XML Vārdu Telpas
un
XML Schema

Previously ...

1. Iezīmēšanas valoda ir vārdu un simbolu kopa, lai aprakstītu dokumenta daļas

2. XML ir standarts vispārēja nolūka datu transportēšanai un uzglabāšanai

3. Parsēšana ir process, kurā XML sintakse ir pārbaudīta un teksts tiek sadalīts atsevišķās, pārvaldāmās daļās

4. XML paredzēts transportēt un uzglabāt datus

Previously ...

5. XML dokuments sastāv no deklarācijas, elementiem, atribūtiem, komentāriem un atsaucēm

6. XML tagi veido dokumenta iezīmēšanu

7. XML dokumentam ir hierarhiskā struktūra, kas sākas “saknes” elementā un atzarojās “lapās”

Praktiskais uzdevums

7. UZDEVUMS

Plāns

- XML vārdu telpas
- XML dokumentu shēma un validācija
- Ieskats DTD dokumentu tipu definīcijā
- XML Schema
- XML Schema funkcijas un labumi
- XML Schema dokuments
- Vienkārša <element> deklarēšana
- Sarežģīta <element> deklarēšana
- Piemēri

1. XML vārdu telpas

- XML Namespaces (angl.)
- W3C rekomendācija

  http://www.w3.org/TR/REC-xml-names/
- XML vārdu telpas ļauj izvairīties no dokumenta elementu vārdu konfliktiem
- Tiek izmantotas, lai atšķirtu elementus un atribūtus no dažādiem XML dokumentu veidiem (vārdnīcām), to apvienojot vienā dokumentā

2. Vārdu konflikts

- XML elementu vārdus definē izstrādātājs
- Vārdu konflikts var rasties apvienojot dažādus XML dokumentus
- Piemērā XML parsētājam nav starpības starp **<title>** dokumenta elementiem

```
<person>
    <name>
        <title>Sir</title>
        <first>John</first>
        <middle>Fitzgerald Johansen</middle>
        <last>Doe</last>
    </name>
    <position>Vice President of Marketing</position>
    <résumé>
        <html>
            <head>
                <title>Resume of John Doe</title>
            </head>
            <body>
                <h1>John Doe</h1>
                <p>John‘s a great guy, you know?</p>
            </body>
        </html>
    </résumé>
</person>
```

3. XML elementu prefikss (1)

XML elementu vārdu konflikta atrisināšanai katram elementam jābūt unikāli identificējamam

- Jāizmanto prefiksu elementa vārdā

<prefix:elementName/>

3. XML elementu prefikss (2)

```
<pers:person>
    <pers:name>
        <pers:title>Sir</pers:title>
        <pers:first>John</pers:first>
        <pers:middle>Fitzgerald Johansen</pers:middle>
        <pers:last>Doe</pers:last>
    </pers:name>
    <pers:position>Vice President of Marketing</pers:position>
    <pers:résumé>
        <xhtml:html>
            <xhtml:head>
                <xhtml:title>Resume of John Doe</xhtml:title>
            </xhtml:head>
            <xhtml:body>
                <xhtml:h1>John Doe</xhtml:h1>
                <xhtml:p>John‘s a great guy, you know?</xhtml:p>
            </xhtml:body>
        </xhtml:html>
    </pers:résumé>
</pers:person>
```

4. XML vārdu telpas (1)

- Lai izmantotu XML prefiksu, tam jādefinē vārdu telpa
- XML Namespaces rekomendācijas no W3C nosaka standartu vārda telpu deklarēšanai un elementa vai atribūta vārda telpas identificēšanai
- XML vārdu telpu izmantošanai dokumentā elementiem jāizmanto **kvalificētos** vārdus (QName)

- QName sastāv no divām daļām
  - Elementa lokāla daļa (<prefix:**elementName**>)
  - Vārda telpas prefikss (<**prefix**:elementName>)

4. XML vārdu telpas (2)

- XML vārdu telpa ir definēta izmantojot **xmlns** atribūtu elementa sākuma tagā
- XML vārdu telpas definēšanas sintakse

  xmlns:prefix="URI"

- Lai deklarētu vārdu telpu “**http://www.wiley.com/pers”** un saistītu **<person>** elementu ar to, ir jādefinē:

  <pers:person

  xmlns:pers=”http://www.wiley.com/pers”/>

4. XML vārdu telpas (3)

- Prefiksu var izmantot jebkurš no **<pers:person>** elementa pectečiem, lai indicētu to piederību **http://www.wiley.com/pers** vārdu telpai

```
<pers:person xmlns:pers="http://www.wiley.com/pers">
    <pers:name>
        <pers:title>Sir</pers:title>
    </pers:name>
</pers:person>
```

5. XML vārdu telpa pēc noklusēšanas (1)

- Vārdu telpā pēc noklusēšanas nav jānorāda prefiksu visiem elementiem, kuri to izmanto

```
<person xmlns="http://www.wiley.com/pers">
    <name>
        <title>Sir</title>
    </name>
</person>
```

5. XML vārdu telpa pēc noklusēšanas (2)

- Elementam var definēt vairāk par vienu vārdu telpām, bet tikai viena varbūt pēc noklusēšanas

```
<person xmlns="http://www.wiley.com/pers"
        xmlns:xhtml="http://www.w3.org/1999/xhtml">
    <name />
    <xhtml:p>This is XHTML</xhtml:p>
</person>
```

5. XML vārdu telpa pēc noklusēšanas (3)

- XML vārdu telpu pēc noklusēšanas var definēt elementam un tā pēctečiem
- Piemērā vārdu telpa pēc noklusēšanas ir pārdefinēta **<p>** elementā uz **http://www.w3.org/1999/xhtml**.

```
<person xmlns="http://www.wiley.com/pers">
    <name />
    <p xmlns="http://www.w3.org/1999/xhtml">
        This is XHTML
    </p>
</person>
```

6. Atšķirīgas notācijas – vienāds rezultāts

```
<pers:person xmlns:pers="http://www.wiley.com/pers"
xmlns:xhtml="http://www.w3.org/1999/xhtml">
<pers:name/>
<xhtml:p>This is XHTML</xhtml:p>
</pers:person>
```

```
<person xmlns="http://www.wiley.com/pers"
xmlns:xhtml="http://www.w3.org/1999/xhtml">
<name/>
<xhtml:p>This is XHTML</xhtml:p>
</person>
```

```
<person xmlns="http://www.wiley.com/pers">
<name/>
<p xmlns="http://www.w3.org/1999/xhtml">This is XHTML</p>
</person>
```

7. XML vārdu telpu URI (1)

- Uniform Resource Identifier (angl.)
- Rakstzīmju virkne resursu identificēša
  - URL (angl., Uniform Resource Locator)


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Response xmlns="http://tempuri/XMLSchemas/v1"
          xmlns:ns2="http://ivis.eps.gov.lv/XMLSchemas/100017/fidavista/v1-2">
    <AccountStatementResponse>
        <ns2:FIDAVISTA>
            <ns2:Header>
                <ns2:Timestamp>20130312214257000</ns2:Timestamp>
                <ns2:From>PARXLV22</ns2:From>
            </ns2:Header>
            <ns2:Statement>
                <ns2:Period>
                    <ns2:StartDate>2013-01-01</ns2:StartDate>
                    <ns2:EndDate>2013-02-01</ns2:EndDate>
                    <ns2:PrepDate>2013-09-27</ns2:PrepDate>
                </ns2:Period>
                <ns2:AccountSet>
                    <ns2:IBAN>LV11PARX000123456789</ns2:IBAN>
```

7. XML vārdu telpu URI (2)

- URN (angl., Universal Resource Name).

