

XSLT

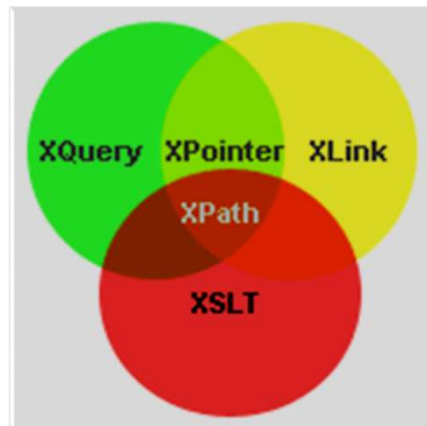
1 18:12

Uvod - 1

- ▶ XSL - **EX**tensible **S**tylesheet **L**anguage.
- ▶ Ima dva dela:
 - ▶ **XSL** Transformations (XSLT)
 - ▶ **XSL** Formatting **O**bjects (XSL-FO)
- ▶ XSLT je XML aplikacija za zadavanje pravila po kojima se jedan dokument transformiše u drugi.
- ▶ XSLT je standardizovan
W3C Recommendation

▶ 2 18:12

Uvod - 2



▶ 3

18:12

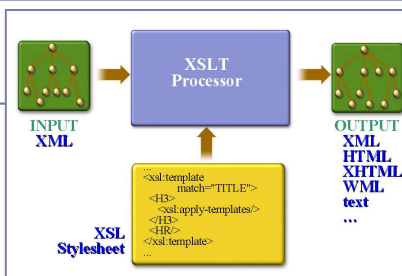
Uvod - 3

- ▶ XSLT dokument (*opis stilova*) jeste skup šablonskih pravila (eng. *template rules*).
- ▶ Šablonsko pravilo ima uzorak i šablon.
- ▶ XSLT procesor poredi elemente tačnije sve čvorove ulaznog XML dokumenta sa obrascima šablona. Kada nađe podudarnost, procesor upisuje odgovarajući uzorak na osnovu odgovarajućeg pravila u izlazno stablo. Pošto to obavi, može se serijalizovati ceo dokument u neki drugi XML, HTML ili čist tekst.

▶ 4

18:12

Primena XSLT



Presentation Oriented Publishing (POP)

- ▶ Korisno za pretraživače i editore.
- ▶ Obično za podatke koje će koristiti čovek

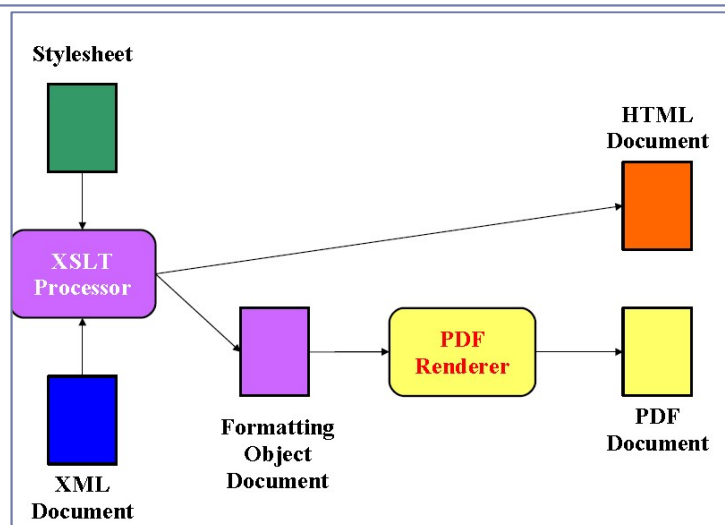
Message Oriented Middleware (MOM)

- ▶ Korisno za razmenu podataka Machine-to-Machine
- ▶ Korisno za komunikaciju tipa Business-to-Business

▶ 5

18:12

Blok šema jednog primera primene XSLT - POP



▶ 6

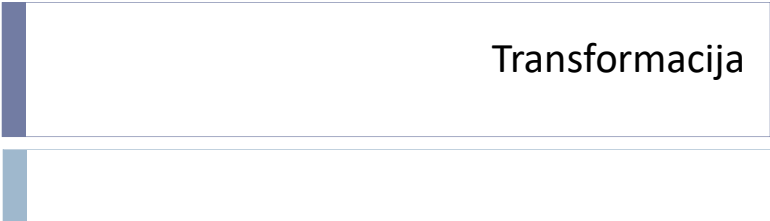
18:12

XSLT i Browsers

- ▶ Svi (najveći) pretraživači imaju podršku za XML i XSLT.
- ▶ **Mozilla Firefox**
- ▶ Firefox podržava *XML*, *XSLT*, i *XPath* od verzije 3.
- ▶ **Internet Explorer**
- ▶ IE podržava XML, XSLT, i XPath od verzije 6.
- ▶ IE5 **NIJE** kompatibilan sa W3C XSL preporukama.
- ▶ **Google Chrome**
- ▶ Chrome podržava XML, XSLT, i XPath od verzije 1.
- ▶ **Opera**
- ▶ Opera podržava XML, XSLT, and XPath od verzije 9. Opera 8 podržava samo XML + CSS.
- ▶ **Apple Safari**
- ▶ Safari podržava XML and XSLT od verzije 3.

