  </xs:sequence>
</xs:group>

```

Referenciranje grupe u XSD-u (I)



- Na mjestu korištenja grupe:
 - imenovana grupa se referencira
 - anonimna grupa se definira

```
<xs:element name="grupni">
  <xs:complexType>
    <xs:group ref="grupa"/>
  </xs:complexType>
</xs:element>
```

```
<xs:element name="grupni">
  <xs:complexType>
    <xs:group>
      <xs:sequence>
        ...
      </xs:sequence>
    </xs:group>
  </xs:complexType>
</xs:element>
```


Korišćenje grupa u definiciji tipova



```
<xs:complexType name="izbor">
  <xs:choice>
    <xs:group ref="grupa" />
    <xs:element name="samoČetvrti" type="xs:string" />
  </xs:choice>
</xs:complexType>
```

```
DTD: <!ELEMENT neki-izbor ((prvi, drugi, treci) |
  (samoČetvrti))>
```

```
<izbornik>
  <prvi> neki tekst </prvi>
  <drugi> neki tekst </drugi>
  <treći> neki tekst </treći>
</izbornik>
```

ili

```
<izbornik>
  <samoČetvrti>
    neki tekst
  </samoČetvrti>
</izbornik>
```

Grupe atributa

- Definira skup zajednički korištenih atributa
- Primjer: grupa “os-version” se sastoji od atributa “os” i “version”, tip atributa “os” definiran lokalno

```

<xs:attributeGroup name="os-version">
  <xs:attribute name="os">
    <xs:simpleType>
      <xs:restriction base="xs:string"/>
        <xs:minLength value="3"/>
        <xs:maxLength value="20"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="version" type="xs:decimal"/>
</xs:attributeGroup>

```

Korištenje grupe atributa

```
<xs:complexType name="bugReport">
  <xs:sequence>
    <xs:element name="date" type="xs:date"/>
    <xs:element name="description" type="xs:string"/>
  </xs:sequence>
  <xs:attributeGroup ref="os-version"
    use="required"/>
  <xs:attribute name="resolved" type="xs:boolean"
    default="false"/>
</xs:complexType>
```

Deklaracija i referenciranje atributa i grupa atributa uvijek nakon deklaracija elemenata

Globalne i lokalne komponente sheme

- Globalne definicije i deklaracije vrijede na razini čitave sheme
- Lokalne definicije i deklaracije vrijede unutar komponente sheme unutar koje su ugniježdene
- Globalne komponente su djeca korijenskog elementa shema:
 - imenovani tipovi, grupe... moraju biti definirani kao djeca korijenskog elementa schema
 - deklaracije elemenata i atributa kao djece korijenskog elementa se također smatraju globalnima
- Globalno deklarirani elementi i atributi mogu se referencirati unutar sheme

Primjer globalnih elemenata

```

<xs:schema xmlns:xs=...>
  <xs:element name="comment" type="xs:string"/>
  <xs:element name="bTree">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="comment" minOccurs="0"/>
        <xs:element name="topNode" type="nodeType"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="nodeType">
    <xs:sequence>
      <xs:element name="lNode" type="nodeType"
        minOccurs="0"/>
      <xs:element name="rNode" type="nodeType"
        minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:integer"
      use="required"/>
  </xs:complexType>
</xs:schema>

```

Globalno deklarirani elementi

Referencirani globalni element

Lokalno deklarirani elementi

Imenovani (globalni) tip

Korijenski elementi

- Samo globalno deklarirani elementi sheme mogu se koristiti kao korijenski elementi valjanog XML dokumenta
- Iz prethodnog primjera: primjeri (dva od beskonačno mogućih) valjanih XML dokumenata:

```
<?xml version="1.0" ?>
<comment> DTD rulez
</comment>
```

```
<?xml version="1.0" ?>
<bTree>
  <topNode id="0">
    <lNode id="1">
      <lNode id="2"/>
      <rNode id="3"/>
      <rNode id="4">
        <lNode id="5"/>
      </rNode>
    </topNode>
  </bTree>
```

Proširenje složenih tipova

- Proširivanje postojećih tipova novim elementima i atributima
- Novi tip posjeduje elemente i attribute baznog tipa, kao i novododane elemente i attribute
- Ekvivalentno nasljeđivanju u objektno-orijentiranim programskim jezicima – klasa dijete dodaje svoja svojstva i metode

