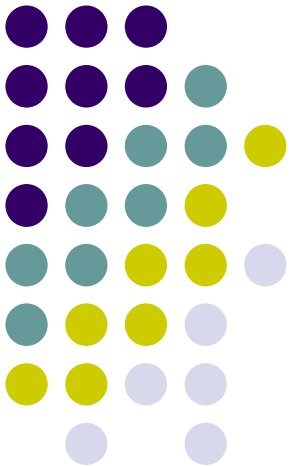
Klijentske tehnologije

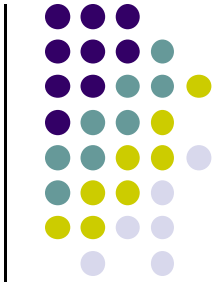
XML

Maja Štula

ak. god. 2011/2012

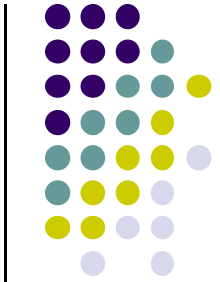


XML



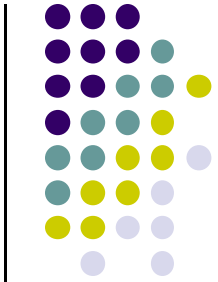
- XML (*eXtensible Markup Language*) – je *markup* jezik (jezik koji kombinira sadržaj koji se objavljuje sa dodatnim podatcima o tome sadržaju) koji se koristi za strukturiranje, pohranjivanje i slanje podataka. XML standard je razvio W3C da bi omogućio jednostavan, otvoren i standardiziran način pohrane samo-opisujućih podatka (*self-describing data*) (podaci opisuju i svoj sadržaj i svoju strukturu).
- Aktualne verzije su:
  - Extensible Markup Language (XML) 1.0 (*Fifth Edition*) (11.2008)
  - Extensible Markup Language (XML) 1.1 (*Second Edition*) (8.2006)

XML



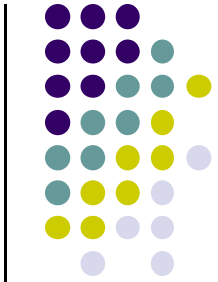
- XML promatran zasebno nije “moćna” tehnologija. Njegova glavna prednost je u tome što je postao “de facto” standardni format podataka koji podržavaju različite aplikacije, različite platforme.
- Npr. podatke iz MSSQL (Microsoft SQL) baze možete jednostavno preoblikovati u XML format i takve ih razmjenjivati sa drugim aplikacijama koje će podatke dobivene u XML obliku jednostavno preoblikovati u vlastiti format. Na taj način mogu komunicirati potpuno različite aplikacije, operacijski sustavi, …
- Do XML nije postojao format zapisa podataka koji je bio toliko proširen i prihvaćen.
- XML format zapisa podržavaju gotovo sve aplikacije i implementiran je u gotovo svim programskim jezicima (postoje gotove biblioteke funkcija koje omogućavaju konverziju iz i u XML oblik, parsiranje i kreiranje XML dokumenata …..) (System.Xml Namespace .net, java paket javax.xml.parsers,…).

Prednosti XML-a



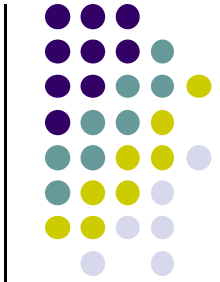
- Format je prilagođen čovjeku,
- podržava Unicode pa se može koristiti sa bilo kojim ljudskim jezikom,
- stroga sintaksa čini parsiranje jednostavnim i efikasnim,
- proširen je (svi ga koriste i podržavaju),
- baziran je na otvorenim standardima,
- XML je tekstualna datoteka (takav format je manje restriktivan od binarnog),
- neovisan je o platformi, pa ga je lakše prenijeti na nove tehnologije (npr. mobitel).

Nedostatci XML-a



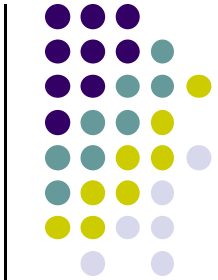
- Podaci zapisani prema XML sintaksi su redundantni i zauzimaju više mjesta u odnosu na zapis u binarnom obliku,
- XML sintaksa je preopširna,
- ne postoji ugrađena podrška za tipove podataka tj. ne postoji notacija za označavanje tipa podatka poput datuma, stringa i sl.
- hijerarhijski model XML-a je ograničen u odnosu na relacijski model podataka,
- XML imenski prostori su složeni za implementaciju u parseru,
- izražavanje veza među XML čvorovima koje nisu hijerarhijske (*overlapping node relationships*) nije definirano u XML-u.

XML



- Ispravnost XML dokumenta definira se na dva načina:
  - Ispravno formatiran (*Well-formed*) XML dokument prilagođen je sintaktičkim pravilima XML. Npr. ako element dokumenta ima otvoreni tag (<ime>) i nema zatvoreni tag (</ime>), a tag nije samo-zatvarajući (*self-closing*) (<ime/>), tada taj dokument nije ispravno formatiran. Takav dokument se ne smatra XML dokumentom i parser koji vodi računa o ispravnosti dokumenta ga neće parsirati (neispravno_formatiran.xml).
  - Ispravan (*Valid*) XML dokument osim poštivanja sintaktičkih pravila poštuje i semantička pravila koja su obično uključena u XML shemu dokumenta. Npr. ako dokument sadrži tag koji nije definiran u XML shemi dokumenta, bez obzira što je uredno otvoren i zatvoren parser koji provjerava validaciju XML dokumenta ga neće parsirati. (neispravno_validiran.xml)

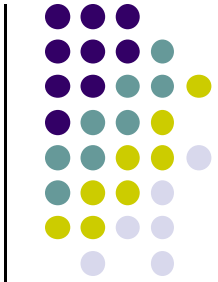
Neispravan XML dokument



```
<?xml version="1.0" encoding="ISO-8859-2"?>
<!DOCTYPE note [
    <!ELEMENT STUDENTI (OSOBA)>
    <!ELEMENT OSOBA (IME, GODINA_STUDIRANJA)>
    <!ELEMENT IME (#PCDATA)>
    <!ELEMENT GODINA_STUDIRANJA (#PCDATA)>
]>
<STUDENTI>
    <OSOBA>
        <IME>Mate</IME>
        <PREZIME>Matić</PREZIME>
        <GODINA_STUDIRANJA>3</GODINA_STUDIRANJA>
    </OSOBA>
    <OSOBA>
        <IME>Ivo</IME>
        <PREZIME>Ivić</PREZIME>
        <GODINA_STUDIRANJA>3</GODINA_STUDIRANJA>
    </OSOBA>
</STUDENTI>
```

DTD je interni tj. sadržan je u XML dokumentu. Tag PREZIME nije definiran u DTD-u dokumenta.