▶ 7

18:12



Transformacija

8

18:12

XML dokument za primere XSLT

```
<?xml version="1.0"?>
<people>
  <person born="1912" died="1954">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <profession>computer scientist</profession>
    <profession>mathematician</profession>
    <profession>cryptographer</profession>
  </person>
  <person born="1918" died="1988">
    <name>
      <first_name>Richard</first_name>
      <middle_initial>P</middle_initial>
      <last_name>Feynman</last_name>
    </name>
    <profession>physicist</profession>
    <hobby>Playing the bongoes</hobby>
  </person>
</people>
```

- Zapazite da:
- Primer ne sadrži DTD/XSD.
=>
- XSLT radi podjednako dobro sa validnim i nevalidnim (ali uvek dobro formiranim) dokumentima.
- XSLT vodi računa o prostorima imena.

▶ 9

18:12

Deklaracija XSLT dokumenta

- ▶ Koreni element koji deklarise XSLT dokument je **<xsl:stylesheet>** ili **<xsl:transform>**
odgovarajućeg imenskog prostora
- ▶ <xsl:stylesheet> i <xsl:transform> su sinonimi i bilo koji može biti korišćen!
- ▶ Korektna deklaracija:
- ▶ **<xsl:stylesheet version="1.0"**
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
- ▶ ili:
- ▶ **<xsl:transform version="1.0"**
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
- ▶ Deklarisanje prostora imena je obavezno!
- ▶ Za korišćenje prostora imena **xmlns:xsl="http://www.w3.org/1999/XSL/Transform"** morate takođe uključiti i atribut verzije: **version="1.0"**.

▶ 10

18:12

Minimalan XSLT

- ▶ URI mora biti tačno naveden! Ako se pogreši makar u 1 slovu, procesor opisa stilova će dati sam opis umesto originalnog ili transformisanog. Mora se navesti i verzija.

- ▶ **Minimalan XSLT je:**

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

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
</xsl:stylesheet>
```

koji daje rezultat transformacije:

```
<?xml version="1.0"
encoding="utf-8"?>
```

Alan
Turing

computer scientist
mathematician
cryptographer

Richard
P
Feynman

physicist
Playing the bongoes

▶ 11

18:12

Komentar

- ▶ **Primena praznog opisa stilova na bilo koji XML dokument je reprodukcija sadržaja bez oznaka.**
- ▶ Dakle, iako eksplicitnih pravila nema, XSLT procesor će primeniti određena pravila tzv. **ugrađena pravila**.
- ▶ Da bismo urdili željenu transformaciju, opisu stilova treba dodati šablonska pravila koja će XSLT procesoru saopštiti kako da obradi pojedine elemente ulaznog dokumenta i/ili da obuhvati druge sadržaje XML dokumenta poput atributa.
- ▶ **XSLT procesor** je softver koji učitava određen XML dokument, učitava XSLT opis stilova i generiše izlazni dokument na osnovu instrukcija.
- ▶ Može biti ugrađen u aplikacije. Može biti ugrađen u serverske aplikacije, kao što je Cocoon (<http://xml.apache.org/cocoon>). Može biti i samostalan (<http://saxon.sourceforge.net> ili <http://xml.apache.org/xalan-j>)

▶ 12

18:12

Uzorci i šabloni

13

18:12

Šabloni

- ▶ Pomoću šablonskih pravila se zadaje koji izlazni element se pravi pomoću kojih ulaznih.
- ▶ Svako pravilo se zadaje jednim elementom

xsl:template

▶ 14

18:12

<xsl:template match="XPath">

- ▶ ...koristi se za formiranje šablona transformacije.
- ▶ **match** – atribut se koristi za pridruživanje šablona uzorku XML dokumenta koji je opisan izrazima *XPath* tipa. To može biti neki XML element, više elemenata, ceo XML dokument,...
- ▶ Vrednost atributa *match* je *XPath* izraz kojim se definiše uzorak tj. deo XML dokumenta na koji se primenjuje šablon.
- ▶ Na primer, ako je *match="/"* onda je uzorak ceo dokument.
- ▶ Tiče se samo izlaza koji odgovara vrednosti u match atributu.

▶ 15

18:12

Primer – 3.2

- ▶ Na izlazu treba prikazati string "osoba" za svaki element "person" u XML dokumentu. Sve ostalo treba ignorisati.

```
<?xml version="1.0" ?>
<people>
<person>...</person>
<person>...</person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="person">osoba</xsl:template>

</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
osoba
osoba
```

▶ 16

18:12

Primer – 3.3

- String "osoba" nazivamo literalom. Osim uzorka, šablon može da sadrži i elemente koji su deo formatiranog izlaza, na primer paragrafa u HTML dokumentu.

<pre><?xml version="1.0" ?> <people> <person>...</person> <person>...</person> </people></pre>	<pre><?xml version="1.0"?> <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:template match="person"> <p>osoba</p> </xsl:template> </xsl:stylesheet></pre>
--	---


```
<?xml version="1.0" encoding="utf-8"?>
<p>osoba</p>
<p>osoba</p>
```

► 17

18:12

Primer – 3.4

- XSLT mora biti dobro formiran XML dokument, iz čega sledi da tekst u šablonu ne može biti proizvoljan.

<pre><?xml version="1.0" ?> <people> <person>...</person> <person>...</person> </people></pre>	<pre><?xml version="1.0"?> <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:template match="person"> A Person<p> </xsl:template> </xsl:stylesheet></pre>
--	--

Nema završnog taga.

► 18

18:12

Primer – 3.5

- Šta se događa ako šablon deluje samo na neke elemente, na primer na „name“ ili „hobby“?

