

Uvod u veb i internet tehnologije





Jezici za obeležavanje SGML





Karakteristike i istorijat SGML

- Standardni opšti jezik za obeležavanje (Standard Generalized Markup Language) je meta jezik za obeležavanje standardizovan od strane medunarodne organizacije za standarde (pod oznakom „ISO 8879:1986 SGML”)
- Jezik je razvijen za potrebe kreiranja mašinski čitljivih dokumenata u velikim projektima industrije, državne uprave, vojske itd.
- Osnovna motivacije prilikom standardizovanja ovog jezika je bila da se obezbedi trajnost dokumentima i njihova nezavisnost od aplikacija kojima su kreirani
- Informacije skladištene u okviru SGML dokumenta su nezavisne od platforme tj. od softvera i hardvera
- Pretečom jezika SGML smatra se jezik GML (Generalized Markup Language) nastao u kompaniji IBM 1960-tih



Karakteristike i istorijat SGML (2)

- Jedna od značajnijih primena jezika SGML je bila izrada drugog, elektronskog, izdanja Oksfordskog rečnika engleskog jezika (OED)

```
Document: Bungler OED           At: "<entry>"

<entry>
  <hwsec>
    <hwgp>
      <hwlem>bungler</hwlem>
      <pron>b<I>ʊ</I>'ŋglə</pron>. </hwgp>
      <vfl>Also <vd>b</vd> <vf>bongler</vf>,
      </vfl>
      <etym>f. as prec. + <xra><xlem>-ER</xlem>

<sen>One who bungles; a clumsy unskilful
  <quot>
    <qdat>1533 </qdat>
    <auth>MORE </auth>
    <wk>Answ. Poyson. Bk. </wk>Wks. (1557
    <qtxt>He is even but a very bungler.
```

Fragment oksfordskog
rečnika obeležen SMGL
elementima



Karakteristike i istorijat SGML (3)

- Može se reći da je najznačajnija primena jezika SGML došla kroz jezik HTML, čije su prve verzije definisane upravo u okviru jezika SGML
- Jezik HTML služi za obeležavanje hipertekstualnih dokumenata i postao je standardni jezik za obeležavanje dokumenata na vebu
- Svaki jezik za obeležavanje koji je definisan u SGML-u naziva se i SGML aplikacija, pa se i jezik HTML smatra SGML aplikacijom
- SGML se koristi da bi se obeležila struktura dokumenata određenog tipa



Ilustracije korišćenja SGML

- Primer: Zbirka pesama sadrži nekoliko pesama, pri čemu se svaka pesma sastoji od nekoliko strofa, a svaka strofa od nekoliko stihova
 - SGML uvodi oznake kojima se obeležavaju elementi dokumenta

```
<!DOCTYPE zbirka SYSTEM "zbirka-pesama.dtd">
<zbirka>
  <pesma autor="Čika Jova Zmaj">
    <strofa>
      <naslov>Žaba čita novine</naslov>
      <stih>Sedi žaba sama
      <stih>na listu lokvanja.
      <stih>Od žarkoga sunca
      <stih>štitom se zaklanja.
    </strofa>
  </pesma>
</zbirka>
```



Ilustracije korišćenja SGML (2)