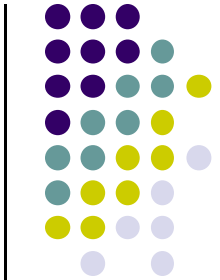
Uključivanje vanjskog DTD dokumenta



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
   <to>Tove</to>
   <from>Jani</from>
   <heading>Reminder</heading>
   <body>Don't forget me this weekend!</body>
</note>
```

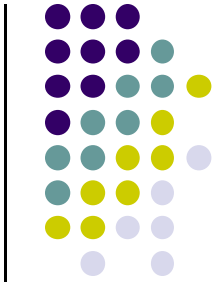
- Isto kao i kod HTML-a u jednoj liniji na početku dokumenta u DOCTYPE (*document type*) deklaraciji.

Binarni podaci u XML-u



- XML dokument je tekstualni dokument.
- Kada se u XML dokument uključuju binarni podaci potrebno je koristiti base64 shemu kodiranja.
- Base64 shema kodiranja je specificirana u RFC 2045 - MIME (*Multipurpose Internet Mail Extensions*) (1521 je *obsolete*).
- Ta je shema napravljena za predstavljanje proizvoljnog niza okteta (tj. bajtova) u tekstualnom formatu.
- Algoritmi za kodiranje (iz binarnog u tekstualni Base64 zapis) i dekodiranje su jednostavni.

Binarni podaci u XML-u



- Shema koristi skup od 64 karaktera (A-Z, a-z, 0-9, +, /) za predstavljanje binarnih podataka plus znak "=" za nadopunu (*padding*).
- Base64 shema binarne podatke obrađuje u grupama od 24 bita, mapirajući ih u 4 kodirana karaktera. Stoga se ponekada naziva i 3-u-4 kodiranje (*3-to-4 encoding*) jer se od 3 okteta podataka dobije 4 karaktera.
- Svaki od 6 bita (2^6=64) unutar grupe od 24 bita je indeks koji se koristi u tablici sa 64 karaktera da bi se dobio karakter koji kodira tih 6 bita.

Base64 skup karaktera

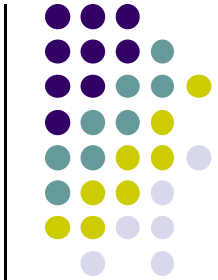
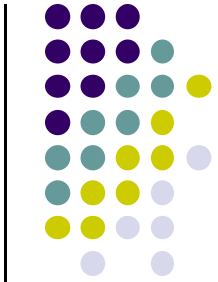


Table 1: The Base64 Alphabet

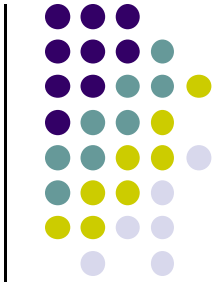
Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

Binarni podaci u XML-u



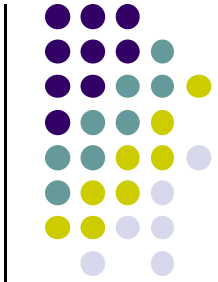
- Primjer xml dokumenta sa binarnim podacima zapisanim u elementu SLIKACIJELA xml_sa_binarnim_podacima.xml
- Sadržaj zapisan u tom elementu je slika u .jpg formatu koja je prebačena u base64 zapis preko on-line base64 enkoder http://www.greywyvern.com/code/php/binary2base64
- Kako dohvatiti podatke koji su u XML dokumentu zapisani binarno tj. tekstualno u formatu base64?
- Trebalo bi ih na neki način parsirati opet u originalni format i onda prikazati.

Binarni podaci u XML-u



```
<?xml version="1.0" encoding="ISO-8859-2"?>
<JA>
<IME>Maja </IME>
<SLIJA>
    D/2wBDAAEBAQEBAQEBAQEBAQECAgMCAgICAgQDAwIDB
    QQFBQUEBAQFBgcGBQUHBgQEBgkGBwgICAgIBQYJCgkICg
    cICAj/2wBDAQEBAQICAgQCAgQIBQQFCAgICAgICAgICA
    gICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAj/wA
    ARCACqAJYDASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAA
    AAAAAAAECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQ
    AAAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0f
    AkM2JyggkKFhcYGRolJicoKSo0NTY3ODk6Q0RFRkdISUpTVF
    VWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh4iJipKTlJWWl5
    iZmqK……..</SLIJA>
</JA>
```

Binarni podaci u XML-u



- xml_sa_binarnim_podacima_konverzija.xml
- prikazi_sliku.xsl

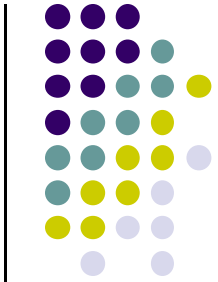
xsl

```
<img>
  <xsl:attribute name="src">
  <xsl:value-of select="/JA/SLIKACIJELA/@izvor" />
  </xsl:attribute>
</img>
```

xml