```
<?xml version="1.0" ?>
<people>
<person>...</person>
<person>...</person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="name">osoba</xsl:template>

</xsl:stylesheet>
```

osoba
computer scientist
mathematician
cryptographer

osoba
physicist
Playing the bongoes

Zaključak: primenom šablona vrši se zamena elementa drugim sadržajem, dok se za ostale primenjuje podrazumevano pravilo

18:12

Primer – 3.6

- Šta se događa ako šablon deluje na isti elemenat više puta? Da li je redosled tada od značaja?

```
<?xml version="1.0" ?>
<people>
<person>...</person>
<person>...</person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="name">A Person</xsl:template>
<xsl:template match="name">Osoba</xsl:template>

</xsl:stylesheet>
```

Osoba:
computer scientist
mathematician
cryptographer

Osoba:
physicist
Playing the bongoes

Zaključak: Ne uzimaju se šabloni koji rade nad istim elementom više puta, već samo poslednji.

18:12

Formiranje vrednosti

- ▶ Izračunavanje znakovne vrednosti i umetanje takve vrednosti u izlaz se postiže primenom elementa:
- ▶ **xsl:value-of**
- ▶ Koristi se za izdvajanje vrednosti nekog XML elementa i dodavanje te vrednosti u izlazni tok podataka transformacije.
- ▶ Vrednost se izdvaja na osnovu vrednosti atributa **select** odnosno na osnovu skupa čvorova koji daje *template*
- ▶ **<xsl:value-of select="..." />**

▶ 21

18:12

Primer – 3.5

- ▶ U izlazni dokument želimo da ubacimo samo imena osoba.

```
<?xml version="1.0" ?>
<people>..
<person>
  <name>...</name>
</person>
<person>
  <name>...</name>
</person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="person">
    <p> <xsl:value-of select="name" /> </p>
  </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
<p>
  Alan
  Turing
</p>
<p>
  Richard
  P
  Feynman
</p>
```

▶ 22

18:12

► ZAKLJUČAK:

- Sve što se podudara sa definisanim šablonom zameniti sa definisanim vrednosti!!!
- Sve ostalo ostaje isto!!!!

► 23

18:12

Primer 3.6 Formiranje HTML dokumenta

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
</catalog>
```

My CD Collection

Title: **Empire Burlesque**Artist: **Bob Dylan**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
    <body>
      <h2>My CD Collection</h2>
      <p>
        Title:
        <span style="color:#ff0000">
          <xsl:value-of select="catalog/cd/title"/>
        </span>
      </p>
      <p>
        Artist:
        <span style="color:#00ff00">
          <xsl:value-of select="catalog/cd/artist"/>
        </span>
      </p>
    </body>
  </html>
</xsl:template>
pao</xsl:stylesheet>
```

18:12

Komentar za primer 3.6

- ▶ Drugi element, **<xsl:stylesheet>**, definiše da je ovaj dokument XSLT transformacija dokumenta (zajedno sa verzijom i atributima).
- ▶ **<xsl:template>** definiše šablon tj. uzorak.
- ▶ **match="/"** pridružuje šablon korenskom elementu XML dokumenta.
- ▶ Sadržaj unutar **<xsl:template>** elementa definiše neki HTML koji će biti ispisan kao izlaz.
- ▶ Posledenje dve linije definišu kraj uzorka.
- ▶ Napomena: Rezultat ovog primera nije izdvajanje podataka iz XML dokumenta.

▶ 25

18:12

Primer 3.7

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <html>
    <body>
    <h2>My CD Collection</h2>
    <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <tr>
      <td><xsl:value-of select="catalog/cd/title"/></td>
      <td><xsl:value-of select="catalog/cd/artist"/></td>
    </tr>
    </table>
    </body>
    </html>
  </xsl:template>

</xsl:stylesheet>

```

<td><xsl:value-of select="catalog/cd/title"/></td>
 <td><xsl:value-of select="catalog/cd/artist"/></td>

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan

▶ 26

Ako se uz match stavi catalog onda se u select stavlja samo cd/title

18:12

Komentar za primer 3.7

- ▶ Vrednost za atribut **select** je *XPath* izraz. *XPath* izraz radi tako što se pronalazi stavka u fajlu; pri tome (/) selektuje podelemente u jednoj putanji.
- ▶ U ovom primeru smo uspeli da uzmemo podatke iz XML fajla, ali samo jedan red podataka.

▶ 27

18:12

Primena `<xsl:apply-templates select = " ">`

- ▶ XSLT procesor čita dokument [od vrha do dna](#), počev od korenskog elementa. [Tim redosledom se primenjuju šablonska pravila](#). Ako, na primer, prvi šablon odgovara imenu i profesiji, `match = "name|profession"`, a ispod idu dva, jedan koji odgovara imenu, a drugi profesiji: `match="name"` odnosno `match="profession"`, onda će ta druga dva da pokriju delovanje prvog!
- ▶ Šablon može da izmeni redosled prolaska XSLT procesora kroz tekst. Šablon može da definiše koji se sledeći element obrađuje ili da spreči obradu nekog elementa.
- ▶ Element `<xsl:apply-templates>` [eksplicitno navodi redosled](#) obrade.
- ▶ Atribut **select** sadrži *XPath* izraz koji kazuje XSLT procesoru koje čvorove treba da obradi pri formiranju izlaza.