```

<xs:complexType name="nodeTypeInt">
  <xs:complexContent>
    <xs:extension base="nodeType">
      <xs:element name="content" type="xs:integer"/>
      <xs:attribute name="isNull" type="xs:boolean"
        default="false"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
  
```

ime novog tipa
 proširuje se složeni sadržaj
 ime tipa koji se proširuje
 element kojim je tip proširen
 atribut kojim je tip proširen

Korištenje proširenih tipova

Schema instance prostor imena

```
<?xml version="1.0" ?>
<bTree xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <topNode id="0">
    <lNode id="1">
      <lNode id="2"/>
      <rNode id="3"/>
      <rNode id="4"
        xsi:type="nodeTypeInt">
        <lNode id="5"/>
        <content>666</content>
      </rNode>
    </topNode>
  </bTree>
```

- parser očekuje sadržaj tipa nodeType (lNode i rNode elemente djecu)
- u sadržaju prisutan element "viška" (content)
- mora se eksplicitno navesti tip sadržaja kako bi parser znao naći njegovu definiciju u shemi i provjeriti sadržaj
- **xsi:type** atribut sadrži ime korištenog tipa koji je proširenje (ili restrikcija) očekivanog tipa sadržaja

Zašto proširivanje tipova?

- Primjer binarnog stabla:
 - definiran osnovni tip čvora koji ne sadrži vrijednost, samo (opcionalno) lijevi i desni čvor dijete
 - želimo koristiti više vrsta čvorova koji mogu sadržavati različite tipove vrijednosti (tekst, cjelobr. vrijednosti, ...)
 - u definiciji sheme moramo definirati sve moguće tipove čvorova kao složene tipove podataka (za one tipove za koje znamo u trenutku definicije sheme!)
 - svaki mogući tip čvora mora moći sadržavati lijevi i desni čvor bilo kojeg mogućeg tipa
 - za n tipova čvorova to je n^3 kombinacija koje moraju biti definirane u shemi!
 - a što ako naknadno moramo dodati još tipova čvorova?

Evo zašto:

- U shemi definiramo osnovni tip čvora (*nodeType*), koji kao djecu prima čvorove tipa *nodeType*
- Čvorove koji sadrže vrijednosti definiramo kao proširenja tipa *nodeType*
 - oni jesu tipa *nodeType*, te se u XML dokumentu mogu navesti kao čvorovi djeca (uz korištenje atributa `xsi:type`)
- Ukupno potreban broj definicija $n+1$:
 - 1-osnovni tip *nodeType*
 - n -čvorovi koji sadržavaju vrijednost
- Olakšano dodavanje novog tipa čvora
 - potrebno je samo definirati novi tip čvora koji proširuje bazni čvor

Apstraktni tipovi



- Izvedeni tipovi:
 - nastali proširenjem ili restrikcijom složenog baznog tipa
- Tip *nodeType* služi samo kao bazni tip
 - za ostale tipove čvorova nije koristan
- Korištenje određenog tipa podatka može se spriječiti njegovim proglašavanjem apstraktnim
 - na mjestu elementa apstraktnog tipa mora se koristiti jedan od izvedenih tipova, uz navođenje imena korištenog izvedenog tipa u atributu `xsi:type`