```
<SLIKACIJELA
    izvor="
    AD/2wBDAAEBAQEBAQEBAQEBAQECAgMCAgICAgQDAwID
    BQQFBQUEBAQFBgcGBQUHB……..
```

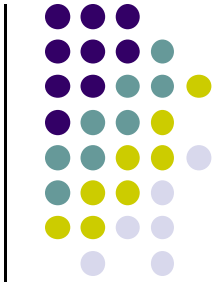
Binarni podaci u XML-u



- Ukoliko se radi o slici (kao u primjeru), slika se može prikazati u pretraživaču korištenjem XSL (*Extensible Stylesheet Language*) jezika za formatiranje XML podataka u HTML format ili neki drugi format. Naime u primjeru se u XSL dokumentu, u *img* tagu kao izvor slike navodi podatkovni URI (*data scheme URI*) specificiran u RFC 2397. Podatkovni URI je specijalni oblik URI-a kod kojeg se podatci sadržani u resursu koji se dohvaća navode u samom URI-ju.
- Kada se koristi podatkovni URI i ako pretraživač podržava podatkovni URI (većina pretraživača da) ponašanje je jednako kao da je resurs uključen običnim URI-jem. Prvi dio URI-ja je shema, pa opcionalno MIME tip, pa opcionalno kodiranje i na kraju podaci:

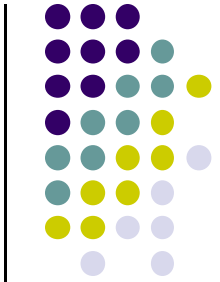
AQEBAQEBAQECAgMCAgICAgQDAwIDBQQFBQUEBAQFBgcGBQUHB….

XML



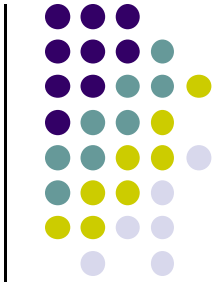
- Dizajniranje XML jezika ili “dijalekata” je jednostavno zbog stroge sintakse XML jezika. To su jezici bazirani na XML-u.
- Primjeri formalno definiranih jezika baziranih na XML-u su RSS (*Really Simple Syndication*) jezik za opisivanje sadržaja web stranica, MathML (*Mathematical Markup Language*) za predstavljanje matematičkih izraza, XHTML, …
- Za definiranje sintakse XML dokumenta tj. za provjeru ispravnosti (validacija) XML dokumenta razvijen je niz shema jezika, ali se najčešće koriste DTD (*Document Type Definition*), W3C XML Shema (WXS) (drugi naziv je XSD (*XML Schema Definition*)) i RELAX NG.
- XSD je novija verzija DTD-a, koristi se za opisivanje strukture i sadržaja XML dokumenata.

XML prolog



- XML dokument započinje sa dijelom koji se zove XML prolog.
- Treba biti naveden na početku XML dokumenta prije *root* elementa.
- Prolog se sastoji od XML deklaracije i/ili (opcionalno) DTD.
- XML deklaracija sadrži verziju XML-a te opcionalno tip kodiranja dokumenta i *standalone* deklaraciju.

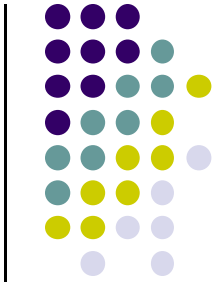
XML prolog



- Deklaracija kodiranja nije obavezna, ali je preporučena kako bi XML parser mogao točno parsirati dokument. Ukoliko nije navedena parser će sam pretpostaviti način kodiranja dokumenta.
- Preporuča se korištenje označavanja načina kodiranja definiranog od IANA organizacije (*Internet Assigned Numbers Authority*).
- Atribut *standalone* se odnosi na to da li se ovaj dokument veže na neki način sa drugim dokumentom (npr. specifikacija tipa dokumenta).
- Primjeri:

1. <?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
2. <?xml version="1.0"?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

XML



- *Tagovi* označavaju početak i kraj XML elementa:

  Početni tag: <tag>

  Krajnji tag: </tag>

- Prazni elementi zahtijevaju samo jedan *tag*:

  <tag/>

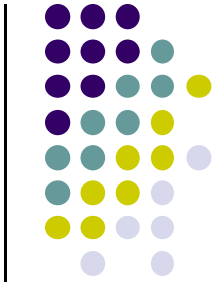
- *Tagovi* su *case-sensitive*:

  <TAG> <Tag> <tag>

- XML komentar je isti kao HTML komentar:

  < ! -- Ovo je komentar -- >

XML



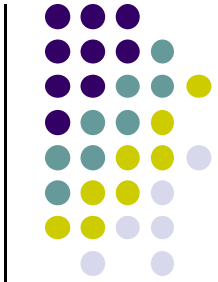
- XML elementi se sastoje od početnog *taga*, završnog *taga* i podataka koji se nalaze između njih

```
<grad> Split </grad>
```

- Elementi moraju biti pravilno ugniježđeni i zatvoreni

```
<gradovi>
    <grad>Split</grad>
    <grad>Dubrovnik</grad>
</gradovi>
```

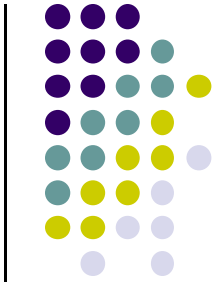
XML



- Svi XML dokumenti moraju imati *root* element.
- Sve ostale elemente moramo smjestiti unutar *root* elementa.
- Elementi mogu imati pod-elemente (*child*). *Child* elementi moraju biti pravilno ugniježđeni unutar *parent* elementa:

```
<root>
        <child>
                <subchild>.....</subchild>
        </child>
</root>
```

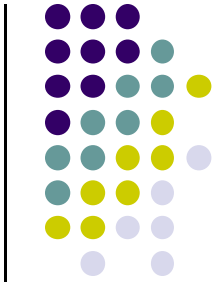
XML



- Elementi mogu sadržavati atribute. Atributi imaju ime i vrijednost.
- Ime atributa i njegova vrijednosti pišu se unutar istog početnog *taga*. Vrijednosti atributa se obavezno pišu u navodnim znakovima (navodni znakovi mogu biti jednostruki ili dvostruki)

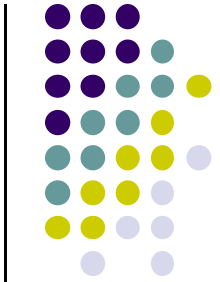
<grad postanski_broj='21000'> Split </grad>

XML



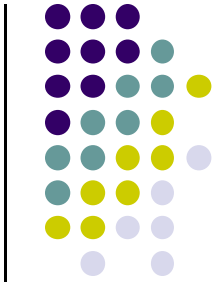
- *Tagovi* u HTML-u su predefinirani, dok kod XML-a korisnik sam definira *tagove* koji su mu potrebni za strukturiranje podataka.
- Korisnik sam definira XML *tagove* poštujući sljedeća pravila:
  - *tagovi* moraju započeti sa slovnim znakom
  - ne smiju sadržavati razmak
  - mogu sadržavati brojeve, ali ne na početku imena
  - ne smiju započeti sa znakovima “xml”, uključujući bilo koji varijaciju temeljenu na malim i velikim slovima

XML imenski prostor



- Kako u XML-u korisnik sam definira elemente i njihove oznake vjerojatno je da će doći do kolizije imena.
- Način rješavanja kolizije je korištenje imenskih prostora.
- XML imenski prostor se definira u početnom tagu korištenjem atributa “xmlns:“ (*XML NameSpace*).
- Atribut “xmlns:“ se koristi kao prefiks koji se dodaje imenima (oznakama) imenskih prostora koji se kreiraju. Dvotočka odvaja ime imenskog prostora od prefiksa xmlns.
- Npr. <nekitag xmlns:moj=“Neki moj prostor”>
- Kada je imenski prostor definiran u početnom tagu svi child elementi sa istim prefiksom kao što se zove imenski prostor su povezani sa tim imenskim prostorom.
- Često se za vrijednost xmlns postavlja nekakav URI.

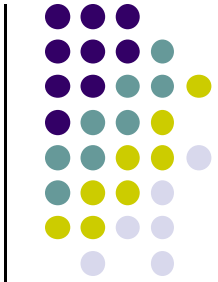
XML imenski prostor



```
<?xml version="1.0"
    encoding="ISO-8859-15"?>
<html>
<body>
<p>Welcome to my Health
    Resource
</p>
</body>
<body>
<height>6ft</height>
<weight>155 lbs</weight>
</body>
</html>
```

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<html:html
    xmlns:html='http://www.w3.org/TR/xhtml1/'>
<html:body>
<html:p>Welcome to my Health Resource
</html:p>
</html:body>
<health:body
    xmlns:health='http://www.example.org/health'>
<health:height>6ft</health:height>
<health:weight>155 lbs</health:weight>
</health:body>
</html:html>
```

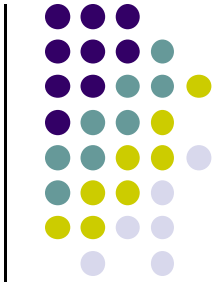
XML imenski prostor



- Imenski prostori mogu se definirati i bez eksplicitnog navođenja oznake imenskog prostora, tj. korištenjem samo atributa xmlns.
- Atribut xmlns definira *defaltni* imenski prostor koji nasljeđuju svi child elementi.

