VEEBIRAKENDUSTE LOOMINE

MTAT.03.230 (6 EAP)

11. Loeng

Helle Hein

Teema: XML stiilid ja transformatsioonid

Täna loengus:

- XPATH
- XSLT - Extensible Stylesheet Language Transformations

On kasutatud Siim Karuse ja Tanel Tammeti slaide

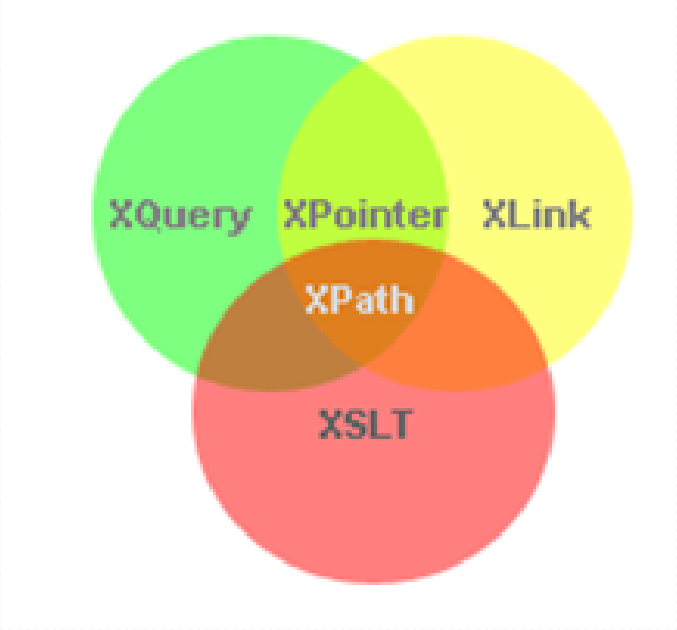
XSL - EXtensible Stylesheet Language

XSL koosneb kolmest osast:

- XSLT - keel XML dokumentide teisendamiseks
- XPath - keel XML dokumentides navigeerimiseks
- XSL-FO – keel XML dokumentide formaatimiseks

XPath

- on süntaks XML dokumendi osade defineerimiseks;
- kasutab avaldisi XML dokumendis navigeerimiseks;
- sisaldab standardfunktsioonide teeki;
- on XSLT põhielemendiks;
- on W3C soovitus.



XPATH

7 tüüpi tippe

- juurtipp (NB! Mitte juurelement!)
- element-tipp (`<a ...>...</a>`)
- tekstitipp (...`Tekst`...)
- atribuuditipp (... `href="http://vabavara.net"` ...)
- kommentaaritipp (... `<!-- ... -->` ...)
- nimeruumitipp (`xmlns`)
- töötlemisjuhistipp (*Processing instruction*) (PI) (... `<? ... ?>` ...)

XPATH

- XPath asukohatee (*location path*) koosneb ühest või mitmest sammust, mille vahel on /

- XPath asukohasamm (*location step*) koosneb kolmest osast:
  - telg (*axisname*) – valiku tüüp, määrab suuna;
  - tipu tüüp (*nodetest*) – tavaliselt nimi, täpsustab valikut;
  - predikaat (*expression*) - nurksulgudes ja täpsustab edasist .valikut. Võib kasutada rohkem kui ühte avaldist

```
axisname :: nodetest[expression1] [expression2] ...
```