▶ 28

18:12

Primer – 3.8

- Iz ulaznog dokumenta izvući imena i prezimena, tako da prezime bude pre imena. **Preskočiti zanimanje i hobi.**

```
<?xml version="1.0"?>
<people>
  <person born="1912" died="1954">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <profession>computer scientist</profession>
    <profession>mathematician</profession>
    <profession>cryptographer</profession>
  </person>
  <person born="1918" died="1988">
    <name>
      <first_name>Richard</first_name>
      <middle_initial>P</middle_initial>
      <last_name>Feynman</last_name>
    </name>
    <profession>physicist</profession>
    <hobby>Playing the bongoes</hobby>
  </person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="name">
  <xsl:value-of select="last_name"/>,
  <xsl:value-of select="first_name"/>
</xsl:template>
```

```
<xsl:template match="person">
  <xsl:apply-templates select="name"/>
</xsl:template>
```

```
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
Turing,
Alan
Feynman,
Richard
```

▶ 29

18:12

Primer – 3.9

- Iz ulaznog dokumenta naći imena, prezimena i hobi, tako da prezime bude pre imena.

```
<?xml version="1.0"?>
<people>
  <person born="1912" died="1954">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <profession>computer scientist</profession>
    <profession>mathematician</profession>
    <profession>cryptographer</profession>
  </person>
  <person born="1918" died="1988">
    <name>
      <first_name>Richard</first_name>
      <middle_initial>P</middle_initial>
      <last_name>Feynman</last_name>
    </name>
    <profession>physicist</profession>
    <hobby>Playing the bongoes</hobby>
  </person>
</people>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="person">
  <xsl:value-of select="hobby"/>,
</xsl:template>
```

```
<xsl:template match="name">
  <xsl:value-of select="last_name"/>,
  <xsl:value-of select="first_name"/>
</xsl:template>
```

```
<xsl:template match="person">
  <xsl:apply-templates select="name|hobby"/>
</xsl:template>
```

```
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="utf-8"?>
Turing,
Alan
Feynman,
RichardPlaying the bongoes
```

▶ 30

18:12

Važna zapažanja

- ▶ Redosled šablonskih pravila u opisu nema značaja.
- ▶ Redosled elemenata u ulaznom dokumentu je od značaja.
- ▶ Primena šablona je važna i kada potomci nemaju vlastite šablone. Primer je kreiranje html zaglavlja.

```
<xsl:template match="people">
  <html>
    <head><title>Famous Scientists</title></head>
    <body>
      <xsl:apply-templates select="person" />
    </body>
  </html>
</xsl:template>
```

odnosi se na sadržaj tj decu el person

▶ 31

18:12

Primer – 3.10

- ▶ Ako želite da se šabloni primene na sve potomke elementa *people*, a ne samo na *person*, onda izostavite atribut *select*.

<pre><?xml version="1.0" ?> <people> <person born="1912" died="1954"> <name> <first_name>Alan</first_name> <last_name>Turing</last_name> </name> <profession>computer scientist</profession> <profession>mathematician</prof ession> <profession>cryptographer</profe ssion> </person> </people></pre>	<pre><?xml version="1.0"?> <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:template match="people"> <html> <head><title>Famous Scientists</title></head> <body> <xsl:apply-templates/> </body> </html> </xsl:template> <xsl:template match="name"> <p><xsl:value-of select="last_name"/>, <xsl:value-of select="first_name"/></p> </xsl:template> <xsl:template match="person"> <xsl:apply-templates select="name"/> </xsl:template> </xsl:stylesheet></pre>	<pre><html> <head> <META http-equiv="Content- Type" content="text/html; charset=utf-8"> <title>Famous Scientists</title> </head> <body> <p>Turing, Alan</p> <p>Feynman, Richard</p> </body> </html></pre>
--	---	---

▶ 32

18:12

Višestruka upotreba elementa: atribut *mode*

▶ Ako je potrebno da se neki ulazni sadržaj pojavi više puta u izlaznom dokumentu, formatiran na osnovu različitih šablona, koristi se atribut **mode**. Na primer:

- ▶ Naslovi poglavlja
 - ▶ Tabela sadržaja
 - ▶ U samim poglavljima

▶ Atribut: **mode** koristi se u dva elementa:

1. xsl:template
2. xsl:apply-templates

Vrednost dodeljena atributu **mode** u elementu **xsl:apply-templates** označava grupu šablona (opisanih elementima **template**) koji imaju istu vrednost atributa **mode**.

▶ 33

18:12

Primer 3.11

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

```
<people>
```

```
<person born="1912" died="1954">
  <name>
    <first_name>Alan</first_name>
    <last_name>Turing</last_name>
  </name>
  <profession>computer scientist</profession>
  <profession>mathematician</profession>
  <profession>cryptographer</profession>
</person>
```

```
<person born="1918" died="1988">
  <name>
    <first_name>Richard</first_name>
    <middle_initial>P</middle_initial>
    <last_name>Feynman</last_name>
  </name>
  <profession>physicist</profession>
  <hobby>Playing the bongoes</hobby>
</person>
```

```
</people>
```

▶ 34

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

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="people">
```

```
  <html> <head><title>Famous
    Scientists</title></head><body>
```

```
    <ul><xsl:apply-templates select="person"
      mode="toc"/></ul>
```

```
    <xsl:apply-templates select="person"/>
  </body> </html>
```

```
</xsl:template>
```

```
<!-- Table of Contents Mode Templates -->
```

```
<xsl:template match="person" mode="toc">
```

```
  <xsl:apply-templates select="name" mode="toc"/>
```

```
</xsl:template>
```

```
<xsl:template match="name" mode="toc">
```

```
  <li><xsl:value-of select="last_name"/>, <xsl:value-of
    select="first_name"/></li>
```