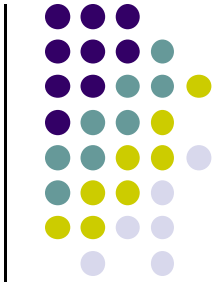
```
<?xml version="1.0" encoding="ISO-
    8859-15"?>
<html
    xmlns='http://www.w3.org/TR/xhtml1
    /'>
<body>
<p>Welcome to my Health Resource
</p>
</body>
<body
    xmlns='http://www.example.org/heal
    th'>
<height>6ft</height>
<weight>155 lbs</weight>
</body>
</html>
```

XML



- XML dokument ne nosi informaciju kako prikazati podatke.
- Formatiranje prikaza XML dokumenta radi se obično korištenjem CSS, XSL, JavaScript-a, ili XML *Data Islands*.
- XSL (*EXtensible Stylesheet Language*) je XML *Style Sheets* i opisuje kako XML dokument treba izgledati.
- XSL je jezik za prevođenje XML-a u XHTML format, kojim se mogu filtrirati, sortirati i formatirati XML podaci.

XML i CSS



Uključivanje CSS dokumenta

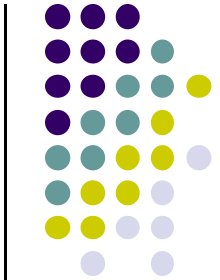
```
<?xml version="1.0"
    encoding="ISO-8859-1" ?>
<?xml-stylesheet type="text/css"
    href="cd_catalog.css"?>
<CATALOG>
 <CD>
   <TITLE>Empire
    Burlesque</TITLE>
   <ARTIST>Bob
    Dylan</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Columbia</COMP
    ANY>
  <PRICE>10.90</PRICE>
  <YEAR>1985</YEAR>
 </CD>
 </CATALOG>
```

catalog.xml

```
CATALOG { background-color:
    #ffffff; width: 100%; }

CD { display: block; margin-
    bottom: 30pt; margin-left: 0; }

TITLE { color: #FF0000; font-size:
    20pt; }

ARTIST { color: #0000FF; font-
    size: 20pt; }

COUNTRY,PRICE,YEAR,COMP
    ANY { display: block; color:
    #000000; margin-left: 20pt; }
```

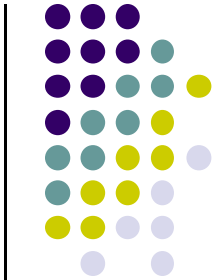
cd_catalog.css

XML i JavaScript



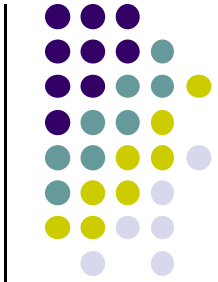
- Za čitanje, nadograđivanje, stvaranje i korištenje XML dokumenata, trebamo XML parser.
- XML parser je sastavni dio većine preglednika.
- Preglednici učitavaju XML dokumente na različite načine.
- U Internet Exploreru možete koristi ActiveX kontrolu koja se zove Microsoft.XMLDOM.

XML i JavaScript



- Firefox (http://developer.mozilla.org/en/docs/XML_Extras) i Opera naravno ne podržavaju ActiveX već imaju na drugi način implementiran XML parser.
- Također je moguće koristiti i XMLHttpRequest objekt za dohvaćanje XML datoteke.
- XMLHttpRequest je W3 konzorcijum standardiziran API koji omogućava klijentu dohvaćanje podataka sa servera (znači ne možete koristiti ovaj objekt za dohvaćanje lokalnog XML dokumenta sa diska), a koristi se kao osnovni dio AJAX tehnologije.

FireFox i XML podrška



Object name	Description
XMLDocument	Supports the standard W3C DOM Document implementation, and can be extracted from the web page document object.
XMLSerializer	Used for converting XML DOM Documents into either a string or streamed binary serial output.
DOMParser	Converts a stream, file or stream representation of an XML document into an internal DOM Document
XSLTProcessor	Creates an XSLT object that can be used for building transformations
XPath Parser	Performs XPath queries on XML documents.
Web Services	Handles SOAP and WSDL Services
XUL/XBL	These components are useful for adding functionality into the browser itself, and are XML based.

- Na slici su prikazani objekti implementirani u FireFox-u za rad sa XML dokumentima.
- XMLDocument objekt se ne poziva direktno već preko sučelja *document.implementation*.

Izvor: http://metaphoricalweb.blogspot.com/2004/08/contender.html

IE i XML podrška

Objekt window.ActiveXObject postoji u IE 5, 6, 7

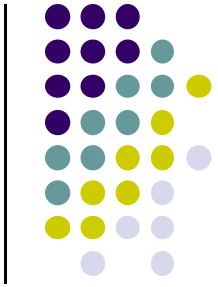
```
if (window.ActiveXObject)
{
var progIDs =
      ["Msxml2.DOMDocument.7.0","Msxml2.DOMD
      ocument.6.0“,"Msxml2.DOMDocument.5.0",
      "Msxml2.DOMDocument.4.0","Msxml2.DOMD
      ocument.3.0","MSXML2.DOMDocument","MSX
      ML.DOMDocument", "Microsoft.XMLDOM"];
for (var i = 0; i < progIDs.length; i++)
{
  try { alert("Pokušaj kreiranja XML parsera verzije "
      + progIDs[i]);
    var xmlDOM = new ActiveXObject(progIDs[i]); }
  catch (ex)
{ alert("Neuspio pokušaj kreiranja XML parsera
      verzije " + progIDs[i]);}
  }
}
```

PRIMJER

- Preporuka je korištenje XML parsera verzije MSXML 6.0, jer je najviše u skladu sa W3 standardima, najsigurniji i najbolje testiran. No starije verzije OS-ova je ne moraju imati instaliranu.
- Verzija MSXML 3.0 je instalirana na gotovo svim OS-ovima.