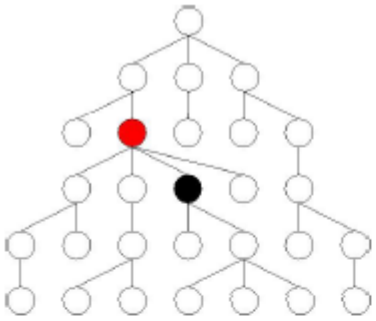
XPATH

Telg - adresseerimine

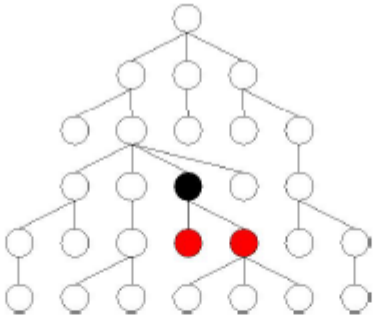
- **`child::`** – alamelemendid (vaikimisi)
- **`attribute::`** – atribuudid
- **`descendant::`** – alluvad elemendid
- **`ancestor::`** - eelnevad (ülem) elemendid
- **`self::`** – tipp ise
- **`following-sibling::`** - järgmised elemendid (õed-vennad)
- **`preceding-sibling::`** - eelmised elemendid (õed-vennad)
- **`following::`** - järgnevad (sama taseme) elemendid
- **`preceding::`** - eelnevad (sama taseme) elemendid
- **`ancestor-or-self`**
- **`descendant-or-self`**
- **`parent::`** – ülemelement
- **`namespace::`** - nimeruumid

XPATH

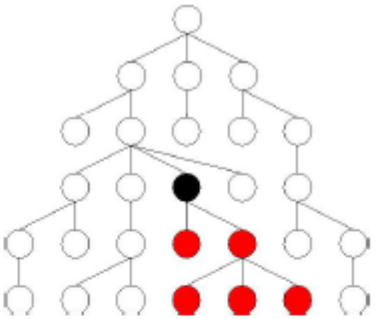
Telg:



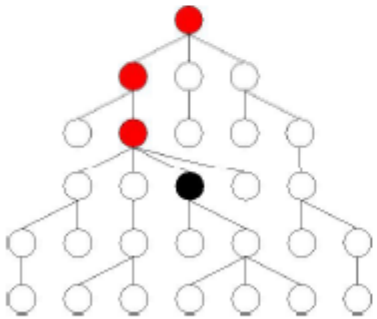
parent



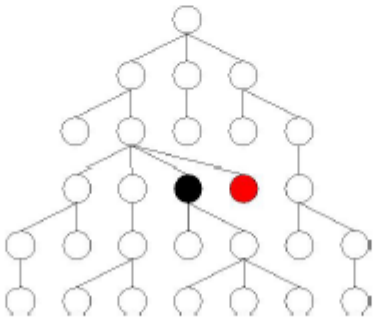
child



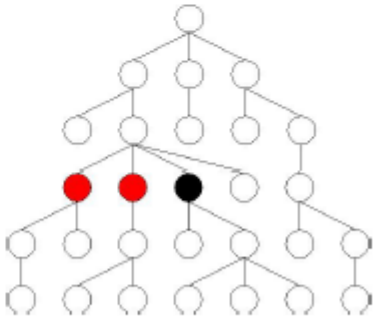
descendant



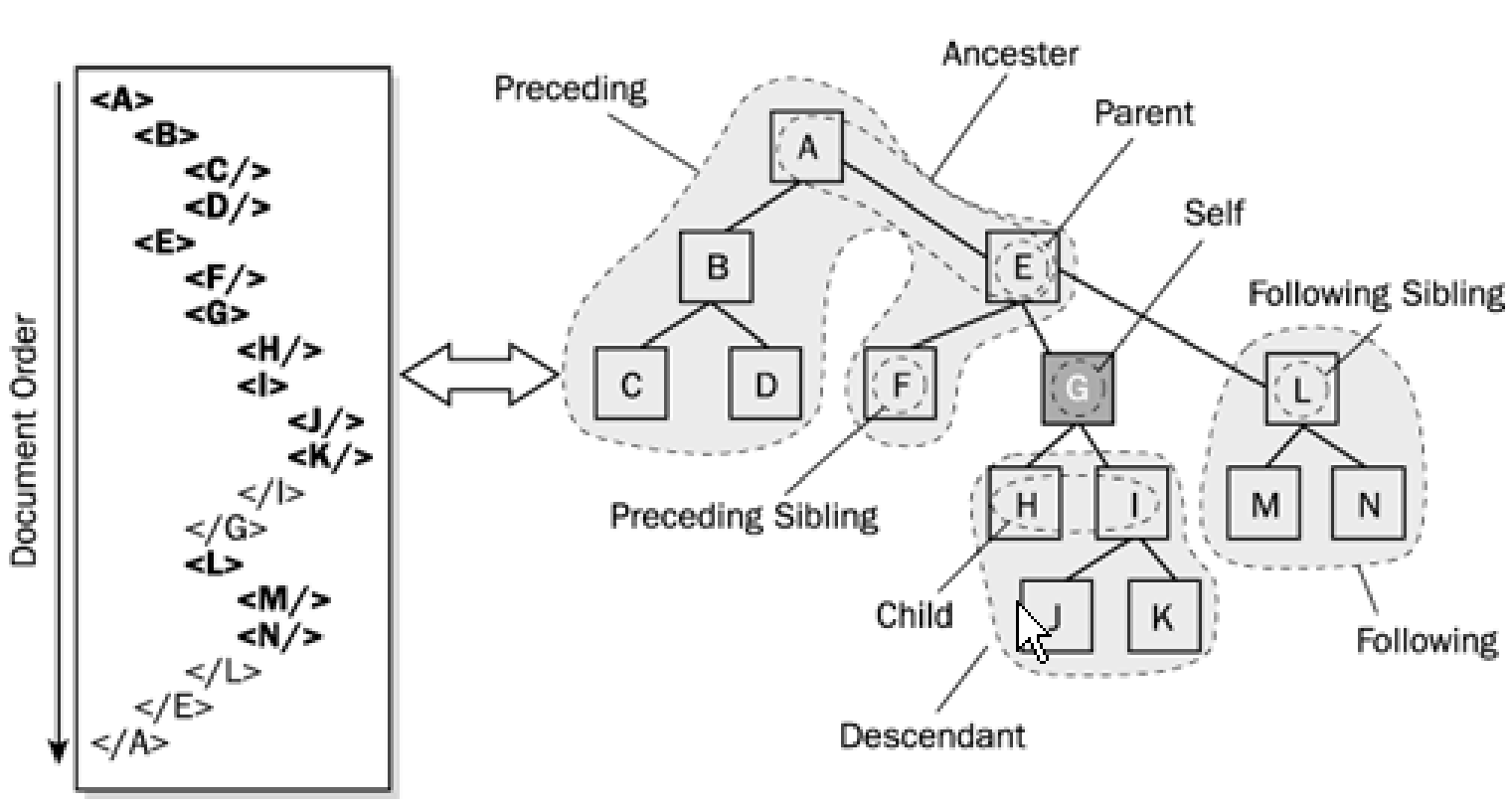
ancestor



following-sibling



preceding-sibling


Document Order
<A>
<B>
<C/>
<D/>
<E>
<F/>
<G>
<H/>
<I>
<J/>
<K/>
</I>
</G>
<L>
<M/>
<N/>
</L>
</E>
</A>
Preceding
Ancester
Parent
Self
Following Sibling
Preceding Sibling
Child
Descendant
Following
A
B
C
D
E
F
G
H
I
J
K
L
M
N

XPATH

Tipu tüüp (*nodetest*)

`text()` – tekstitipud;

`node()` – kõik tipud sõltumata tüübist;

`comment()` – kommentaaritipud;

`*` – kõik elementtipud

`name` – antud nimega elementtipud;

`nsname:*` - antud nimeruumiga tipud;

XPATH

Adresseerimine

Absoluutne tee algab `/`

`/` - valik alates juurtipust

`//` – tippude valik antud tipust alates dokumendis sõltumata asukohast, ekvivalentne `/descendant-or-self::node()/`