```
<?xml version="1.0" encoding="UTF-8"?>
<CBICF:CB.ICF.BlkCdtTrf
    xmlns:CBICF="urn:CBICF:xsd:CB.ICF.BlkCdtTrf"

    <CBICF:FIToFICstmrCdtTrf
        xmlns="urn:iso:std:iso:20022:tech:xsd:sct:pacs.008.001.02.EKS">
        <GrpHdr>
            <MsgId>Payments001</MsgId>
            <CreDtTm>2010-06-30T08:31:00</CreDtTm>
            <NbOfTxs>1</NbOfTxs>
            <TtlIntrBkSttlmAmt Ccy="EUR">546.34</TtlIntrBkSttlmAmt>
            <IntrBkSttlmDt>2010-06-30</IntrBkSttlmDt>
            <SttlmInf>
```

7. XML vārdu telpu URI (3)

- W3C rekomendācija izmanto vārdu telpas **nosaukuma** terminu
- Bieži vārdu telpas URI definēšanai tiek izmantots URL HTTP shēmā (piem., http://www.w3.org/1999/xhtml), neskatoties uz XSD formālas saistības trūkumu ar HTTP protokolu
  - Izvēlēta URI apraksta resursu, kas atrodas XML vārdnīcas autora kontrolē. Piemēram, autora tīmekļa servera URL

Piemērs: ISO20022 maksājumu iniciēšana

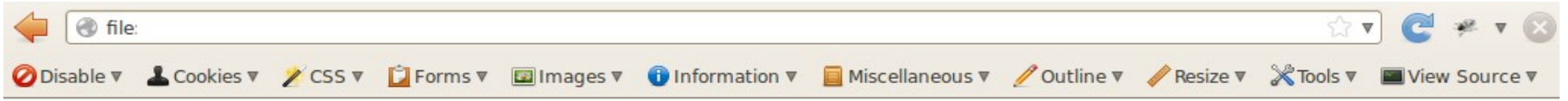
```
<?xml version="1.0" encoding="UTF-8"?>
<Document xmlns="urn:iso:std:iso:20022:tech:xsd:pain.001.001.03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="urn:iso:std:iso:20022:tech:xsd:pain.001.001.03
    <CstmrCdtTrfInitn>
        <GrpHdr>
            <MsgId>87fbf20111125/1</MsgId>
            <CreDtTm>2011-11-25T11:16:58.696</CreDtTm>
            <NbOfTxs>3</NbOfTxs>
            <CtrlSum>2500</CtrlSum>
            <InitgPty>
                <Nm>AS XML</Nm>
            </InitgPty>
        </GrpHdr>
        <PmtInf>
            <PmtInfId>PMTID001</PmtInfId>
            <PmtMtd>TRF</PmtMtd>
            <BtchBookg>true</BtchBookg>
            <NbOfTxs>3</NbOfTxs>
            <PmtTpInf>
                <SvcLvl>
                    <Cd>SEPA</Cd>
                </SvcLvl>
```

Piemērs: XHTML 1.1 + MathML 2.0 + SVG 1.1

```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:svg="http://www.w3.org/2000/svg">
    <body>
        <h2>MathML</h2>
        <p>
            <math xmlns="http://www.w3.org/1998/Math/MathML">
                <mfrac>
                    <mi>a</mi>
                    <mi>b</mi>
                </mfrac>
            </math>
        </p>

        <h2>SVG</h2>
        <p>
            <svg:svg width="50px" height="50px">
                <svg:circle cx="25px" cy="25px" r="20px" fill="green" />
            </svg:svg>
        </p>
    </body>
</html>
```

Piemērs: XHTML 1.1 + MathML 2.0 + SVG 1.1



MathML

$$\frac{a}{b}$$

SVG

Praktiskais uzdevums

5.UZDEVUMS

XML Dokumentu Shēma

DTD

XML Schema

1. XML shēma

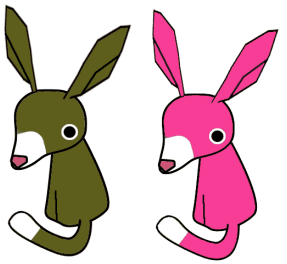
- XML dokumenta kvalitātes kontroles rīks
- Ļauj pārliecināties, ka XML dokumenta instance atbilst sintakses un struktūras noteikumu kopai
- XML shēmu var uztvert par programmu parsētājam kā lasīt dokumentu

2. XML shēmas valodas

- DTD (angl. Document Type Definition)
  - vecākā XML dokumenta shēmas valoda
  - speciālā ne-XML sintakse

- **W3C XML Schema**
  - **Pilnvērtīgs XML dokuments**

- RELAX NG
- Schematron



3. XML dokumenta shēmas validācija

- Derīgs XML dokuments ir korekti noformēts XML dokuments, kurš arī atbilst XML dokumenta shēmas (DTD, XML Schema) noteikumiem

- **Validācijas process**
  1. Parsētājs ielāde shēmas noteikumus un deklarācijas
  2. Tiek konstruēts speciāla tipa validācijas parsētājs
  3. Parsētājs pieņem XML dokumenta instanci un ģenerē tas validācijas atskaiti

4. Ieskats DTD dokumentu tipu definīcijā

- (angl.) Document Type Definition

- Nosaka
  - atļautus XML dokumenta blokus
  - dokumenta struktūru, izmantojot sarakstu ar atļautiem dokumenta elementiem un atribūtiem

- DTD var specificēt XML dokumenta iekšā (internal) vai atsevišķajā failā (external)

4. Ieskats DTD dokumentu tipu definīcijā

- Iekšēja DTD piemērs
- <!DOCTYPE root-element[element-declaration]>

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

5. XML Schema (1)

- “XML Schema” ir W3C tehnoloģija XML dokumenta shēmas aprakstam
- Viens no svarīgiem XML tehnoloģiju pamatiem
- Līdzīgi DTD XML Schema ļauj aprakstīt XML dokumenta struktūru
- XML Schema ir DTD alternatīva un pēctecis balstīts uz XML

5. XML Schema (2)

- XML Schema valodu parasti sauc arī par XSD (angl. XML Schema Definition)

- XSD ir XML valoda
  - izmanto XML vārdu telpas
- XSD ir specifiskāka par DTD
  - izmanto datu tipus

6. XML Schema pamata funkcijas

- Definē kādi elementi var parādīties dokumentā
- Definē kādi elementu atribūti var parādīties dokumentā
- Definē kādu elementus var iekļaut citos elementos
- Definē elementu kārtību
- Nosaka iespējamo elementu skaitu
- Definē vai elements ir tukšs vai var iekļaut tekstu
- Definē datu tipus elementiem un atribūtiem
- Definē vērtības pēc noklusēšanas elementiem un atribūtiem

7. XML Schema labumi (1)

- XML Schema tiek izveidota izmantojot XML pamata sintaksi (atšķirībā no DTD speciālās sintakses)
- XML Schema pilnīgi atbalsta W3C Namespace rekomendācijas

7. XML Schema labumi (2)

- Atļauj vienkārša veidā veidot sarežģītus un atkārtoti lietojamus satura modeļus (piem., ISO20022 ziņojumu standarts)
- Atbalsta programmēšanas valodu koncepciju modelēšanu (piem. mantošanas saistības no objekt-orientētās programmēšanas)

7. XML Schema labumi (3)

- ISO20022 ziņojumu formāta atkārtotā lietošana

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:lv8="urn:iso:std:iso:20022:tech:xsd:sct:pacs.008.001.02.EKS"
           xmlns:lv4="urn:iso:std:iso:20022:tech:xsd:sct:pacs.004.001.02.EKS"
           xmlns:NHICF="urn:NHICF:xsd:NH.ICF.BlkCdtTrf"
           targetNamespace="urn:NHICF:xsd:NH.ICF.BlkCdtTrf"
           elementFormDefault="qualified">

    <xs:import namespace="urn:iso:std:iso:20022:tech:xsd:sct:pacs.008.001.02.EKS"
        schemaLocation="pacs.008.001.02.EKS.xsd"/>
    <xs:import namespace="urn:iso:std:iso:20022:tech:xsd:sct:pacs.004.001.02.EKS"
        schemaLocation="pacs.004.001.02.EKS.xsd"/>
    <xs:element name="NH.ICF.BlkCdtTrf">
```