Izvor: http://blogs.msdn.com/xmlteam/archive/2006/10/23/using-the-right-version-of-msxml-in-internet-explorer.aspx

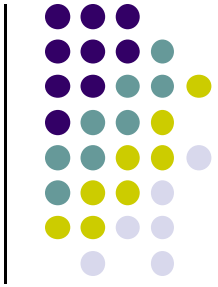
XML i JavaScript + XML parser objekt



```
function ucitavanje_xml(ime_xml_datoteke)
{
    var xmlDoc;
    // code for IE
    if (window.ActiveXObject)
      {
      var xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
      }
    // code for Mozilla, Firefox, Opera, etc.
    else
      {
      var xmlDoc=document.implementation.createDocument("","",null);
      }
     xmlDoc.async=false;
     xmlDoc.load(ime_xml_datoteke);
     return xmlDoc;
}
```

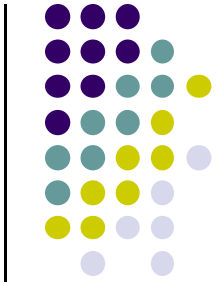
PRIMJER

XML i JavaScript + XML parser objekt



```
// kreiranje instance Microsoft XML parsera
var xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
// isključivanje nesinkroniziranog učitavanja koje osigurava da parser
// neće nastaviti izvršavanje prije nego je učitan cijeli dokument
xmlDoc.async="false";
//učitavanje XML dokumenta
xmlDoc.load("catalog.xml");

// kreiranje instance FireFox XML parsera
var xmlDoc=document.implementation.createDocument("","",null);
// Prvi parametar je imenski prostor koji će se koristiti uz XML dokument (ovdje
// je prazan “”,  drugi parametar je XML root element, a treći je uvijek null jer još
// nije implementiran.
// isključivanje nesinkroniziranog učitavanja koje osigurava da parser
// neće nastaviti izvršavanje prije nego je učitan cijeli dokument
xmlDoc.async="false";
//učitavanje XML dokumenta
xmlDoc.load("catalog.xml");
```

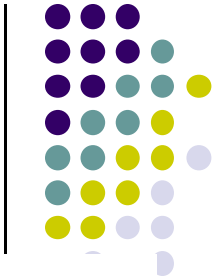
XML i JavaScript + XMLHTTPRequest objekt



```
function ucitavanje_xml(url)
{
      if(window.XMLHttpRequest)
      {
          var Loader = new XMLHttpRequest();
          Loader.open("GET", url ,false);
          Loader.send(null);
          return Loader.responseXML;
       }
}
```

PRIMJER

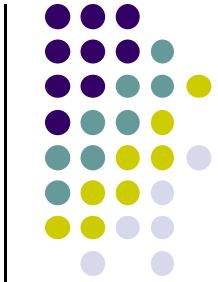
XML i JavaScript



```
var dokument=ucitavanje_xml("catalog.xml");
var dijeca = dokument.childNodes;
for(i=0;i<dijeca.length;i++)
{
	document.write("Ime čvora: <br />" + dijeca[i].nodeName + "<br /><br />");
	document.write("Vrijednost čvora:<br /> " + dijeca[i].text + "<br />");
	document.write("----------<br />");
	var dijeca_djece = dijeca[i].childNodes;
	for(j=0;j<dijeca_djece.length;j++)
	{
		document.write("Ime čvora: <br />" + dijeca_djece[j].nodeName + "<br /><br />");
		document.write("Vrijednost čvora:<br /> " + dijeca_djece[j].text + "<br />");
		document.write("++++++++++++++++++++++++++++++++<br />");
		var dijeca_djece_2 = dijeca_djece[j].childNodes;
		for(k=0;k<dijeca_djece_2.length;k++)
		{
		document.write("Ime čvora: <br />" + dijeca_djece_2[k].nodeName + "<br /><br />");
		document.write("Atribut gender čvora: <br />" + dijeca_djece_2[k].getAttribute('GENDER') +
	"<br /><br />");
		document.write("Vrijednost čvora:<br /> " + dijeca_djece_2[k].text + "<br />");
		document.write("**********<br />");
		}
    }
}
```

PRIMJER:

XML *Data Islands*

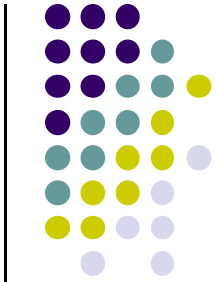


- *Data Islands* može služiti za pristup XML datoteci (podržano samo u IE i to ne u novijim verzijama IE poput IE8).
- U ovom primjeru , XML datoteka "cd_catalog.xml" će se učitati u “nevidljivi" data Island nazvan "xmldso".

```
<html><body>
<xml src=“cd_catalog.xml” id=“xmldso” async=“false”>
</xml>
<table border="1" datasrc="# xmldso"> <tr>
<td><span datafld="ARTIST"></span></td>
<td><span datafld="TITLE"></span></td> </tr> </table>
</body></html>
```

Primjer:

XSL



- XSL (*Extensible Stylesheet Language*) je jezik za transformiranje i formatiranje XML dokumenta u drugi XML dokument (tipično XHTML format) (W3C standard).
- Uključivanje XSL u XML dokument radi se na početku XML dokumenta iza XML prologa:

  <?xml version="1.0" encoding="ISO-8859-1"?>

  <?xml-stylesheet type="text/xsl" href="ime.xsl"?>

- PRIMJER: simple.xml simple.xsl

- XSL dokument ima .xsl ekstenziju, a počinje sa:

  <?xml version="1.0"?> ← XML deklaracijom

XSL

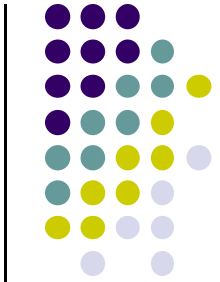
simple.xml

```
<?xml version="1.0"
    encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl"
    href="simple.xsl"?>
<breakfast_menu>
<food>
<name>Belgian Waffles
</name>
<price>$5.95</price>
<description> two of our famous
    Belgian Waffles
</description>
<calories>650</calories>
</food>
</breakfast_menu>
```