- Primer: Jedan jednostavni HTML dokument

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Moj prvi HTML dokument</title>
  </head>
  <body>
    <p>Zdravo svete! 
    <p>Copyright (&copy;) Milena
  </body>
</html>
```

- U oba prethodna primera, sadržaj dokumenta je obeležen oznakama koje određuju njegovu strukturu



Struktura SGML

- Dokumenti se sastoje od međusobno ugnježenih **elemenata**
 - Za obeležavanje elemenata se koriste **etikete** (tagovi) oblika `<ime-elementa>` i `</ime-elementa>` (na primer `<strofa>` i `</strofa>` ili `<body>` i `</body>`)
 - Elementi sadrže tekst, druge elemente ili kombinaciju i jednog i drugog
- Elementi mogu biti dodatno okarakterisani **atributima**
 - Atributi su oblika `ime-atributa="vrednostatributa"` (na primer `naslov="Žaba čita novine"`)
- U okviru teksta mogu se pojaviti i znakovni entiteti
 - Oni su oblika `&ime-entiteta;` (na primer `©`) koji označavaju određene znakove



Struktura SGML (2)

- Sadržaj i značenje elemenata nije propisano meta jezikom, već svaki jezik definisan u okviru SGML-a definiše sopstveni skup etiketa koje koristi za obeležavanje i definiše njihovo značenje i moguće međusobne odnose
- Svakom dokumentu, pridružen je njegov tip
- Tip dokumenta određuje sintaksu dokumenta tj. određuje koji elementi, atributi i entiteti se mogu javiti u okviru dokumenta i kakav je njihov međusobni odnos
- Posebni programi koji se nazivaju SGML parseri ili SGML validatori mogu da ispituju da li je dokument u skladu sa svojim tipom tj. da li zadovoljava sva sintaksna pravila propisana odgovarajućim tipom



Struktura SGML (3)

- Pripadnost određenom tipu dokumenta, izražava se deklaracijom `<!DOCTYPE>` koja se navodi na početku samog dokumenta
 - U okviru ove deklaracije se nalaze informacije o imenu tipa dokumenta, organizaciji koja ga je kreirala i sl.
 - Obično se u okviru ove deklaracije nalazi uputnica na definiciju tipa dokumenta (Document type definition - DTD)
 - Ove datoteke definišu elemente od kojih se grade konkretni dokumenti
 - U prvom primeru tip dokumenta je definisan datotekom `zbirka-pesama.dtd`
 - U drugom primeru tip dokumenta je definisan datotekom <http://www.w3.org/TR/html4/strict.dtd>
 - Oznaka `PUBLIC` u drugom primeru ukazuje na to da je tip dokumenta javan i dostupan



Struktura SGML (4)

- Primer:

- Tip dokumenta zbirke pesama uvodi elemente **zbirka**, **pesma**, **strofa** i **stih** i zahteva da se zbirka sastoji od nekoliko pesama, da se svaka pesma sastoji od nekoliko strofa, a da se svaka strofa sastoji od nekoliko stihova
- U okviru ove definicije tipa dokumenta, specificirano je da pesma ima atribut **autor** kao i šta sve može biti vrednost ovog atributa

```
<!ELEMENT zbirka - - (pesma+)>  
<!ELEMENT pesma - - (naslov?, strofa+)>  
<!ATTLIST pesma autor CDATA #REQUIRED>  
<!ELEMENT naslov - 0 (#PCDATA)>  
<!ELEMENT strofa - 0 (stih+)>  
<!ELEMENT stih 0 0 (#PCDATA)>
```

- Dakle, korišćenje SGML-a podrazumeva kreiranje sopstvenih ili korišćenje javnih tipova dokumenata i obeležavanje dokumenata u skladu sa njihovim željenim tipom



Struktura SGML (5)

- Proces kreiranja novih tipova dokumenata podrazumeva izradu
 - SGML deklaracije - formalnog opisa leksike samih dokumenata koja prevashodno određuje koji znaci se koriste prilikom kreiranja dokumenata
 - Definicije tipa dokumenta - formalnog opisa sintakse samih dokumenata koja određuje od kojih elemenata, etiketa, atributa i entiteta se dokument sastoji i kakav je njihov međusobni odnos
 - Semantičke specifikacije - neformalnog opisa semantike elemenata, etiketa i atributa koji se koriste u okviru dokumenata
 - Ovakva specifikacija može u sebi da sadrži i neka dodatna ograničenja koja se ne mogu izraziti u okviru formalne definicije tipa dokumenta



Osnovni konstrukti SGML

- Elementi i etikete
 - Osnovna gradivna jedinica SGML dokumenata su **elementi**
 - Elementi su obično označeni **etiketama** (tag). Razlikuju se otvarajuće etikete koje označavaju početak elementa i koje su oblika **<ime-elementa>** i zatvarajuće etikete koje označavaju kraj elementa i koje su oblika **</ime-elementa>**
 - Treba istaći da elementi nisu isto što i etikete
 - Element sačinjava početna etiketa, završna etiketa i sav sadržaj (tekst i drugi elementi) koji se nalaze između njih
 - Ime elementa se navodi i početnoj etiketi i u završnoj etiketi
 - Imena elemenata dozvoljeno je pisati i malim i velikim slovima i ne pravi se razlika između velikih i malih slova



Osnovni konstrukti SGML (2)

● Elementi i etikete

- Primer: element **ul** jezika (tipa dokumenta) HTML, služi da označi neku listu nabrojanih stavki, i njegov sadržaj čine tri elementa **li**, čiji su sadržaji niske Lista 1, Lista 2 i Lista 3