```
</xsl:template>
```

```
<!-- Normal Mode Templates -->
```

```
<xsl:template match="person">
```

```
  <p><xsl:apply-templates/></p>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```

```
<html>
<head><title>Famous
Scientists</title></head>
<body>
```

```
  <ul>
    <li>Turing, Alan</li>
    <li>Feynman,
      Richard</li>
  </ul>
```

```
  <p>
    Alan
    Turing
```

```
    computer scientist
    mathematician
    cryptographer
```

```
</p>
```

```
  <p>
    Richard
    M
    Feynman
```

```
    physicist
    Playing the bongoes
```

```
</p>
```

```
</body>
```

```
</html>
```

18:12

apply-templates - rezime

- ▶ Ovaj elemenat sprovodi (primenjuje) šablon na tekući element **ili na podelemente tekućeg elementa**, tako što primeni sve šablone.
- ▶ Ako se doda atribut *select* na ovaj element, <xsl:apply-templates>, tada će se vršiti primena samo na podelementima koji odgovaraju vrednosti atributa. Možemo koristiti select atribut i za specifikaciju redosleda u kome će se podelementi obrađivati.
- ▶ Ako ne postoji atribut *mode*, onda to pravilo ne deluje na sve šablone već samo na one koji takođe nemaju *mode*.

▶ 35

18:12

Primer 3.12

- ▶ Transformisati sledeći XML dokument ("cdcatalog.xml") u XHTML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  .
  .
</catalog>
```

My CD Collection

Title: *Empire Burlesque*
Artist: *Bob Dylan*

Title: *Hide your heart*
Artist: *Bonnie Tyler*

Title: *Greatest Hits*
Artist: *Dolly Parton*

Title: *Still got the blues*
Artist: *Gary Moore*

Title: *Eros*
Artist: *Eros Ramazzotti*

Title: *One night only*
Artist: *Bee Gees*

▶ 36

18:12

<pre> <?xml version="1.0" encoding="ISO-8859-1"?> <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:template match="/"> <html> <body> <h2>My CD Collection</h2> <xsl:apply-templates/> </body> </html> </xsl:template> <xsl:template match="cd"> <p> <xsl:apply-templates select="title"/> <xsl:apply-templates select="artist"/> </p> </xsl:template> <xsl:template match="title"> Title: <xsl:value-of select="."/>
 </xsl:template> <xsl:template match="artist"> Artist: <xsl:value-of select="."/>
 </xsl:template> </xsl:stylesheet> </pre>	<pre> <html> <body> <h2>My CD Collection</h2> <p> Title: Empire Burlesque
 Artist: Bob Dylan
</p> Title: Hide your heart
 Artist: Bonnie Tyler
</p> ***** </body> </html> </pre>
---	--

▶ 37
18:12

Ugrađena pravila

▶ XML dokument ima 7 vrsta čvorova:

1. Korenski
2. Čvorove elemenata
3. Atributa
4. Teksta
5. Komentara
6. Instrukcija za obradu
7. Prostora imena

Za svaku vrstu čvora postoji ugrađeno pravilo koje se primenjuje u slučaju kada se ne opiše u stilovima.

Pravila za tekst i atribute

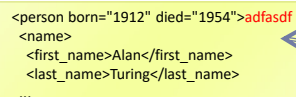
```
<xsl:template match="person">
  <xsl:value-of select="@atrName"/>
</xsl:template>
```

❖ Uzorak `@atrName` se podudara sa čvorovima atributa.

```
<xsl:template match="person">
  <xsl:value-of select="text()"/>
</xsl:template>
```

► Uzorak `text()` se podudara sa svim čvorovima teksta.

```
<person born="1912" died="1954">adfasdf
<name>
  <first_name>Alan</first_name>
  <last_name>Turing</last_name>
...
```



- ❖ Pomoću vertikalne crte mogu da se kombinuju oba uzorka.
- ❖ Šablon govori procesoru da se u izlazni dokument upiše vrednost svakog čvora (teksta ili atributa). Vrednost čvora teksta je sam tekst, a vrednost čvora atributa je vrednost atributa, a ne njegov naziv (ime).
- ❖ Napomena: `@` izdvaja atribut ne element, pa nema smisla da stoji uz match osim kao [predikat \(videti kasnije\)](#)
- ❖ Ako se selektuje jedno ili drugo (`text()|@*`), a postoji i jedno i drugo, biće prikazano ono drugo.

► 39

18:12

Primer – 3.13

- Šablonsko pravilo uzima vrednosti, ali njih bira eksplicitno definisano pravilo.

```
<?xml version="1.0" ?>
<people>
  <person born="1912"
    died="1954">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <profession>computer
    scientist</profession>

    <profession>mathematician</prof
    ession>

    <profession>cryptographer</profe
    ssion>
  </person>
</people>
```

► 40

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL"

  <xsl:template match="people">
    <html>
      <head><title>Famous Scientists</title></head>
      <body>
        <dl>
          <xsl:apply-templates/>
        </dl>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="person">
    <dt><xsl:apply-templates select="name"/></dt>
    <dd><ul>
      <li>Born: <xsl:apply-templates select="@born"/>
      <li>Died: <xsl:apply-templates select="@died"/>
    </ul></dd>
  </xsl:template>

</xsl:stylesheet>
```

```
<html>
<head>
  <META http-equiv="Content-Type"
    content="text/html; charset=utf-8">
  <title>Famous Scientists</title>
</head>
<body>
  <dl>
    <dt>
      Alan
      Turing
    </dt>
    <dd>
      <ul>
        <li>Born: 1912</li>
        <li>Died: 1954</li>
      </ul>
    </dd>
    <dt>
      Richard
      P
      Feynman
    </dt>
    <dd>
      <ul>
        <li>Born: 1918</li>
        <li>Died: 1988</li>
      </ul>
    </dd>
  </dl>
</body>
</html>
```