simple.xsl

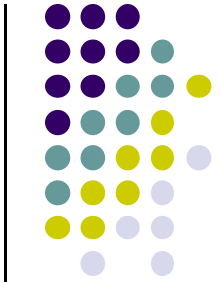
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<html xsl:version="1.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns="http://www.w3.org/1999/xhtml">
 <body style="font-family:Arial,helvetica,sans-serif;font-
      size:12pt;
       background-color:#EEEEEE">
  <xsl:for-each select="breakfast_menu/food">
   <div style="background-
    color:teal;color:white;padding:4px">
     <span style="font-weight:bold;color:white">
     <xsl:value-of select="name"/></span>
     - <xsl:value-of select="price"/>
   </div>
   <div style="margin-left:20px;margin-bottom:1em;font-
    size:10pt">
     <xsl:value-of select="description"/>
     <span style="font-style:italic">
       (<xsl:value-of select="calories"/> calories per
    serving)
     </span>
   </div>
  </xsl:for-each>
 </body>
</html>
```

XSL



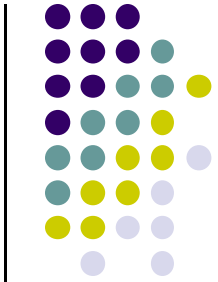
- XSL *stylesheet* procesor prihvaća dokument ili podatke u XML-u i jedan XSL *stylesheet* i stvara prezentaciju XML izvornog sadržaja na način definiran u XSL *stylesheetu*. Proces stvaranja prezentacije se sastoji od dva dijela:
  1. Prvi dio je kreiranje rezultirajućeg stabla na osnovu izvornog XML stabla.
  2. Drugi dio je formatiranje rezultirajućeg stabla u prezentaciju prilagođenu mediju prezentacije (na ekran, na papir, u govorni oblik ili drugi medij).
- Prvi dio se naziva transformacija stabla (*tree transformation*) (XML parser), a drugi se naziva formatiranje (*formatting*). Formatiranje najčešće radi mehanizam prikaza preglednika.
- Transformacija stabla omogućava mijenjanje strukture izvornog XML stabla. Npr. podaci iz originalnog XML stabla mogu se sortirati po abecedi. Prilikom generiranja rezultirajućeg stabla dodaju se i informacije o formatiranju prezentacije.

XSL



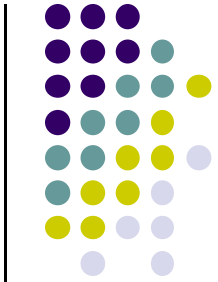
- XSL se sastoji od tri dijela:
  - XSLT (*XSL Transformations*) - jezik za prevođenje XML dokumenata
  - XPath - jezik za navigaciju u XML dokumentima
  - XSL-FO (*xsl formatting*) – sintaksa i semantika XSL-a za formatiranje XML dokumenata
- Transformacija stabla je definirana XSLT-om. U XSL-u rezultirajuće stablo se naziva stablo elemenata i atributa (*element and attribute tree*) sa objektima koji se nalaze u imenskom prostoru objekata formatiranja. Objekt formatiranja je predstavljen kao XML element, sa svojstvima predstavljenim parovima XML atributa-vrijednosti. Sadržaj objekata formatiranja je sadržaj XML elemenata.

XPath



- XPath je W3C standard sa sintaksom za definiranje određenih dijelova XML dokumenta.
- Koristi izraze koji pokazuju putanju za navigaciju unutar XML dokumenta.
- Pomoću tih izraza XSLT selektira čvorove i njihove podskupove unutar XML dokumenta.
- XPath sadrži biblioteku standardnih funkcija.

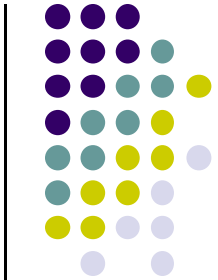
XPath



- **XML** dokumente tretiramo kao stabla čvorova. Korijen stabla zovemo čvor dokumenta (*document node* ili *root node*).**XPath** ima više vrsta čvorova:
  - Čvor dokumenta (*root*)
  - Element
  - Atribut
  - Tekst
  - Instrukcija procesa
  - Komentar

- Čvorovi koji nemaju dijete nazivaju se **Atomi**

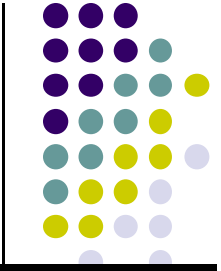
XPath



- **Roditelj (*parent*)**:
  Svaki čvor ima jednog roditelja (osim *root* čvora)
- **Dijete (*child*)**:
  Svaki čvor može imati nijedno, jedno ili više djece
- **Braća (*siblings*)**:
  Čvorovi koji imaju istog roditelja nazivaju se braća
- **Preci (*ancestors*)**:
  Roditelj čvora, roditelj roditelja,... se nazivaju preci
- **Potomci (*descendants*)**:
  Dijete čvora, dijete djeteta,... se nazivaju potomci

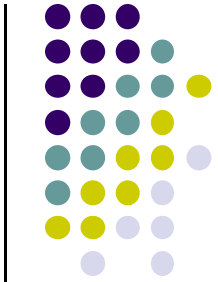
- **XPath** nam pokazuje put, odnosno relacije, između čvorova.

XPath izraz



Izraz	Opis
Ime_čvora (Nodename)	Selektira svu djecu čvora
/	Selektira sve unutar root čvora
//	Selektira čvorove od čvora na kojem se trenutno nalazimo
.	Selektira čvor na kojem se nalazi
..	Selektira roditelj čvora
@	Selektira atribute od čvora na kojem se trenutno nalazimo
*	Koristi se za nepoznate dijelove XML
|	Selektiramo više čvorova

XPath izraz



- drugagodina – dohvaća sve child čvorove od čvora drugagodina
- /drugagodina – dohvaća čvor drugagodina
- drugagodina/student – dohvaća sve čvorove student koji su child čvor od čvora drugagodina
- //student – dohvaća sve čvorove student bez obzira gdje su pozicionirani u dokumentu
- drugagodina//student – dohvaća sve čvorove student kojim je predak čvor drugagodina (ne mora biti prvi predak)
- //@date – dohvaća sve atribute koji se zovu date

XML dokument:

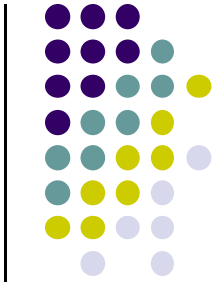
```
<drugagodina>
<student>
    <id date="12/11/2002">1</id>
    <ime>Vedran</ime>
    <prezime>Alebić</prezime>
    <prosjek>4.0</prosjek>
</student>
<student>
    <id>2</id>
    <ime>Marijan</ime>
    <prezime>Babić</prezime>
</student>
</drugagodina>
```

XPath

dohvaća vrijednosti svih child čvorova od root čvora drugagodina
<xsl:value-of select="drugagodina" />
ili ovakva sintaksa
<xsl:value-of select="/" />
dohvaća vrijednosti svih child čvorova od čvora /drugagodina/student
<xsl:value-of select="/drugagodina/student" />
dohvaća vrijednosti svih child čvorova od zadnjeg čvora /drugagodina/student
<xsl:value-of select="/drugagodina/student[last()]" />
dohvaća vrijednosti svih child čvorova od čvora /drugagodina/student čiji child čvor ime ima vrijednost Toni
<xsl:value-of select="/drugagodina/student[ime='Toni']" />
dohvaća vrijednosti svih child čvorova od čvora na kojem je trenutno pozicioniran
<xsl:value-of select="//*" />
dohvaća trenutni čvor na kojem je trenutno pozicioniran
<xsl:value-of select="." />
dohvaća čvor parent od čvora na kojem je trenutno pozicioniran
<xsl:value-of select=".." />
dohvaća sve atribute koji se zovu date od čvora na kojem je trenutno pozicioniran
<xsl:value-of select="//@date" />
dohvaća sve atribute bilo kako da se zovu od čvora na kojem je trenutno pozicioniran
<xsl:value-of select="//@*" />

primjer_xpath.xsl

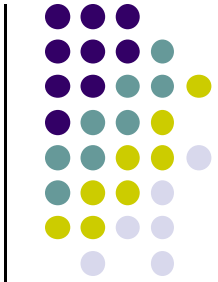
XPath predikati



- XPath predikati su izrazi u kojima se koriste XPath oznake čvorova, operatori (=,!=,<,>,>=,<=, +, -, *, div, and [ime= 'Marko' and prezime= ‘Markov'], or, mod, |) i funkcije.
- Predikati se koriste za dohvaćanje specifičnog čvora ili čvora koji sadrži specifičnu vrijednost.
- Uvijek se navode u uglatim zagradama.
- Npr. [last()] – dohvaćanje zadnjeg čvora, [ime='Toni'] – dohvaćanje čvora koji u čvoru ime sadrži vrijednost Toni,

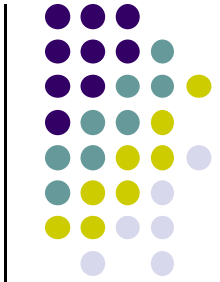
  [@date] – dohvaćanje čvora koji sadrži atribut date,…

XPath funkcije



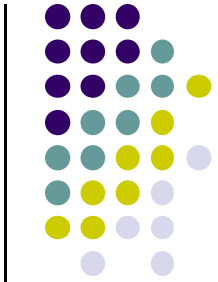
- Osnovne funkcije koje uvijek trebaju biti implementirane mogu se pronaći na (http://www.w3.org/TR/xpath#corelib).
- Neke od tih funkcija su:
  - *number* **last**() – funkcija vraća broj čvorova promatranog XPath elementa.
  - *number* **position**() – funkcija vraća poziciju promatranog XPath elementa.
  - *number* **count**(*node-set*) – funkcija vraća broja čvorova u čvoru definiranom u argumentu node-set.
  - *node-set* **id**(*object*) – funkcija selektira čvor prema jedinstvenom ID-u. To je vrijednost atributa koji je u DTD definira s tipom ID. Dva elementa ne smiju imati isti ID. Ukoliko imaju drugi element s tretira kao da nema jedinstveni ID.

XPath funkcije



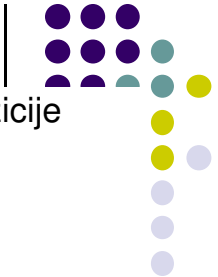
- *number* **string-length**(*string*?) – funkcija vraća broj karaktera u stringu.
- *string* **normalize-space**(*string*?) – funkcija vraća string bez razmaka na početku i kraju stringa.
- *string* **translate**(*string*, *string*, *string*) - funkcija vraća prvi string u kojem su zamijenjeni karakteri iz drugog stringa s onima iz trećeg stringa. Npr. translate("bar","abc","ABC") vraća BAr.
- *boolean* **boolean**(*object*) – funkcija pretvara vrijednost argumenta u *boolean* vrijednost ovisno o vrijednost argumenta. Npr. čvor koji je prazan pretvoriti će se u false.
- *boolean* **not**(*boolean*) - funkcija vraća true ako je argument false i obratno.
- *boolean* **true**() - funkcija vraća true.
- *boolean* **false**() - funkcija vraća false.

XPath funkcije



- *boolean* **lang**(*string*) – funkcija vraća true ako je jezik korišten u promatranom čvoru isti kao i argument funkcije.
- *number* **number**(*object*?) – funkcija konvertira objekt u broj.
- *number* **sum**(*node-set*) – funkcija vraća sumu vrijednosti čvorova navedenih u argumentu.
- *number* **floor**(*number*) – funkcija vraća najveći cijeli broj ne veći od argumenta.
- *number* **ceiling**(*number*) – funkcija vraća najmanji cijeli broj ne veći od argumenta.
- *number* **round**(*number*) – funkcija vraća cijeli broj najbliži argumentu.

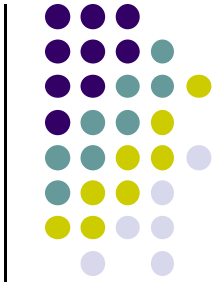
primjer_xpath_funkcije.xsl