```
<ul>
<LI>Lista 1</LI>
<li>Lista 2</li>
<li>Lista 3</li>
</ul>
```

- Kod nekih SGML elemenata moguće je izostaviti završne etikete, dok je kod nekih čak moguće izostaviti i početne etikete
- Primer: u jeziku HTML, elementi **p** služe da označe pasuse. Pasusi ne zahtevaju navođenje završne etikete **</p>**. Početak novog pasusa **<p>** implicitno označava kraj prethodnog, slično kao i oznaka kraja obuhvatajućeg elementa **</body>**

```
<body>
  <p>Zdravo svete!
  <p>Copyright (&copy;) Milena
</body>
```



Osnovni konstrukti SGML (3)

- Elementi i etikete
 - Neki SGML elementi nemaju svoj sadržaj
 - Primer: HTML element koji označava prelazak u novi red **br**
 - Kod praznih elemenata najčešće je zabranjeno navoditi završnu etiketu



Osnovni konstrukti SGML (4)

● Atributi

- Atributi sadrže dodatne informacije o SGML elementima
- Atributi imaju svoj naziv i vrednost
 - Naziv atributa je razdvojen od vrednosti znakom jednakosti
 - Vrednost atributa treba biti navedena u okviru navodnika (") ili apostofa (')
 - U okviru navodnika moguće je korišćenje apostofa i obratno
 - Ponekad navodnici i/ili apostofi, kod vrednosti atributa, mogu biti izostavljeni
 - Atributi elementa se navode u okviru njegove početne etikete
- Primer: atribut `href` elementa `a` jezika HTML određuje odredište hiperveze

```
<a href="http://www.google.com">Link na google</a>
```



Osnovni konstrukti SGML (5)

● Atributi

- Atributi sadrže dodatne informacije o SGML elementima
- Atributi imaju svoj naziv i vrednost
 - Naziv atributa je razdvojen od vrednosti znakom jednakosti
 - Vrednost atributa treba biti navedena u okviru navodnika (") ili apostofa (')
 - U okviru navodnika moguće je korišćenje apostofa i obratno
 - Ponekad navodnici i/ili apostofi, kod vrednosti atributa, mogu biti izostavljeni
 - Atributi elementa se navode u okviru njegove početne etikete
- Primer: atribut `href` elementa `a` jezika HTML određuje odredište hiperveze

```
<a href="http://www.google.com">Link na google</a>
```

- Imena atributa su nezavisna od veličine slova, dok vrednosti nekada zavise, a nekada ne zavise od veličine slova



Osnovni konstrukti SGML (6)

● Entiteti

- SGML daje mogućnost imenovanja delova sadržaja na portabilan način
- Koncept entiteta u SGML uvodi izvesnu vrstu makro zamena
- Zamena entiteta se vrši kada se dokumenti analiziraju odgovarajućim parserom
- Primer: moguće je deklarirati entitet pod imenom `uvit` koji se zamenjuje tekстом `Uvod u Veb i Internet tehnologije`, i zatim se u okviru ovog dokumenta na ime predmeta pozivati korišćenjem reference na entitet
- Postoji nekoliko vrsta entiteta i referenci na entitete:
 1. obični entiteti (regular entities)
 2. parametarski entiteti (parameter entities)
 3. znakovni entiteti (character entities)



Osnovni konstrukti SGML (7)

● Entiteti

1. Obični entiteti

- Reference na obične entitete počinju sa znakom `&` i završavaju se sa `;`
- Moguće ih je navoditi u okviru teksta dokumenta (ne u okviru DTD)
- Primer: ako se negde u okviru dokumenta javi sadržaj

```
Nastava iz predmeta "&uvit;" se odvija utorkom.
```

ovim je u stvari kodiran tekst