8. XML Schema vs. DTD (1)

XML Schema mehānism ir jaudīgāks

- var atkārtoti lietot shēmas
- var definēt speciālus datu tipus, atvasinot no iebūvētiem
- var norādīt vairākas shēmas vienā dokumentā

8. XML Schema vs. DTD (2)

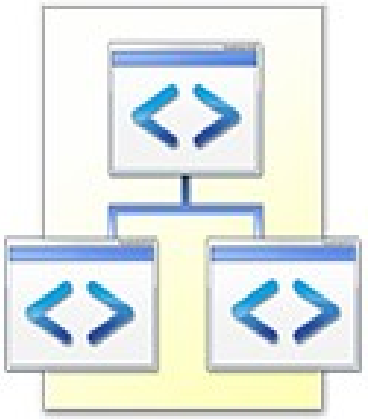
- XML Schema atbalsta datu tipus
  - Vieglāk aprakstīt atļauto dokumentu saturu
  - Vieglāk pārbaudīt datu pareizumu
  - Vieglāk apstrādāt datus no datu bāzēs
  - Vieglāk aprakstīt datu ierobežojumus un šablonus
  - Vieglāk konvertēt datus
- XML Schema tiek rakstīta XML valodā
  - Nav nepieciešamības mācīties jauno valodu
  - Var izmantot XML redaktoru un parsētāju

9. XML Schema standarts

- XML Schema standarts ir sadalīts trīs daļās:
  - XML Schema pamata koncepcijas
    - www.w3.org/TR/xmlschema-0/
  - Struktūras, izmantotas XML Schemas
    - www.w3.org/TR/xmlschema-1/
  - XML Schema data tipu apraksts
    - www.w3.org/TR/xmlschema-2/

10. XML Schema dokuments (1)

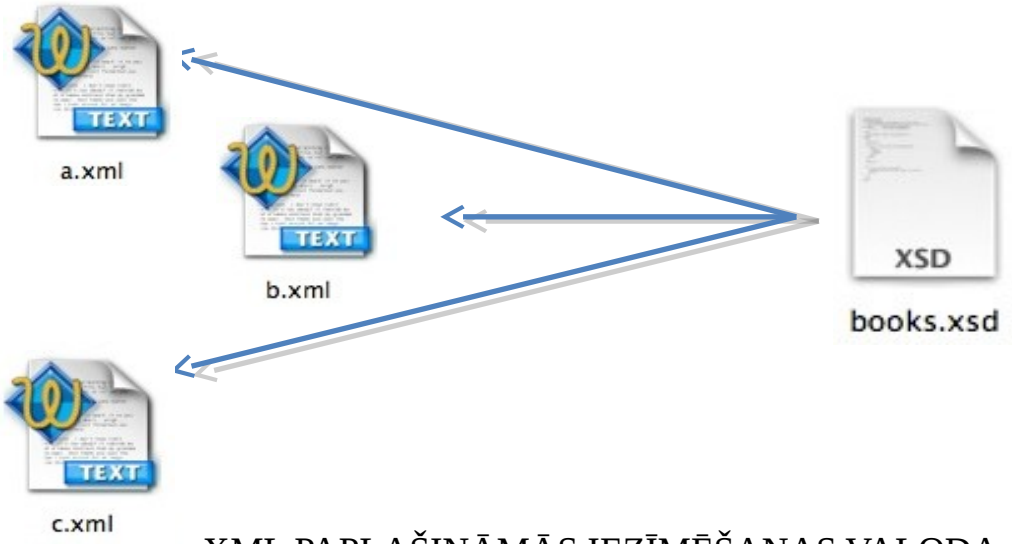
- XML Schema tiek saglabāta atsevišķajā XML dokumentā
- XML dokuments kurš ir uzbūvēts atbilstoši noteiktai XML Schema shēmas vārdnīcai tiek saukts par XML Schema instances dokumentu
- XML Schema faili parasti izmanto **.xsd** paplašinājumu
- “schema” sāknes elements



10. XML Schema dokuments (2)

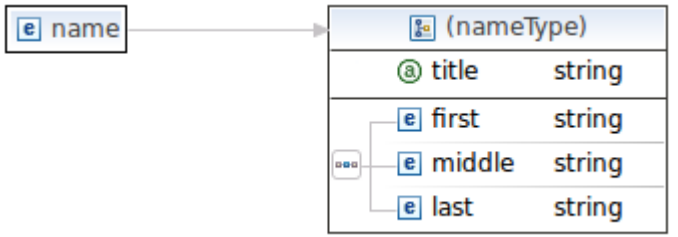
- XML dokumenti satur norādi uz XML Schema dokumentu, kas nosaka to vārdnīcu

```
<note xmlns="http://www.nh.lv/books"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.nh.lv/books note.xsd">
```



10. XML Schema dokuments (3)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.example.org/NewXMLSchema"
    xmlns:tns="http://www.example.org/NewXMLSchema"
    elementFormDefault="qualified">
    <element name="name">
        <complexType>
            <sequence>
                <element name="first" type="string" />
                <element name="middle" type="string" />
                <element name="last" type="string" />
            </sequence>
            <attribute name="title" type="string" />
        </complexType>
    </element>
</schema>
```



11. <schema> elements (1)

- <schema> elements ir XML Schema dokumenta sāknes elements

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

11. <schema> elements (2)

- <schema> elementā definējami papildus atribūti