`.` – antud tipp, ekvivalentne `self::node()`

`..` – *parent* tipp, ekvivalentne `parent::node()`

`@` - atribuudid, nt `attribute::grade` on ekvivalentne `@grade`

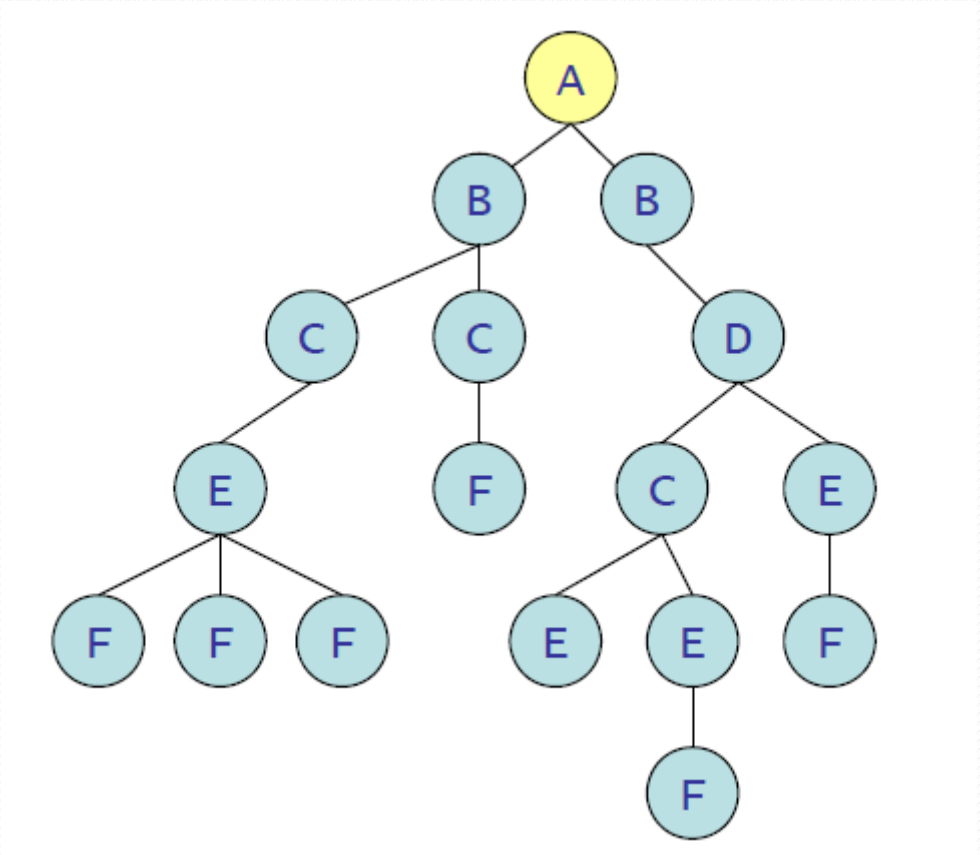
`@*` - kõik atribuut-tipud

XPATH

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <bookstore>
  <book>
   <title lang="eng">Harry Potter</title>