```
Nastava iz predmeta "Uvod u Veb i Internet tehnologije"  
se odvija utorkom.
```

2. Parametarski entiteti

- Reference na parametarske entitete počinju znakom `%` i završavaju se sa `;`
- Moguće ih je navoditi samo u okviru DTD dokumenta (ne u okviru objektnih dokumenata)



Osnovni konstrukti SGML (8)

● Entiteti

3. Znakovni entiteti

- Njima se uvode imena koja označavaju određene znakove
- Koriste se da bi se naveli znakovi koji imaju specijalno značenje, zatim neki retko korišćeni znakovi, znakovi koji nisu podržani tekućim kodiranjem ili znakovi koje je nemoguće uneti u okviru softvera za kreiranje dokumenata
- Primer: u jeziku HTML "<" označava znak <, dok """ označava znak "
- Pored referenci na znakovne entitete, za predstavljanje znakova u dokumentima je moguće koristiti i numeričke znakovne reference
- One se navode kao brojevi (dekadni ili heksadekadni) zapisani između &# i ;
- Obično ove vrednosti odgovaraju ISO 10646, tj. UNICODE-u
- Heksadekadni kodovi počinju sa x ili X



Osnovni konstrukti SGML (9)

● Komentari

- U okviru SGML dokumenata moguće je navoditi i komentare, i to na sledeći način:

```
<!-- Ovo je jedan komentar -->  
<!-- Ovo je komentar,  
      koji ne staje u jedan red -->
```

● Označene sekcije

- Označene sekcije (marked sections) se koriste da bi se označili delovi dokumenta koji zahtevaju posebnu vrstu procesiranja
- One su sledećeg oblika:

```
<![ ključna reč [ ... označena sekcija ... ]]>
```

- Najčešće korišćene ključne reči su:

- CDATA - označava doslovan sadržaj koji se ne parsira
- IGNORE - označava da se sekcija ignoriše tokom parsiranja
- INCLUDE - označava da se sekcija uključuje tokom parsiranja
- TEMP - označava da je sekcija privremeni deo dokumenta



Osnovni konstrukti SGML (10)

- Instrukcije procesiranja
 - Instrukcije procesiranja (processing instructions) su lokalne instrukcije aplikaciji koja obrađuje dokument
 - One su napisane na način specifičan za aplikaciju
 - Navode se između `<? i ?>`
 - Primer: u delu HTML dokumenta

```
<p>Sada je <?php echo date("h:i:s"); ?> </p>
```

instrukcija `<?php echo date("h:i:s"); ?>` govori PHP interpretatoru koji obrađuje dokument da je u pitanju deo PHP koda koji je onda potrebno interpretirati



Definicije tipa dokumenta

Svaki element i atribut u okviru neke SGML aplikacije se definiše u okviru definicije tipa dokumenta (DTD)

● Deklaracije entiteta

- Entiteti se deklarišu korišćenjem `<!ENTITY` za kojim sledi ime entiteta, vrednost entiteta pod navodnicima i završni znak `>`

- Primer: Ovim je deklarisan entitet

```
<!ENTITY uvit "Uvod u Veb i Internet tehnologije">
```

- U slučaju parametarskih entiteta, koristi se oznaka `%`

- Primer: Ovim je deklarisan parametarski entitet

```
<!ENTITY % fontstyle "TT | I | B | BIG | SMALL">
```

- Već deklarisani entitet može učestvovati u deklaraciji drugih entiteta

- Primer: Pethodno deklarisan entitet se dalje koristi u okviru DTD za deklaraciju drugih entiteta