```
<?xml version="1.0"?>
<xs:schema
xmlns:xs= “http://www.w3.org/2001/XMLSchema”
targetNamespace= “http://www.w3schools.com”
xmlns= “http://www.w3schools.com”
elementFormDefault= "qualified">
...
</xs:schema>
```

12. XML Schema vārdu telpa (1)

- Var izmantot jebkuru no variantiem

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

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

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

12. XML Schema vārdu telpa (2)

- Nosaka, ka elementi un datu tipi izmantoti XML shēmas definīcijā nāk no XMLSchema vārdnīcas (http://www.w3.org/2001/XMLSchema)

- Nosaka, ka elementi un datu tipi no “http://www.w3.org/2001/XMLSchema” jāraksta ar “xs:” vai “xsd:” prefiksu

13. Mērķa vārdu telpa (targetNamespace)

- Pirmatnēja XML Schema nozīme ir XML vārdnīcas deklarēšana vārdu telpai, specificētai “targetNamespace” atribūtā

```
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.example.org/NewXMLSchema"
  xmlns:tns="http://www.example.org/NewXMLSchema">
```

14. XML Schema datu tipi

- Atļauj vairāk specifikas par DTD
- Var norādīt datumus, numurus, diapazonus, unc

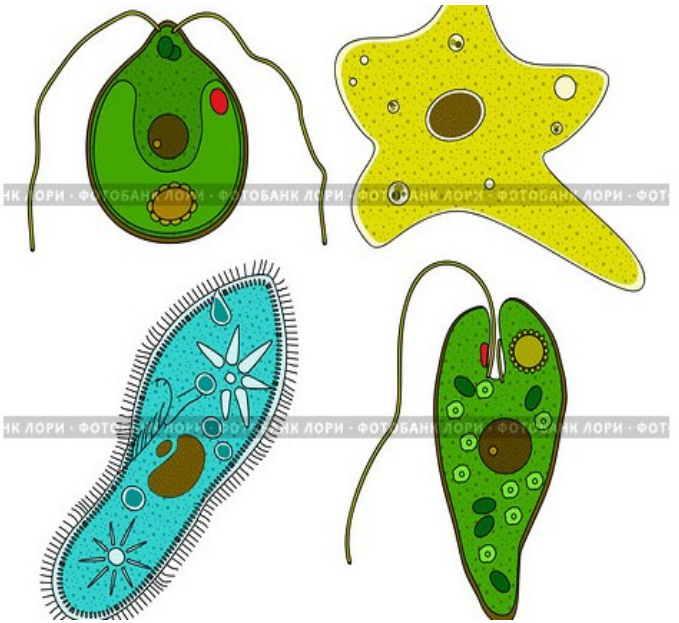
Var iedalīt divās kategorijās:

Vienkāršie - pamatvērtības

Sarežģītie - apraksta sarežģītākas konstrukcijas

15. Vienkāršie datu tipi (1)

- Apraksta tekstu, ciparus, datumus
- Bieži sauktas par “primitīvajiem”
- Vienkāršo datu tipu elementiem nevar būt atribūtu
- Iebūvētie datu tipi XML Schema vārdu telpā



15. Vienkāršie datu tipi (2)

String

Boolean

Numbers

Integer

Decimal

Float

Double

Date/time

Time

TimeInstant

Duration

Date

Month

Year

...

16. Vienkāršā <element> deklarēšana (1)

<xs:element name="NAME" type="TYPE"/>

- NAME – elementa nosaukums
- TYPE – elementa datu tips

- XML Schema definē vairākus iebūvētus datu tipus
  - xs:string
  - xs:decimal
  - xs:integer
  - xs:boolean
  - xs:date
  - xs:time

```
<lastname>Jones</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

```
<xs:elementname = lastname " type="xs:string" />
<xs:elementname = age " type="xs:integer" />
<xs:elementname = dateborn " type="xs:date" />
```

16. Vienkāršā <element> deklarēšana (2)

```
<xsd:element name="firstname"
             type="xsd:string" />

<xsd:element name="ableToSwim"
             type="xsd:boolean" />

<xsd:element name="date"
             type="xsd:date" />
```

16. Vienkāršā <element> deklarēšana (3)

- Kardinalitāte (angl. cardinality) nosaka cik reizes specifisks elements var parādīties satura modelī

<element name=”first” type=”string”

minOccurs=”x” maxOccurs=”y”/>

- x : minimālais elementa parādīšanas skaits
- y : maksimālais elementa parādīšanas skaits

16. Vienkāršā <element> deklarēšana (4)

- Vienkāršiem elementiem XML shēmā var definēt vērtības pēc noklusēšanas vai nemainīgas vērtības
  - Vērtība pēc noklusēšanas tiek automatiski piešķirta elementam, kad netiek specificēta cita vērtība

  <xs:element name="color" type="xs:string" default="red"/>

- Nemainīgas vērtības tiek automātiski piešķirtas elementam. Elementa vērtību nevar izmainīt uz citu.

  <xs:elementname="color" type="xs:string" fixed="red"/>

17. Vienkāršais vs. sarežģītais

- Vienkāršais
  - **<grade>7</grade>**
  - nav citu iekļauto elementu

- Sarežģītais
  - **<students><student>Jack</student></students>**
  - ir iekļautie elementi

18. Sarežģīta elementa deklarēšana (1)

- Sarežģīts XML elements (angl. complex element) satur citus elementus un/vai atribūtus
- Ir četri sarežģīto elementu tipi:
  - Tukšie elementi
  - Elementi, kuri ietver tikai elementus
  - Elementi tikai ar tekstu
  - Elementi, kuri ietver gan tekstu gan citus elementus

- Katrs no tiem elementu tipiem var ietvert atribūtus

18. Sarežģīta element deklarēšana (2)

- Tukšie elementi
  - Sarežģīts XML elements, "product", kurš ir tukšs

    <product pid="1345"/>

- Sarežģīts XML elements "employee", kurš satur tikai citus elementus

  <employee>

  <firstname>John</firstname>

  <lastname>Smith</lastname>

  </employee>

18. Sarežģīta element deklarēšana (3)

- Sarežģīts XML elements "description", kurš satur gan tekstu gan citus elementus

<description> It happened on

<date lang="norwegian">03.03.99</date>

</description>

19. <complexType> deklarēšana (1)

- "employee" elementu tipa deklarēšana

```
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.example.org/employee"
    xmlns:tns="http://www.example.org/employee"
    elementFormDefault="qualified">

    <xs:element name="employee">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="firstname" type="xs:string" />
                <xs:element name="lastname" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

19. <complexType> deklarēšana (2)

- "employee" elementu tipa deklarēšana
  - Tikai “employee” elements var izmantot deklarēto sarežģītu tipu
  - Elementi "firstname" un "lastname" apvienoti zem viena <sequence> indikatora, kas nozīmē, ka dokumentā tiem jāparādās tādā pati kārtībā, kā deklarēti shēmā

20. <complexType> deklarēšana (3)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.example.org/employee"
    xmlns:tns="http://www.example.org/employee"
    elementFormDefault="qualified">

    <xs:element name="employee" type="tns:EmployeeType" />

    <xs:complexType name="EmployeeType">
        <xs:sequence>
            <xs:element name="firstname" type="xs:string" />
            <xs:element name="lastname" type="xs:string" />
        </xs:sequence>
    </xs:complexType>

    <xs:element name="department">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="employee"
                type="tns:EmployeeType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

21. <attribute> deklarēšana

- XML atribūta definēšanas sintakse:

  <xs:attributename="ANAME"
  type="ATYPE"/>

- ANAME – atribūta nosaukums
- ATYPE – atribūta datu tips

- Vienkāršajiem elementiem nevar būt atribūti

Praktiskais uzdevums

6.UZDEVUMS (TODO saraksts)

Papildus XSD iezīmes

- **Iebūvēto bāzes tipu pārveidošana**
  - Vērtību ierobežojumi
  - Diapazoni
  - Saraksti