18:12

Podsećanje: Značenje nekih HTML tagova

- ▶ `<dl>` tag definiše definicionu listu, `<dt>` definiše stavku u listi a `<dd>` daje opis za stavku.
- ▶ `<li>` tag definiše jednu stavku liste (engl. *list item*).
- ▶ `<li>` tag se koristi za dve vrste listi, uređene i neuređene: (`<ol>`) odnosno (`<ul>`) lists.

```
<ol>
<li>Kafa
<li>Šećer
<li>Mleko
<li>Med
</ol>
```

```
1.Kafa
2.Šećer
3.Mleko
4.Med
```

```
<ul>
<li>Kafa
<li>Šećer
<li>Mleko
<li>Med
</ul>
```

```
• Kafa
• Šećer
• Mleko
• Med
```

```
<dl>
<dt>Kafa
<dd>Pijem je svakog jutra
<dt>Šećer
<dd>Volim ga s kafom
<dt>Mleko
<dd>Pijem ga svakog jutra
<dt>Med
<dd>Odličan je sa mlekom
</dl>
```

```
Kafa
Pijem je svakog jutra
Šećer
Volim ga s kafom
Mleko
Pijem ga svakog jutra
Med
Odličan je sa mlekom
```

▶ 41

18:12

Pravilo za korenski i ostale elemente

```
<xsl:template match="*" />
  <xsl:apply-templates/>
</xsl:template>
```

- ❖ Zvezdica `*` je *XPath* džokerski znak koji odgovara nekom (bilo kom ali jednom) čvoru elementa, bez obzira na imena i prostore imena. Ne sadrži atribut.
- ❖ Kosa crta `/` odgovara korenskom elementu.

▶ 42

18:12

Pravilo za čvorove komentara i instrukcije za obradu

```
<xsl:template match="processing-  
instruction() | comment()">  
</xsl:template>
```

- ❖ Podrazumevano šablonsko pravilo ništa ne ispisuje u izlazno stablo.
- ❖ Bez eksplicitnih pravila koja odgovaraju komentarima i instrukcijama za obradu, nijedan njihov deo se neće kopirati u izlazni dokument.

```
<person born="1912"  
died="1954">adfasdf  
<!--ime i prezime-->  
<?php instrukcija obrade....?>  
<name>  
  <first_name>Alan</first_name>  
  <last_name>Turing</last_name>  
</name>  
<!--profesije-->  
<profession>computer  
scientist</profession>  
</person>
```

```
<xsl:template match="comment()">  
  komentar je: "<xsl:value-of  
select="."/>"  
</xsl:template>
```

▶ 43

18:12

Pravilo za čvorove prostora imena

- ❖ Takođe, šablonsko pravilo koje odgovara čvorovima prostora imena kazuje procesoru da nijedan deo čvora prostora imena ne kopira u izlazni dokument.
- ❖ Ovo pravilo je ugrađeno u kodu XSLT procesora i ono se ne može napisati u XSLT opisu stilova, tj. ne postoji XPath uzorak koji bi odgovarao čvoru prostora imena.

▶ 44

18:12

XSLT za kvalifikovane elemente tj. koji su u definisanom imenskom prostoru - 1

- ▶ Uzorci identifikuju elemente pomoću URI identifikatora, njihovog lokalnog dela i prostora imena. Pri tome se prefiks prostora imena ne uzima u obzir.
- ▶ Na primer, za prethodni pimer, ako bi sve smestili u prostor imena <http://www.cafeconleche.org/namespaces/people>
- ▶ dobijamo dokument:

```
<?xml version="1.0"?>
<people
  xmlns="http://www.cafeconleche.org/namespaces/people"
>

  <person born="1912" died="1954">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <profession>computer scientist</profession>
    <profession>mathematician</profession>
    <profession>cryptographer</profession>
  </person>

  <person born="1918" died="1988">
    <name>
      <first_name>Richard</first_name>
      <middle_initial>P</middle_initial>
      <last_name>Feynman</last_name>
    </name>
    <profession>physicist</profession>
    <hobby>Playing the bongoes</hobby>
  </person>
</people>
```

▶ 45

18:12

XSLT za kvalifikovane elemente tj. koji su u definisanom imenskom prostoru - 2

- ▶ Osim ugrađenih pravila, na ovaj dokument ne utiče nijedno do sada definisano pravilo!
- ▶ Da bi pravila bila primenljiva potrebno je definisati prefiks za ciljani imenski prostor, a zatim pri definisanju pravila koristiti puna imena.