```
<!ENTITY % inline  
  "#PCDATA | %fontstyle; | %phrase; | %special; | %formctrl;">
```




Definicije tipa dokumenta (2)

● Deklaracije elemenata

- Većina DTD se sastoji od deklaracija elemenata i njihovih atributa
- Deklaracija elementa počinje sa `<!ELEMENT`, završava se sa `>`, a između se navodi:

1. Ime elementa
2. Pravila minimalizacije, koja određuju da li je neka od etiketa opcionalna
 - Dve crtice - nakon imena označavaju da su obe etikete obavezne
 - Crtica - za kojom sledi O označava da se završna etiketa može izostaviti
 - O- za kojom sledi crtica označava da se početna etiketa može izostaviti
 - Dva slova O označavaju da se obe etikete mogu izostaviti
3. Sadržaj elementa.

Dozvoljeni sadržaj elementa se naziva model sadržaja (content model)

Za definiciju modela sadržaja koriste se:

- prosti modeli sadržaja
- složeni modeli sadržaja



Definicije tipa dokumenta (3)

- Deklaracije elemenata
 - prosti modeli sadržaja
 - EMPTY - elementi koji nemaju sadržaj, tj. prazni elementi
 - ANY - element može imati proizvoljan sadržaj koji se sastoji od teksta i drugih elemenata
 - CDATA (character data) – sadržaj koji se neće analizirati pomoću SGML parsera
Sadržaj se tumači doslovno kako je napisan tj. reference na entitete se ne zamenjuju entitetima, a etikete koje se u njemu nalaze ne označavaju elemente.
 - RCDATA (replacable character data) - slično kao CDATA, osim što se reference zamenjuju (etikete i dalje ne označavaju elemente)
 - složeni modeli sadržaja - koriste se u slučaju kada element može da sadrži druge uneždene elemente
 - Modeli grupe su predstavljeni izrazima u zagradama



Definicije tipa dokumenta (4)

● Deklaracije elemenata

○ Atomi u izrazima modela grupe za složeni modeli sadržaja su:

- imena elemenata - označavaju uneždene elemente
- #PCDATA (parsed character data) - tekst koji će se analizirati pomoću parsera

Reference na entitete se u okviru ovog teksta se zamenjuju entitetima i etikete koje se u njemu nalaze označavaju elemente

○ Ovi atomi se dalje mogu kombinovati sledećim veznicima

- $A?$ - atom A se može, ali ne mora pojaviti
- A^+ - atom A se mora pojaviti jedan ili više puta
- A^* - atom A se mora pojaviti nula ili više puta
- $A | B$ - ili atom A ili atom B se mora pojaviti, ali ne oba
- A, B - oba atoma A i B se moraju pojaviti u tom redosledu
- $A \& B$ - oba atoma A i B se moraju pojaviti u bilo kom redosledu



Definicije tipa dokumenta (4)

● Deklaracije elemenata

- Moguće je definisati dodatna pravila uključivanja i isključivanja sadržaja
 - +(S) - sadržaj S se može pojaviti.
 - -(S) - sadržaj S se ne sme pojaviti
- Definicije elemenata mogu da sadrže reference parametarskih entiteta
- Primer: Delovi DTD za zbirku pesama

```
<!ELEMENT zbirka - - (pesma+)>
```

Element **zbirka** u sebi sadrži jedan ili više elemenata **pesma**, pri čemu se obe etikete moraju navoditi

```
<!ELEMENT pesma - - (naslov?, strofa+)>
```

Element **pesma** može, a ne mora, da sadrži element **naslov** za kojim sledi jedan ili više elemenata **strofa**. Obe etikete se opet moraju navesti



Definicije tipa dokumenta (5)

- Deklaracije elemenata

- Primer: Delovi DTD za zbirku pesama

```
<!ELEMENT stih O O (#PCDATA)>
```

Sadržaj elementa **stih** je proizvoljan tekst koji može da uključi i reference entiteta, ali ne sme da uključi druge elemente

- Primer: Element u HTML-u koji predstavlja hiper-vezu