```
<simpleType name="Degrees">
  <restriction base="string">
  <enumeration value="AA" />
  <enumeration value="AS" />
  </restriction>
</simpleType>
```

Papildus XSD iezīmes

- **Saraksts (angl. enumeration)**

```
<xsd:simpleType name="car">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Audi"/>
    <xsd:enumeration value="Golf"/>
    <xsd:enumeration value="BMW"/>
  </xsd:restriction>
</xsd:simpleType>
```

Papildus XSD iezīmes

- **Šablons ar REGEX**

```
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[a-z]"/>
  </xsd:restriction>
</xsd:simpleType>
```

Papildus XSD iezīmes

REGEX Examples

```
<xs:pattern value="[A-Z][A-Z][A-Z]"/>
<xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
<xs:pattern value="[xyz]"/>
<xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
<xs:pattern value="([a-z])*"/>
<xs:pattern value="male|female"/>
<xs:pattern value="[a-zA-Z0-9]{8}"/>
```

Papildus XSD iezīmes

Elementu kārtībā

- **sequence**: elements parādās shēmas kārtībā
- **all**: elementi parādās jebkura kārtība

```
<complexType>
  <all>
    <element name="A" type="anyType" />
    <element name="B" type="anyType" maxOccurs="unbounded" />
  </all>
</complexType>
```

Papildus XSD iezīmes

Elementu alternatīva

```
<xsd:element name="person">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element name="employee" type="employee"/>
      <xsd:element name="member" type="member"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
```

Tukšs elements ar atribūtu

- XML