   <price>29.99</price>
  </book>
  <book>
   <title lang="eng">Learning XML</title>
   <price>39.95</price>
  </book>
</bookstore>
```

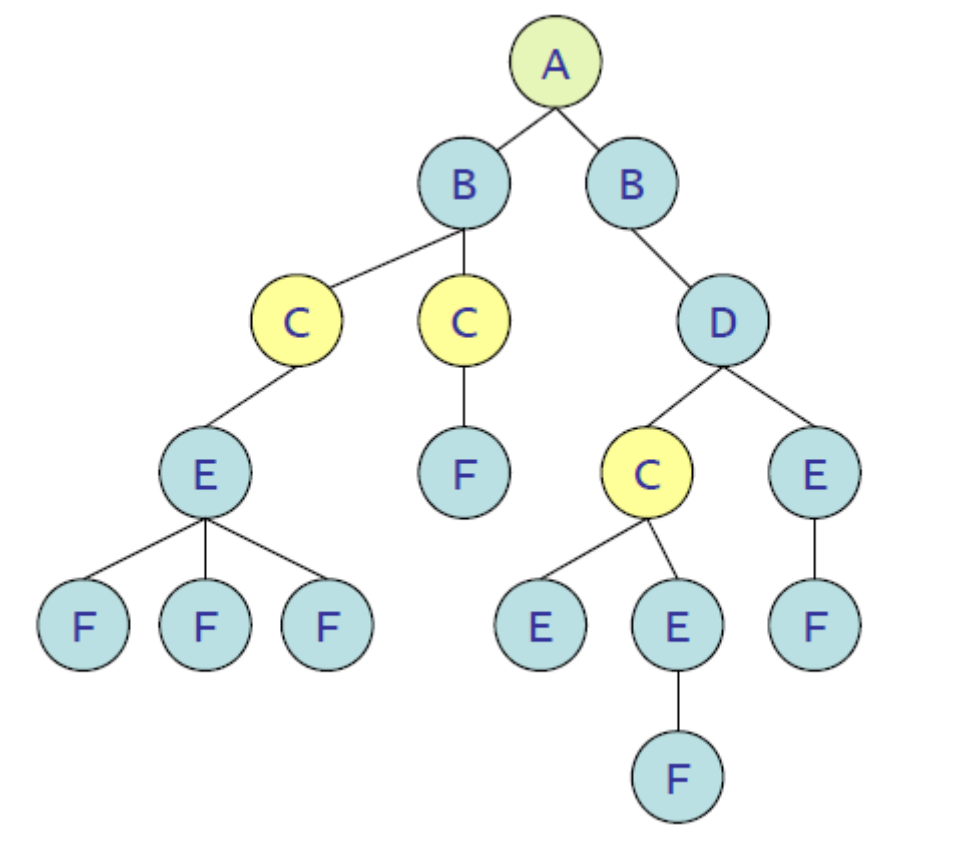
`bookstore` - Kõik elemendi `bookstore` alamtipud
`/bookstore` – juurelement `bookstore`
`bookstore/book` - kõik elementtipud `book`, mis on `bookstore` alamtipud