```
<!ELEMENT A - - (%inline;)* -(A)>
```

Ovde je korišćeno je dodatno pravilo isključivanje sadržaja, pa element **A** sadrži nula ili više elemenata obuhvaćenih parametarskim entitetom **%inline;**, ali ne sme da sadrži drugi element **A**



Definicije tipa dokumenta (6)

● Deklaracije atributa

○ Deklaracija atributa u okviru DTD počinje sa `<!ATTLIST`, nakon koga se navodi element za koji se deklariše atribut, potom sledi lista deklaracija pojedinačnih atributa i na kraju se navodi simbol `>`

○ Svaka deklaracija pojedinačnih atributa je trojka koja definiše:

1. Ime atributa

2. Tip vrednosti atributa, ili eksplicitno naveden skup dopustivih vrednosti
Najčešće korišćeni tipovi su:

- CDATA (character data) - kao i u slučaju elemenata, označava tekst koji se neće analizirati pomoću SGML parsera
- NAME - označava imena
- ID - Označava jedinstvene identifikatore tj. imena koja moraju biti jedinstvena u celom dokumentu
- NUMBER - Označava brojevne vrednosti

3. Naznaku da li je vrednost atributa implicitna, fiksirana ili zahtevana



Definicije tipa dokumenta (7)

● Deklaracije atributa

- Ako je naznačeno da je vrednost atributa implicitna (ključna reč #IMPLIED), to znači da podrazumevanu vrednost određuje softver koji vrši obradu dokumenta
 - Ako je naznačeno da je vrednost atributa fiksirana (ključna reč #FIXED), to podrazumeva da atribut može da ima samo jedinu moguću vrednost koja je u nastavku navedena
 - Ako je naznačeno da je vrednost atributa zahtevana (ključna reč #REQUIRED), tada je na ovom mestu moguće i eksplicitno specificirati podrazumevanu vrednost atributa
- Naravno, definicije atributa mogu da sadrže reference parametarskih entiteta



Definicije tipa dokumenta (8)

- Deklaracije atributa

- Primer: Delovi DTD za zbirku pesama

```
<!ATTLIST pesma
  autor CDATA #REQUIRED
>
```

Ovim je za element **pesma** deklarisan atribut **autor**, čija je vrednost neki tekst, pri čemu je navođenje atributa obavezno

- Primer: Delovi DTD za tabelu u HTML-u

```
<!ATTLIST td
  rowspan NUMBER 1
  colspan NUMBER 1
>
```

Ovim se za element **td** uvode atributi **rowspan** i **colspan** čije su vrednosti brojevi, dok je podrazumevana vrednost za oba atributa 1



Definicije tipa dokumenta (9)

- Deklaracije atributa

- Primer: Delovi DTD za deo HTML-a

```
<!ATTLIST html  
  version CDATA #FIXED "%HTML.Version"  
>
```

Ovim se označava da vrednost atributa `version` elementa `html` može da bude isključivo vrednost određena parametarskim entitetom `HTML.Version` (koji definiše tekuću verziju)



Uključivanje DTD

- DTD može biti naveden ili kroz unutrašnju ili kroz spoljašnju deklaraciju
 - Unutrašnja deklaracija podrazumeva da se DTD deklaracije nalaze u zaglavlju datoteke u kojoj je smešten dokument

Primer: Unutrašnja deklaracija

```
<!-- test.sgml -->
<!DOCTYPE test
  <!ELEMENT test - - PCDATA>
>
<test>Zdravo</test>
```

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend</body>
</note>
```



Uključivanje DTD (2)

○ Primer: Unutrašnja deklaracija

```
<?xml version="1.0"?>  
<!DOCTYPE note [  
  <!ELEMENT note (to,from,heading,body)>  
  <!ELEMENT to (#PCDATA)>  
  <!ELEMENT from (#PCDATA)>  
  <!ELEMENT heading (#PCDATA)>  
  <!ELEMENT body (#PCDATA)>  

```

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

```



Uključivanje DTD (2)

- DTD može biti naveden ili kroz unutrašnju ili kroz spoljašnju deklaraciju
 - Spoljašnja deklaracija podrazumeva da se DTD deklaracije nalaze u spoljašnjoj datoteci, bilo na lokalnom sistemu ili javno na vebu
 - U tom slučaju se u okviru `<!DOCTYPE>` navodi ime datoteke koja sadrži DTD
 - Primer: Spoljašnja deklaracija

```
<!-- test.sgml -->  
<!DOCTYPE test SYSTEM "test.dtd">  
<test>Zdravo</test>
```

```
<!-- test.dtd -->  
<!ELEMENT test - - PCDATA>
```



Uključivanje DTD (3)

Primer: Spoljašnja deklaracija

note.dtd

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

note.xml

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



Zahvalnica

Delovi materijala ove prezentacije su preuzeti iz:

- Skripte iz predmeta Uvod u veb i internet tehnologije, na Matematičkom fakultetu Univeziteta u Beogradu, autor prof. dr Filip Marić
- Skripte iz predmeta Informatika na Univerzitetu Milano Bicocca, autor dr Mirko Cesarini