```
<student id="A1" />
```

- Schema

```
<xsd:element name="student" type="student_type" />

<xsd:complexType name="student_type">
   <xsd:attribute name="id" type="xsd:ID"/>
</xsd:complexType>
```

XML shēmu imports

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:lv8="urn:iso:std:iso:20022:tech:xsd:sct:pacs.008.001.02.EKS"
    xmlns:lv4="urn:iso:std:iso:20022:tech:xsd:sct:pacs.004.001.02.EKS"
    xmlns:CBINT="urn:CBINT:xsd:CB.INT.SingleTxInf"
    targetNamespace="urn:CBINT:xsd:CB.INT.SingleTxInf"
    elementFormDefault="qualified">

    <xs:import namespace="urn:iso:std:iso:20022:tech:xsd:sct:pacs.008.001.02.EKS"
        schemaLocation="pacs.008.001.02.EKS.xsd" />
    <xs:import namespace="urn:iso:std:iso:20022:tech:xsd:sct:pacs.004.001.02.EKS"
        schemaLocation="pacs.004.001.02.EKS.xsd"/>

    <xs:element name="CB.INT.SingleTxInf">
        <xs:complexType>
            <xs:sequence>
             <xs:choice>
                <xs:element name="CdtTrfTxInf" type="lv8:CreditTransferTransactionInformation11" />
                <xs:element name="TxInf" type="lv4:SCLSCTPaymentTransactionInformation27"/>
                </xs:choice>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

Praktiskais uzdevums

6.UZDEVUMS
(PAYMENT + TODO ar PAYMENT)

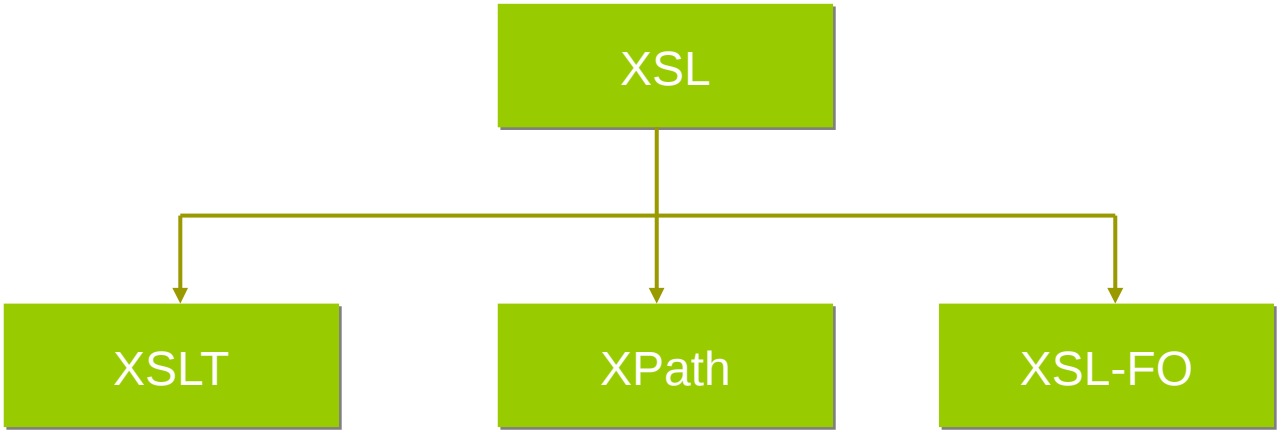
XPATH
un
XSLT

Ievads XSL

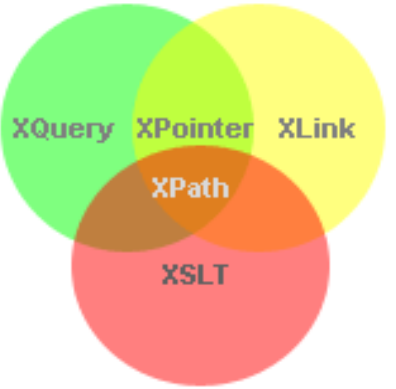
- (angl.) Extensible Stylesheet Language
- W3C XML standarts
- XML balstīta valoda stila lapu izveidošanai
- apraksta XML dokumentu formatēšanu vai transformēšanu

Ievads XSL

- **XSL sastāvs**
  - XSL Transformations (XSLT)
    - XML valoda XML dokumentu transformēšanai
  - XML Path language (XPath)
    - Specializēta valoda navigācijai XML dokumentā
  - XSL Formatting Object (XSL-FO)
    - XML valoda specializētai XML formatēšanai



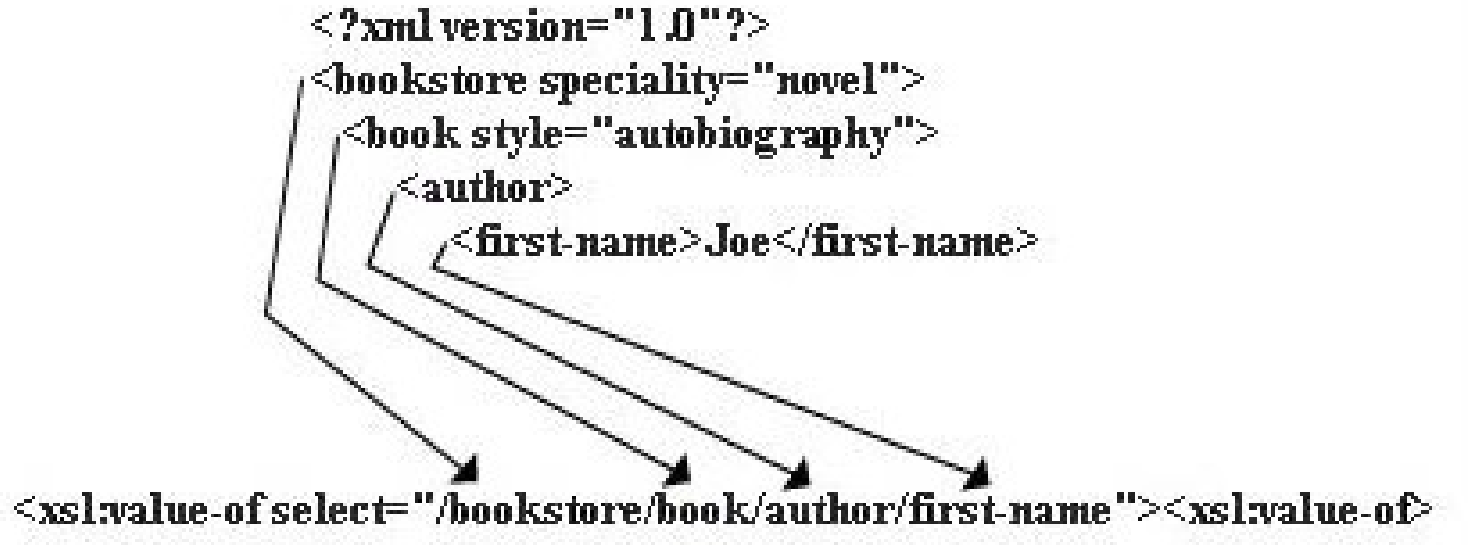
XPath valoda



- W3C recomendācija
- Informācijas atrašanai XML dokumentā
- Sintakse XML dokumenta daļu definēšanai
- Nodrošina standarto funkciju bibliotēku
- Izmanto speciālas izteiksmes navigācijai XML dokumentā
  - līdzīgi izteiksmēm izmantotām darbā ar datoru faila sistēmu

XPath valoda

- XPath labumi
  - Vienkārša sintakse standartiem lietošanas gadījumiem
  - Jebkuru ceļu XML dokumentā ar papildus nosacījumiem ir iespējams specificēts ar XPath
  - Jebkuru mezglu XML dokumentā var unikāli identificēt

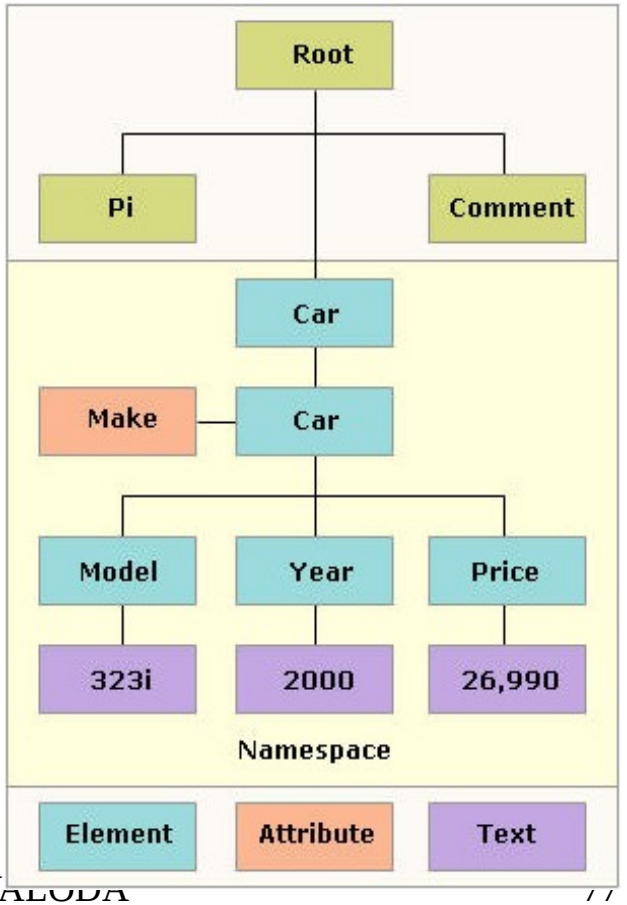


XPath valoda

- Xpath valodā XML dokuments konceptuāli tiek uztverts par koku, kurā katra dokumenta daļa ir mezgls

XPath nosaka septiņus mezgla tipus

- Sakne
- Elements
- Atribūts
- Teksts
- Kommentārs
- Apstrādes instrukcija
- Vārdu telpa



XPath izteiksmes

- XPath izteksme var atgriezt dokumenta mezglu kopu, boolean, simbolu virknes vai skaitlisko vērtību.

Expression	Description