▶ 46

18:12

Primer upotrebe imenskih prostora

<pre><?xml <people xmlns="http://www.cafeconleche.org/namespaces/people"> <person born="1912" died="1954"> <name> <first_name>Alan</first_name> <last_name>Turing</last_name> </name> <profession>computer scientist</profession> <profession>mathematician</profession> <profession>cryptographer</profession> </person> <person born="1918" died="1988"> <name> <first_name>Richard</first_name> <middle_initial>P</middle_initial> <last_name>Feynman</last_name> </name> <profession>physicist</profession> <hobby>Playing the bongoes</hobby> </person> </people> </people></pre>	<pre><?xml version="1.0"?> <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:pe="http://www.cafeconleche.org/namespaces/people"> <xsl:template match="pe:people"> <html> <head><title>Famous Scientists</title></head> <body> <xsl:apply-templates/> </body> </html> </xsl:template> <xsl:template match="pe:name"> <p><xsl:value-of select="pe:last_name"/>, <xsl:value-of select="pe:first_name"/></p> </xsl:template> <xsl:template match="pe:person"> <xsl:apply-templates select="pe:name"/> </xsl:template> </xsl:stylesheet></pre>	<pre><html xmlns:pe="http://www.cafeconleche.org/namespaces/people"> <head> <META http-equiv="Content-Type" content="text/html; charset=utf-8"> <title>Famous Scientists</title> </head> <body> <p>Turing, Alan</p> <p>Feynman, Richard</p> </body> </html></pre>
	18:12	

Šabloni za kreiranje vrednosti atributa

- ▶ Unapred poznate vrednosti atributa se lako ubacuju u izlazni dokument kao doslovni sadržaj (literal).
- ▶ Na primer, ako se kreira atribut *class* i ako dodaje mu se vrednost "person":

```
<xsl:template match="person">
<span class="person"><xsl:apply-templates/></span>
</xsl:template>
```

Ako vrednost treba da se učitava iz dokumenta, koriste se šabloni vrednosti atributa. Šablon atributa je *XPath* je izraz u vitičastim zagradama i smešten u vrednost atributa unutar opisa stilova. Na primer, da bi dobili:

```
< puno_ime ime="Richard" srednje_slovo ="P" prezime ="Feynman"/>
```

```
<xsl:template match="person/name">
<puno_ime ime="{first_name}"
srednje_slovo="{middle_initial}"
prezime="{last_name}" />
</xsl:template>
```

Još neki elementi transformacije

49

18:12

```
<xsl:for-each select = "node-set-expression">
<!-- (xsl:sort*, template) -->
</xsl:for-each>
```

- ▶ Ovaj element se koristi za izdvajanje tj. selekciju svakog XML elementa za zadati *node-set*:

```
<xsl:template match="/">
  <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:for-each>
</xsl:template>
```

▶ 50

18:12

Filtriranje

- ▶ Može se izvršiti filtriranje izlaznih podataka iz nekog XML fajla **dodavajući kriterijum za `select` atribut u elementu `<xsl:for-each>`.**
- ▶ `<xsl:for-each select="catalog/cd[artist='Bob Dylan']">`

Operatori za filtriranje su:

= equal
!= not equal
< less than
> greater than

▶ 51

18:12

```
<xsl:sort select = "string-expression"
data-type = "text" | "number"
lang = "langcode"
order = "ascending" | "descending" case-order = "upper-first" | "lower-first" />
```

- ▶ Element koji obezbeđuje sortiranje na izlazu.
- ▶ **Gde se postavlja ova informacija?**
- ▶ Da bi se obezbedilo sortiranje, jednostavno se dodaje neki `<xsl:sort>` element unutar petlje tj. elementa `<xsl:for-each>` u XSL fajlu:

```
<xsl:for-each select="catalog/cd">
  <xsl:sort select="artist"/>
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
</xsl:for-each>
```

▶ 52

18:12

```
<xsl:if test = "boolean-expression">
<!-- template -- >
</xsl:if>
```

Koristi se za postavljanje uslova u odnosu na sadržaj XML fajla.

Sintaksa

```
<xsl:if test="expression">
...some output if the expression is true...
</xsl:if>
```

Primer

```
<xsl:for-each select="catalog/cd">
  <xsl:if test="price > 10">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:if>
</xsl:for-each>
```

▶ 53

18:12

Zadatak

- ▶ Napisati XSLT transformaciju koja će iz cdCatalog.xml formirati html fajl sa tabelom naslova i glumaca, tako da svi cd-ovi čija je cena veća od 10 budu sa zelenom pozadinom, a ostali sa plavom.

▶ 54

18:12

```
<xsl:variable name = "QualifiedName"
select = "expression">
<!-- template -->
</xsl:variable>
```

- Povezuje promenljivu odedenog tipa (string, number, node-set, itd.) i vrednost. Ova promenljiva može biti korišćena u izrazima koristeći **\$name**. Atribut name je obavezan. Atribut **select** je opcioni, a sadrži *XPath* izraz koji podešava vrednost variable. Ako je xsl:variable prazan element i nema select atribut, onda je i vrednost prazan string.

► 55

18:12

```

► <?xml version="1.0" encoding="ISO-8859-1"?>
  <xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Trans
  form">

  <xsl:variable name="header">
    <tr>
      <th>Element</th>
      <th>Description</th>
    </tr>
  </xsl:variable>

  <xsl:template match="/">
    <html>
      <body>
        <table>
          <xsl:copy-of select="$header" />
          <xsl:for-each select="reference/record">
            <tr>
              <xsl:if category="XML">
                <td><xsl:value-of select="element"/></td>
                <td><xsl:value-of select="description"/></td>
              </xsl:if>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
  <td><xsl:value-of select="description"/></td>
</xsl:if>
</tr>
</xsl:for-each>
</table>
<br />
<table>
  <xsl:copy-of select="$header" />
  <xsl:for-each select="table/record">
    <tr>
      <xsl:if category="XSL">
        <td><xsl:value-of select="element"/></td>
        <td><xsl:value-of select="description"/></td>
      </xsl:if>
    </tr>
  </xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

► 56

18:12