`//book` – kõik elemendid `book` sõltumata sellest, kus nad asuvad
`bookstore//book` – kõik elemendid `book`, mis on järglased elemendile `bookstore` või tema alluvatele sõltumata sellest, kus nad asuvad
`//@lang` – kõik atribuudid nimega `lang`

XPATH



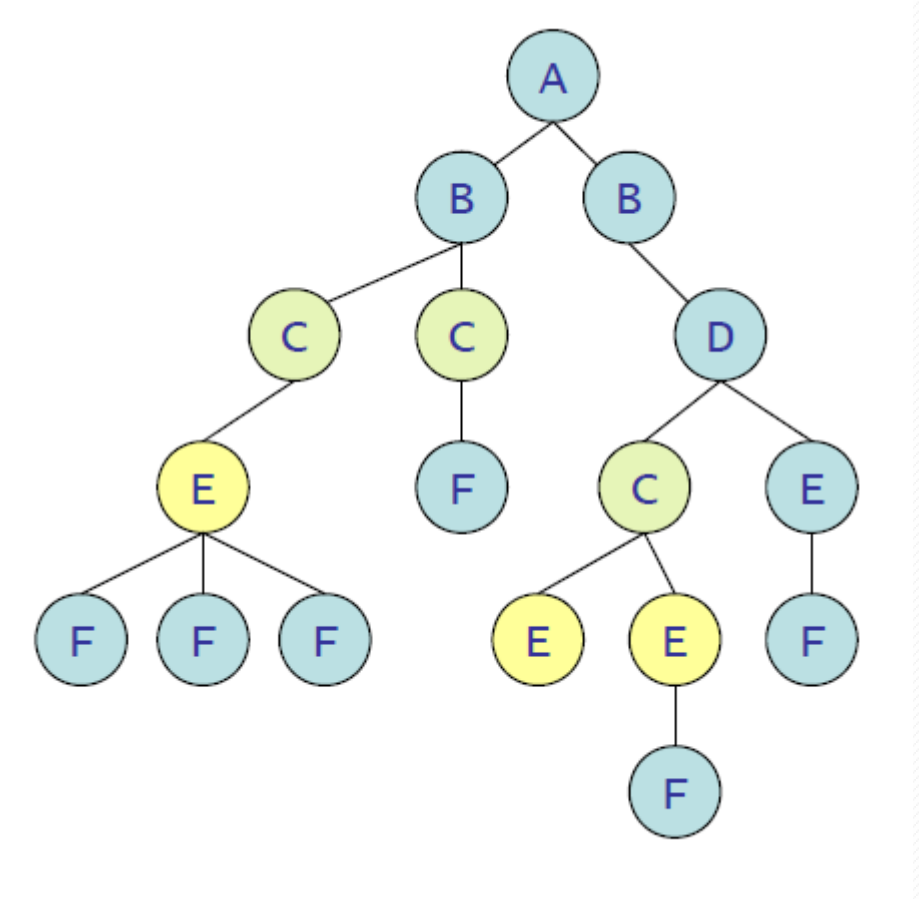
algustipp

XPATH



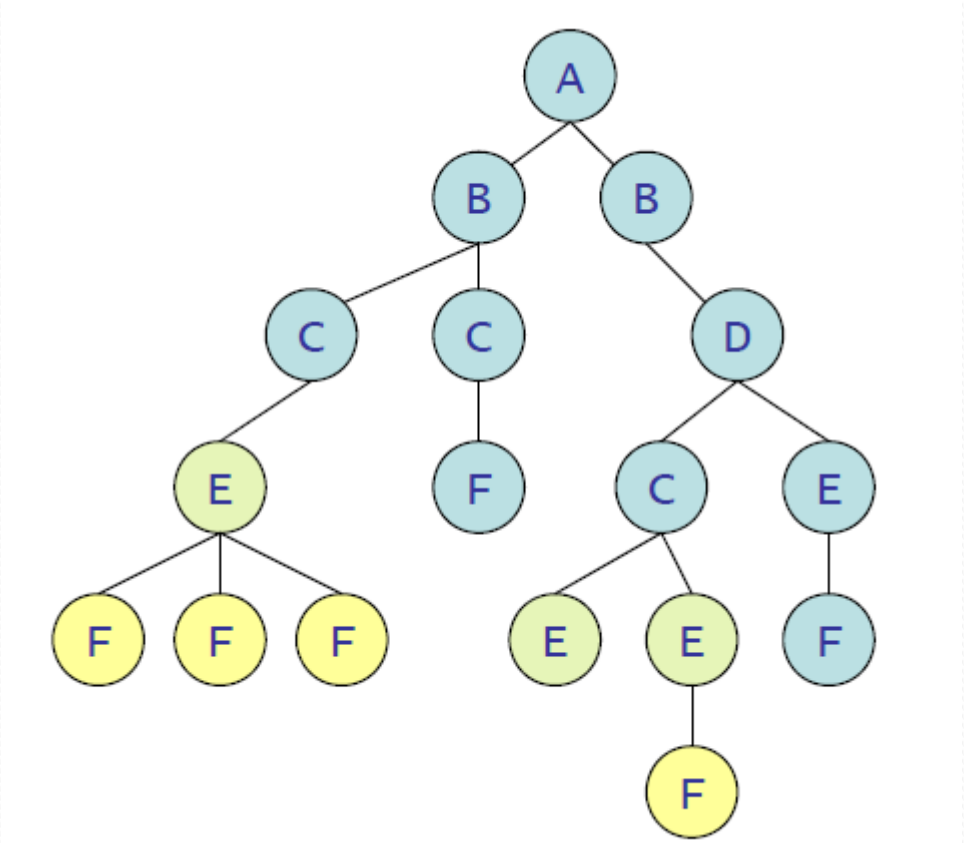