nodename	Selects all nodes with the name "*nodename*"
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes

XPath izteiksmes piemēri

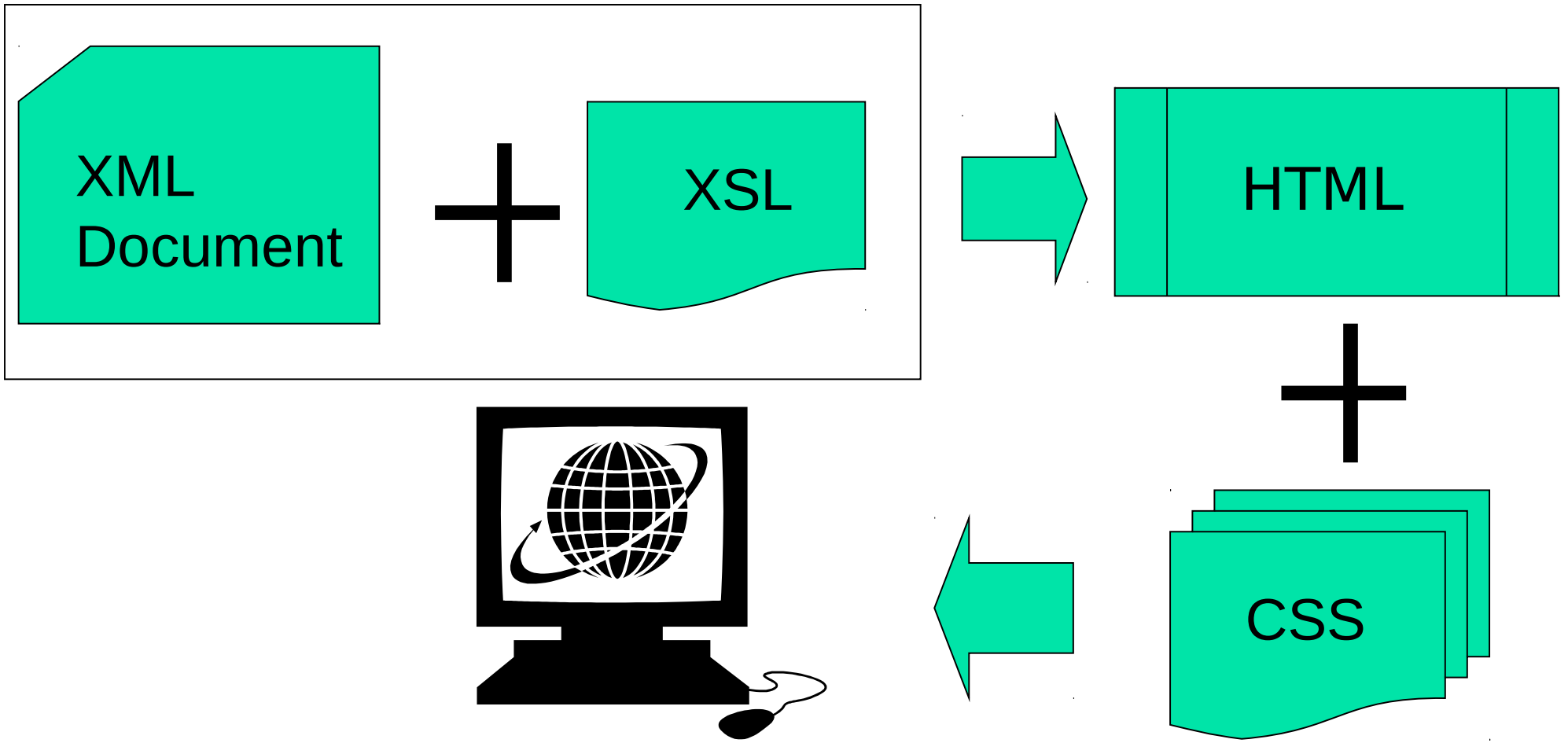
Path Expression	Result
bookstore	Selects all nodes with the name "bookstore"
/bookstore	Selects the root element bookstore **Note:** If the path starts with a slash (/) it always represents an absolute path to an element!
bookstore/book	Selects all book elements that are children of bookstore
//book	Selects all book elements no matter where they are in the document
bookstore//book	Selects all book elements that are descendant of the bookstore element, no matter where they are under the bookstore element
//@lang	Selects all attributes that are named lang

Path Expression	Result
/bookstore/*	Selects all the child nodes of the bookstore element
//*	Selects all elements in the document
//title[@*]	Selects all title elements which have any attribute

XSL Transformations

- XSLT = transformācijas komponents XSL tehnoloģijā
- Apraksta XML dokumenta transformācijas procesu, izmantojot transformācijas dzinēju
- XML documents un XSL stilu lapa(s) kalpo par ievaddatiem XSLT procesorā (XML transformācijas dzinēja)

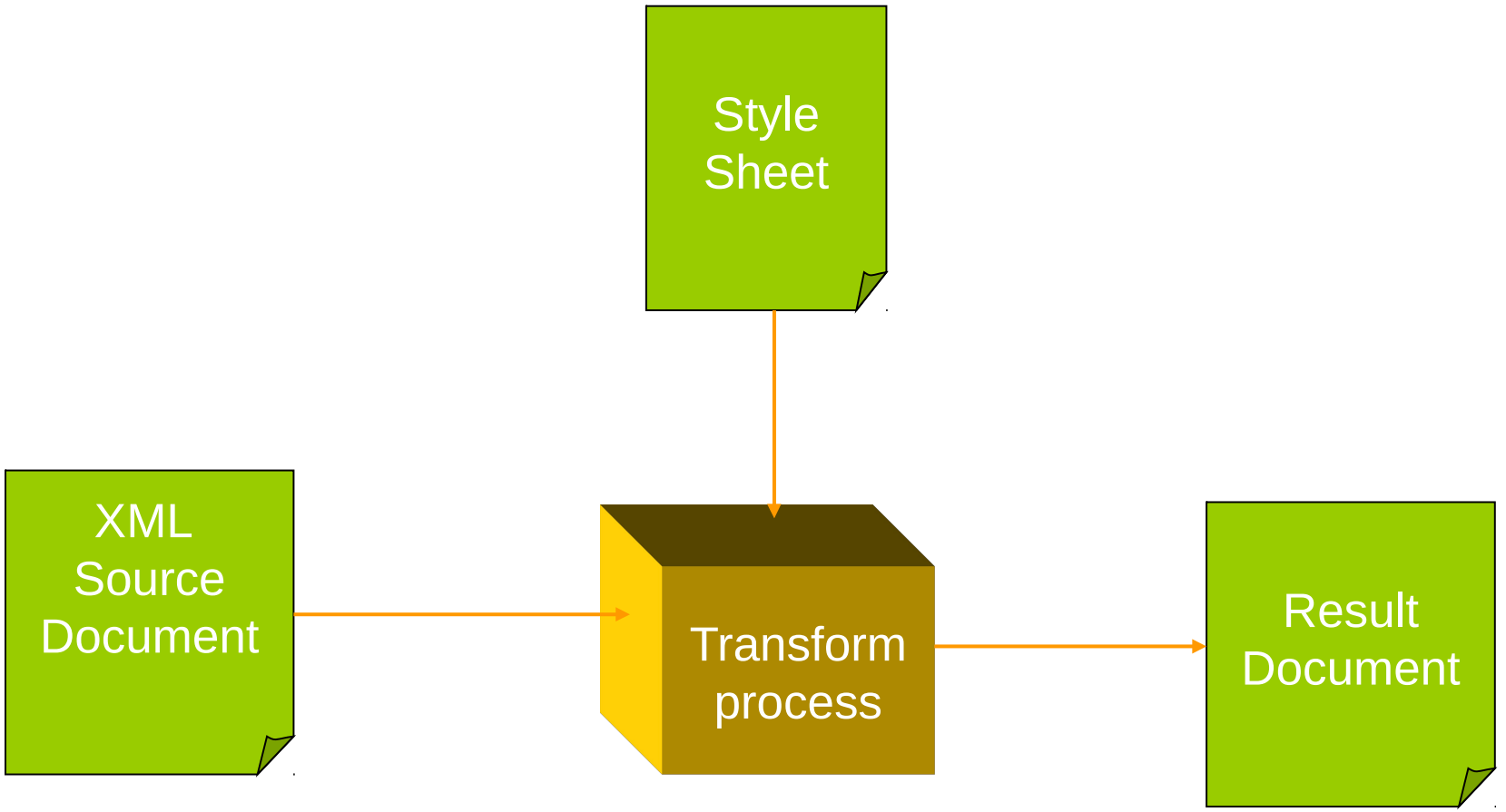
XSL Transformations



XSLT tehnoloģijas pamati

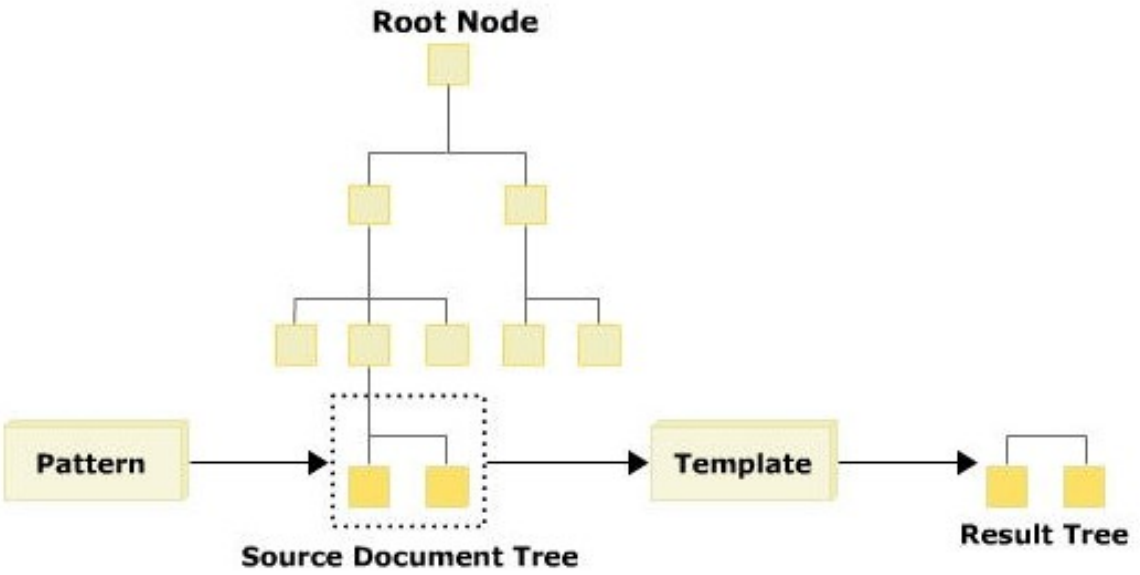
- XSLT ir W3C Recommendation
- XSLT ir svarīgākā XSL daļa, kura atbildīga par XML dokumenta transformāciju citā XML dokumentā
- XSLT izmanto XPath navigācijai XML dokumentā
- Pārlūkprogrammas ir atbalsta XML un XSLT.

XSLT tehnoloģijas pamati



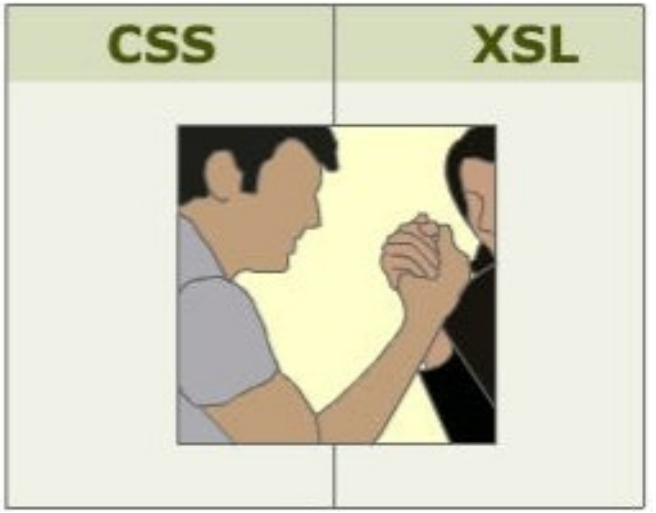
XSL apstrādes modelis

- XML processors nolasa XML dokumentu un parse to hierarhiskajā mezglu kokā
- XSL processors pielieto XSL stilu lapas noteikumus dokumenta mezglu kokam



CSS vs. XSL

- XSL un CSS ir divas atšķirīgas stilu lapu valodas
- Abas ir W3C rekomendēta pieeja XML dokumentu stilistiskai noformēšanai
- XSL ir jaudīgāka un sarežģītāka par CSS



CSS vs. XSL

- **Atšķirības**
  - CSS – stila lapu valoda HTML un XML dokumentiem
  - XSL – stila lapu valoda XML dokumentiem

  - CSS nosaka vizuālo lapas atspoguļošanu, nemainot oriģinālo dokumenta struktūru
  - XSL – tehnoloģija XML dokument transformēšanai

  - CSS neatbalsta kontrolējošas struktūras (piem. “if”)
  - XSL nodrošina kontrolējošas struktūras, ciklus

  - CSS – izmanto savu specifisku notāciju
  - XSL – izmanto XML notāciju

XSLT Struktūra un Sintakse

- Stilistiskie noteikumi tiek rakstīti failā ar .xsl paplašinājumu.

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

- .xsl saite ar dokumenta instanci notiek izmantojot instrukciju:

<?xml-stylesheet href=“xsl file” type=“text/xsl”?>

<xsl:template> elements

- XSL stilu lapas sastav no viena vai vairākām noteikumu kopām, sauktiem par **šābloniem** (angl. template)