```
<xsl:element name="name" namespace="URI"
use-attribute-sets="namelist">
</xsl:element>
```

► Kreira odgovarajući elemenat

```
► <?xml version="1.0" encoding="utf-8"?>
► <xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
►   <xsl:template match="/">
►     <xsl:for-each select="catalog/cd">
►       <xsl:element name="singer">
►         <xsl:value-of select="artist" />
►       </xsl:element>
►       <br />
►     </xsl:for-each>
►   </xsl:template>
► </xsl:stylesheet>
```

► 57

18:12

```
<xsl:copy-of select="expression"/>
```

- Kopira u izlazni dokument sve što identifikuje *select*.
Podčvorovi, prostori imena, atributi, automatski se kopiraju!

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:variable name="header">
    <tr>
      <th>Element</th>
      <th>Description</th>
    </tr>
  </xsl:variable>
  <xsl:template match="/">
    <html>
      <body>
        <table>
          <xsl:copy-of select="$header" />
          <xsl:for-each select="reference/record">
            <tr>
              <xsl:if test="category='XML'">
                <td>
                  <xsl:value-of select="element"/>
                </td>
                <td>
                  <xsl:value-of select="description"/>
                </td>
              </xsl:if>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

► 58

18:12

Primer

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="family.xsl"?>
<family>
  <person>
    <given-name age="10">
      <name>Fred</name>
      <nick-name>Freddy</nick-name>
    </given-name>
    <family-name>Smith</family-name>
  </person>
  <person>
    <given-name age="10">
      <name>Robert</name>
      <nick-name>Bob</nick-name>
    </given-name>
    <family-name>Smith</family-name>
  </person>
</family>
```

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

  <xsl:template match="person">
    <p>
      <xsl:copy-of select="given-name"/>
      <xsl:text> </xsl:text>
      <xsl:copy-of select="family-name"/>
    </p>
  </xsl:template>

</xsl:stylesheet>
```

```
<?xml version="1.0"?>
<p><given-name age="10">
  <name>Fred</name>
  <nick-name>Freddy</nick-name>
</given-name>
  <family-name>Smith</family-name></p>
<p><given-name age="10">
  <name>Robert</name>
  <nick-name>Bob</nick-name>
</given-name>
  <family-name>Smith</family-name></p>
```

► 59 18:12

<xsl:attribute name = "QualifiedName"
namespace = "URI"> </xsl:attribute>

► Dodaje atribut određenom elementu u stablu rezultata.

Primer 1

```
<xsl:template match="pe:name">
  <ppp>
    <xsl:attribute name="name" >
      <xsl:value-of select="pe:first_name"/>
    </xsl:attribute>
    <xsl:value-of select="pe:last_name"/>,
    <xsl:value-of select="pe:first_name"/>
  </ppp>
</xsl:template>
```

Dodaje se name atribut elementu **ppp** i dodeljuje mu vrednost iz elementa first_name

Primer 2

```
<IMG>
  <xsl:attribute name="src">
    <xsl:value-of select="imagenames/imagenam">
  </xsl:attribute>
</IMG>
```


 Dodaje se **src** atribut elementu IMG

► 60 18:12

```
<xsl:attribute-set name = "QualifiedName" use-
attribute-sets = "QualifiedNames"> </xsl:attribute-set>
```

- ▶ Definiše jednu kolekciju atributa koja se može primeniti na elemente bilo gde u dok. Može se koristiti kao atribut u sledećim elementima primenom atributa use-attribute-sets:

- ▶ xsl:element
- ▶ xsl:copy
- ▶ xsl:attribute-set

```
<xsl:template match="chapter/heading">
  <fo:block quadding="start" xsl:use-attribute-sets="title-style">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>
<xsl:attribute-set name="title-style">
  <xsl:attribute name="font-size">12pt</xsl:attribute>
  <xsl:attribute name="font-weight">bold</xsl:attribute>
</xsl:attribute-set>
```

▶ 61

18:12

<xsl:choose>

```
<!-- (xsl:when+, xsl:otherwise?) --> </xsl:choose>
```

- ▶ Koristi se zajedno sa elementima
- ▶ <xsl:when test=""> i <xsl:otherwise>
- ▶ za iskazivanje višestrukih provera.

Sintaksa:

```
<xsl:choose>
  <xsl:when test="expression">
    ... some output ...
  </xsl:when>
  <xsl:otherwise>
    ... some output ....
  </xsl:otherwise>
</xsl:choose>
```

▶ 62

18:12

Primer 1

```
....
<xsl:for-each select="catalog/cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <xsl:choose>
      <xsl:when test="price > 10">
        <td bgcolor="#ff00ff">
          <xsl:value-of select="artist"/></td>
        </xsl:when>
      <xsl:otherwise>
        <td><xsl:value-of select="artist"/></td>
      </xsl:otherwise>
    </xsl:choose>
  </tr>
</xsl:for-each>
....
```

▶ 63

18:12

Primer 2

```
....
<xsl:for-each select="catalog/cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <xsl:choose>
      <xsl:when test="price > 10">
        <td bgcolor="#ff00ff">
          <xsl:value-of select="artist"/></td>
        </xsl:when>
      <xsl:when test="price > 9">
        <td bgcolor="#cccccc">
          <xsl:value-of select="artist"/></td>
        </xsl:when>
      <xsl:otherwise>
        <td><xsl:value-of select="artist"/></td>
      </xsl:otherwise>
    </xsl:choose>
  </tr>
</xsl:for-each>
....
```

▶ 64

18:12