```
<xs:complexType name="nodeType" abstract="true">
  <x:sequence>
    ...
  <x:sequence>
    <x:attribute name="id" type="xs:integer"
      use="required"/>
</xs:complexType>
```

proglašavanje složenog
tipa podatka *apstraktnim*

XSD i prostori imena



XML deklaracija

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

XSD prostor imena

`<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"`

`targetNamespace="targetNamespaceURI"`

`xmlns="targetNamespaceURI"`

prostor imena komponenti sheme

`xmlns...`

korišteni prostor imena (jedan ili više)

`>`

`...`

`</xsd:schema>`

Prostori imena (I)

- Odvajanje imena XML Schema language komponenata i komponenata u opisivanoj shemi
- XML Schema language elementi, atributi, ugrađeni tipovi u prostoru imena
 - <http://www.w3.org/2001/XMLSchema>
 - dobra praksa korištenje lokalnog prefiksa **xsd** ili **xs**
- Elementi, atributi, tipovi definirani unutar XSD dokumenta:
 - smještaju se u prostor imena naveden u atributu `targetNamespace`
 - ako `targetNamespace` nije naveden, komponente se smještaju u neimenovani prostor imena

Prostor imena (II)

- Nužno definirati lokalni prefiks za korištenje novo definiranih komponenti “pohranjenih” u `targetNamespace` prostor imena

novi tip će biti pohranjen u prostor imena
određen u *targetNamespace* pod imenom “pct”

default prostor imena

lokalni prefiks “ex” za prostor imena

```
<xsd:schema ...
targetNamespace="http://www.fer.hr
/eIndex"
xmlns="http://www.fer.hr/eIndex"
...
<xsd:element name="postotno"
type="pct"/>
<xsd:simpleType name="pct">
...
```

```
<xsd:schema ...
targetNamespace="http://www.fer.hr/
eIndex"
xmlns:ex="http://www.fer.hr/eIndex"
...
<xsd:element name="postotno"
type="ex:pct"/>
<xsd:simpleType name="pct">
...
```

XSD i prostori imena

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

```
<xsd:schema ...
```

```
...
```

```
elementFormDefault="value"
```

```
attributeFormDefault="value">
```

```
...
```

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

qualified – svi elementi u XML dokumentu koji koristi ovu shemu moraju imati naveden lokalni prefiks prostora imena (određen u definiciji sheme s atributom *targetNamespace*)

unqualified – korištenje prefiksa nužno samo za globalno definirane elemente

podrazumijevana vrijednost: *unqualified*

qualified – svi atributi u XML dokumentu koji koristi ovu shemu moraju imati naveden lokalni prefiks prostora imena (određen u definiciji sheme s atributom *targetNamespace*)

unqualified – korištenje prefiksa nužno samo za globalno definirane attribute

podrazumijevana vrijednost: *unqualified*

Definicija sheme u više dokumenata



- Definicija sheme može biti podijeljena u više datoteka, pojedini dokumenti se uključuju u druge dokumente
- *targetNamespace* uključujućeg i uključenog dokumenta mora biti jednak
- Funkcionalnost slična direktivi `#include` u jeziku C/C++

```
<schema targetNamespace=...  
...  
  <include schemaLocation="includedSchemaURI" />  
...  
</schema>
```


Komponente definirane u drugim shemama



- Kombiniranje komponenata iz različitih shema
 - validacija sadržaja dokumenata preko više prostora imena
- Iz drugog prostora imena (tj. scheme) mogu biti uvezene samo globalno definirane komponente
 - elementi, atributi, tipovi...
- Import direktive moraju biti navedene neposredno
 - na početku sadržaja elementa schema
- Sa svakim prostorom imena koji se uvozi mora se asocirati lokalni prefiks
 - koristi se kod navođenja komponenata iz tog prostora imena
- *schemaLocation* atribut može pomoći parseru u pronalaženju datoteke s definicijom uvažane sheme

Biblioteke tipova

- Uvažanje komponenata drugih prostora imena omogućava stvaranje biblioteke osnovnih, često korištenih tipova podataka
- Ti tipovi podataka se uvoze u specifične definicije shemi, te se koriste:
 - izravno, za deklaraciju tipova elemenata u drugim shemama
 - kao baza za izvođenje novih tipova u drugim shemama

Što nije spomenuto?

- ... većina ugrađenih tipova podataka, anotacije, “nil” vrijednosti, restrikcije složenih tipova, redefiniranje tipova i grupa, substitucijske grupe, apstraktni elementi, upravljanje s korištenjem izvedenih tipova, jedinstvenost podataka, ključevi i reference na ključeve, Any element, Any atribut ...