```
1  <?xml version="1.0"?>
2  <xsl:stylesheet version="1.0"
3    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4    <xsl:template match="/">
5      <html>
6        <body>
7          <h1> XSL TEMPLATE SAMPLE</h1>
8        </body>
9      </html>
10   </xsl:template>
11 </xsl:stylesheet>
```

<xsl:template> elements

- Šablons tiek izmantots, lai kontrolētu XSLT procesora darbību
- Šablons sastāv no **Xpath izteiksmes**, kura identificē XML mezglu un **darbības** šī mezgla transformācijai

<xsl:template> elements

```
<?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>
  ...
 </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

<xsl:value-of> elements

- <xsl:value-of> elements tiek izmantots, lai izvilktu atlāsīta mezgla vērtību un pievienot to izvādes plūsmai

```
<xsl:template match="/">
  <html>
    <b><xsl:value-of select="CATALOG/NAME"/></b>
...
    <tr>
      <td><xsl:value-of select="catalog/cd/title"/></td>
      <td><xsl:value-of select="catalog/cd/artist"/></td>
...
  </html>
</xsl:template>
```

<xsl:for-each> elements

- <xsl:for-each> elements atļauj izveidot ciklus XSLT transformācijā.
- ar **<for-each>** XPath izteiksmi tiek atlasīta mezglu kopa, katrs no kuriem tiek atsevišķi apstrādāts pielietojot šablonā definētus noteikumus.

```
<xsl:template match="/">
  <html>
  <body>
    <xsl:for-each select="catalog/cd">
    <tr/>
    </xsl:for-each>
```

<xsl:sort> elements

<xsl:sort> XSLT elementu var izmantot līdzīgo elementu grupas kārtošanai.

- Sintakse

  <xsl:sort case-order=“upper-first” | “lower-first” | “name” | “text” order=“ascending” | “descending” select=“expression”>

  </xsl:sort>

```
<table border="1">
    <xsl:for-each select="CATALOG/CD">
    <xsl:sort select="TITLE" order="descending" />
    ...
<tr>
```

<xsl:if> elements

- <xsl:if> aprēķina Xpath nosacījumu attiecībā uz mezgla saturu un pielieto aprakstītu šablonu pozitīvai (true) **test** vērtībai.

- Sintakse

  <xsl:if test=“expression”>

  </xsl:if>

```
<xsl:if test="PRICE &gt; 10">
    Great price!
</xsl:if>
```

Piemērs

```
<xsl:template match="/">
    <html>
        <body>
            <h2>My CD Collection</h2>
            <b><xsl:value-of select="CATALOG/NAME"/></b>
            <table border="1">
                <tr>
                    <th>Title</th>  <th>Artist</th>
                </tr>
                <xsl:for-each select="CATALOG/CD">
                <xsl:sort select="TITLE" order="descending" />
                <tr>
                    <td><xsl:value-of select="TITLE" /></td>

                    <xsl:if test="PRICE &gt; 10">
                        SUPER PRICE
                    </xsl:if>
                    <td>ZZZ</td>
                </tr>
                </xsl:for-each>
            </table>
        </body>
    </html>
</xsl:template>
```