```
dohvaća vrijednosti child čvora ime prezime od čvora drugagodina/student iznad 25 pozicije i ispod 4 pozicije
<xsl:for-each select="drugagodina/student">
<xsl:if test="position() > 25 or position() &lt; 4">
<xsl:value-of select="ime" /> 
<xsl:value-of select="prezime" /><br />
</xsl:if>
</xsl:for-each>
sumira vrijednosti svih child čvorova prosjek
Ukupni prosjek = <xsl:value-of select="sum(//prosjek)" /><br />
<b>srednja vrijednosti svih child čvorova prosjek</b><br />
Srednji prosjek = <xsl:value-of select="sum(//prosjek) div count(//prosjek)" /><br />
lokalno ime čvora /drugagodina/student/ime
<xsl:value-of select="local-name(/drugagodina/student/ime)" /><br />
ime čvora /drugagodina/student/ime
<xsl:value-of select="name(/drugagodina/student/ime)" /><br />
string vrijednost atributa date elementa id prvog čvora drugagodina/student
<xsl:value-of select="string(/drugagodina/student/id/@date)" /><br />
dohvaća vrijednosti čvorova ime i prezime ako ime ili prezime počima s M
<xsl:for-each select="/drugagodina/student/ime[starts-with(.,'M')] | /drugagodina/student/prezime[starts-
      with(.,'M')]">
<xsl:value-of select="../ime" /> 
<xsl:value-of select="../prezime" /><br />
</xsl:for-each>
dohvaća vrijednosti čvorova ime i prezime ako je prezime dulje od 6 slova
<xsl:for-each select="drugagodina/student">
<xsl:if test="string-length(./prezime) > 6">
<xsl:value-of select="ime" /> 
<xsl:value-of select="prezime" /><br />
</xsl:if>
</xsl:for-each>
```

Specijalni karakter <

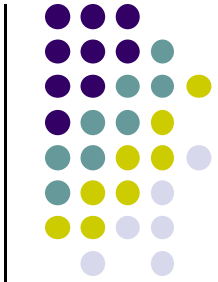
XQuery, XLink, XPointer



Uz XPath su usko vezana tri jezika:

- XQuery (*XML Query Language*)
  - jezik za izvršavanje upita nad XML dokumentima
  - traži elemente unutar XML dokumenata i čita njihove atribute
  - temeljen je na Xpath izrazima i dio je W3C standarda
- XLink (*Xml Linking Language*)
  - jezik za kreiranje hiperlinkova u XML dokumentima
  - sličan je HTML linkovima, ali daje još puno više mogućnosti
  - podržava jednostavne linkove i proširene linkove (linkanje više izvora zajedno)
- XPointer (*XML Pointer Language*)
  - jezik koji omogućava kreiranje hiperlinkova koji pokazuju na točno određene dijelove XML dokumenta
  - pri tome koristi XPath izraze

XSLT

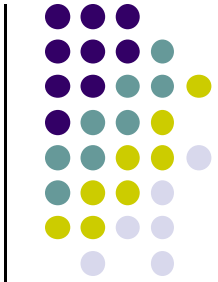


- Prilikom transformacije originalnog stabla u rezultirajuće stablo XSLT koristi XPath za definiranje dijelova originalnog stabla na koje se treba primijeniti određeni predložak transformacije.
- <xsl:stylesheet> i <xsl:transform> su XSLT elementi koji definiraju root element *stylesheeta*.
- Stylesheet se predstavlja xsl:stylesheet elementom, a xsl:transform element je ustvari sinonim za xsl:stylesheet element koji se također može koristiti.
- Element xsl:stylesheet mora imati atribut *version* čija vrijednost kaže XSL *stylesheet* procesoru koju verziju XSLT zahtjeva taj *stylesheet*.
- Primjer:

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

Imenski prostor je opcionalan atribut

XSLT elementi



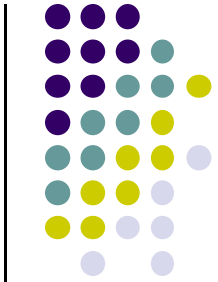
- XSL *stylesheet* se sastoji tj. sadrži jedan ili više seta pravila koje zovemo predlošci (*template*).:

  <xsl:template match="/">... </xsl:template>

- Predlošci uključuju različite XSL elemente. Osnovni element definira predložak:
  - <template> element sadrži pravila koja se primjenjuju kad se pronađe odgovarajući čvor. <xsl:template> *tag* definira početak *template-a*. Atribut *match* povezuje predložak sa XML elementom. U ovom primjeru "/" je to root XML element.

<xsl:template match="/">
<html><body>
</body></html>
</xsl:template>

XSLT elementi



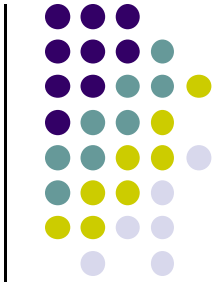
- <value-of> element se koristi za selektiranje vrijednosti XML elementa. Atribut *select* je XPath izraz (obavezan) kojim se pristupa elementu (elementima) u XML-u na kojeg se XSLT element primjenjuje.

```
<td><xsl:value-of select="ime" /></td>
```

- <for-each> element dozvoljava primjenu petlji u XSLT-u. Atribut *select* je XPath izraz (obavezan) kojim se pristupa elementu (elementima) u XML-u na kojeg se XSLT element primjenjuje.

```
<xsl:for-each select="drugagodina/student">
<tr>
   <td><xsl:value-of select="ime" /></td>
   <td><xsl:value-of select="prezime" /></td>
</tr>
</xsl:for-each>
```

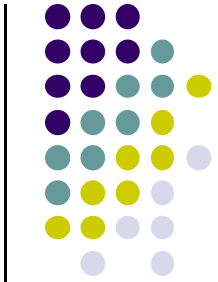
XSLT elementi



- <sort> element se koristi za sortiranje vrijednosti XML elementa. Atribut *select* je XPath izraz (obavezan) kojim se pristupa elementu/ima u XML-u na kojeg se XSLT element primjenjuje.

```
<xsl:for-each select="drugagodina/student">
<xsl:sort select="prezime"/>
<tr>
   <td><xsl:value-of select="ime" /></td>
   <td><xsl:value-of select="prezime" /></td>
</tr>
</xsl:for-each>
```

XSLT elementi

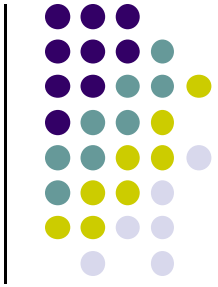


- <if> element sadrži *template* koji se primjenjuje samo ako je zadovoljen određeni uvjet. Uvjet se navodi u atributu *test* koji je obavezan.

```
<xsl:if test="string-length(./prezime)">
<tr>
    <td><xsl:value-of select="ime" /></td>
    <td><xsl:value-of select="prezime" /></td>
</tr>
</xsl:if>
```

Ovaj dio će se primijeniti samo ako je zadovoljen uvjet u xsl:if elementu

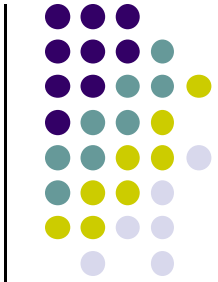
XSLT elementi



- <choose> element se koristi zajedno s <xsl:when> i <xsl:otherwise> za iskazivanje višestrukih uvjetnih testova.

```
<xsl:for-each select="drugagodina/student">
<xsl:choose>
<xsl:when test="number(./prosjek) >= 4.0">
<div style="background-color:teal;color:white;padding:4px"><span style="font-
    weight:bold;color:white">
<xsl:value-of select="ime" /> <xsl:value-of select="prezime" /><br /></span></div>
</xsl:when>
<xsl:when test="number(./prosjek) >= 3.0">
 <div style="background-color:red;color:white;padding:4px"><span style="font-
    weight:bold;color:white">
<xsl:value-of select="ime" /> <xsl:value-of select="prezime" /><br /></span></div>
</xsl:when>
<xsl:otherwise>
<div style="background-color:yellow;color:white;padding:4px"><span style="font-
    weight:bold;color:white">
<xsl:value-of select="ime" /> <xsl:value-of select="prezime" /><br /></span></div>
</xsl:otherwise>
</xsl:choose>
</xsl:for-each>
```

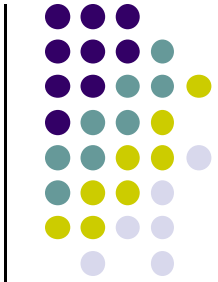
XSLT funkcije



- XSLT tj. XPath (XSLT i XPath koriste istu biblioteku funkcija) ima preko 100 ugrađenih funkcija za rad sa stringovima, brojevima, datumom, i sl.
- Imenski prostor XSLT funkcija je URI: http://www.w3.org/2005/02/xpath-functions
- Prefiks funkcija u imenskom prostoru je fn:.
- Primjer:
  - fn:position() – vraća poziciju čvora koji se trenutno obrađuje
- Imenski prostor se ne treba navoditi jer je to defaultni prostor.

```
<xsl:if test="position() mod 2 > 0">
<tr>
   <td><xsl:value-of select="ime" /></td>
   <td><xsl:value-of select="prezime" /></td>
</tr>
</xsl:if>
```

XSL primjeri



- prvi.xsl
- drugi.xsl
- treci.xsl