```
descendant::C
```

XPATH




```
descendant::C/child::E
```

XPATH




```
descendant::C/child::E/child::F
```

Predikaadid

- Nurksulgudes

  `*[@id=12]`

- Kontekst on nurksulgudele eelneva lause kontekst.

  `and`, `or`

  `=`, `!=`

  `<`, `>`, `<=`, `>=`

  `begins-with(string, string)`, `contains(string, string)`

  `boolean(object)`, `not(boolean)`, `true()`, `false()`, `lang(string)`

  - järjekorranumber

XPath funktsioonid

- Funktsioonid:
  - tippudega manipuleerimiseks:

    `name(node),position(), last(), …`
  - sõnedega manipuleerimiseks:

    `string-length(string), begins-with(string, string), …`
  - numbritega manipuleerimiseks:

    `+, -, mod, div, …`

FUNCTIONS	DESCRIPTION
number last()	Returns the number of the selected nodes (*context size*).
number position()	Returns the index of the selected node (*context position*).
number count(*node-set*)	Returns the number of nodes in the argument node-set.
string local-name(*node-set?*)	Returns the local part of the node. The first node in the argument node-set in document order is used as the target node.
string namespace-uri (*node-set?*)	Returns the namespace URI of the node. The first node in the argument node-set in document order is used as the target node.
string concat(*string, string, string**)	Returns the concatenation of its arguments.
boolean starts-with(*string, string*)	Checks that the first argument string starts with the second argument string
boolean contains (*string, string*)	Checks that the first argument string contains the second argument string.
number string-length (*string?*)	Returns the length of the argument.
string normalize-space (*string?*)	Normalizes the whitespace in the argument string by stripping leading and trailing whitespace characters and by replacing a sequence of white space characters with a single space.
boolean not(*boolean*)	Negates the argument boolean.
boolean true()	Always returns `true`.
boolean false()	Always returns `false`.
number sum(*node-set*)	Returns the sum of the argument node-converted to a number through its string representation.

XPATH EXAMPLE	ABBREVIATION	DESCRIPTION
/	/	Selects the document root. Note that this is not the document element
child::person	person	Selects the person element(s) among child elements under the context node.
child:person | /	person | /	Selects person element(s) or the document root. "|" stands for a union (logical OR).
/child::Address Book	/AddressBook	Selects the AddressBook element under the document root. That is, AddressBook is selected if it is the document element; otherwise, no element is selected.
child::*	*	Selects all the child elements under the context node. Note that this does not select any text nodes or attributes.
child::text()	text()	Selects all the text nodes under the context node. Note that if text nodes appear before and after an element, they are not concatenated but are selected as a set of text nodes.
attribute::type	@type	Selects the type attribute of the context node.
/descendant-or-self::node ()/child::email	//email	Selects all the email elements in a document. Note that // is short for /descendant-or-self::node()/. The node () function selects all nodes, whatever their node type.
child::person/child::email [position()=1]	person/email[1]	Selects the first email element under all the person child elements in a context node. You can specify additional (or detail) conditions in "[ ]." This part is called the ***predicate*** in XPath.
child::person [child::email and attribute:: id="12345"]	person[email and@id="12345"]	Selects all the person elements in a context node that have at least one email element and an id attribute with the value 12345
child::name[attribute:: family="neyama"] [position()= last()]	name [@family="neyama"] [last()]	Selects the last name element in a context node that has a family attribute and the value neyama

XPATH EXAMPLE	ABBREVIATION	DESCRIPTION
`/descendant-or-self::node()/child::email`	`//email`	Selects all the `email` elements in a document. Note that `//` is short for `/descendant-or-self::node()/`. The node () function selects all nodes, whatever their node type.
`child::person/child::email[position()=1]`	`person/email[1]`	Selects the first `email` element under all the `person` child elements in a context node. You can specify additional (or detail) conditions in "`[ ]`." This part is called the ***predicate*** in XPath.
`child::person [child::email and attribute:: id="12345"]`	`person[email and@id="12345"]`	Selects all the `person` elements in a context node that have at least one `email` element and an `id` attribute with the value `12345`
`child::name[attribute:: family="neyama"] [position()= last()]`	`name [@family="neyama"] [last()]`	Selects the last `name` element in a context node that has a `family` attribute and the value `neyama`

/*[1]//ajakulu[@operatsioon = ../*[2]/@operatsioon]/@aeg

```
<?xml version="1.0" encoding="utf-8"?>
<html tiitel="VabaVaraVeeb">
    <teated/>
    <moodul_vvvprofileerija>
        <ajakulud nimi="Ajakulu arvestus">
            <ajakulu moodul="init" operatsioon="init" aeg="0.0541"/>
            <ajakulu moodul="VVVNavi" operatsioon="init" aeg="0.0083"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="teeOperatsioonid"
              aeg="0.0004"/>
            <ajakulu moodul="VVVProfileerija" operatsioon="esita" aeg="0.237"/>
            <ajakulu moodul="VVVFilter" operatsioon="esita" aeg="0.022"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="esitaMoodulid"
              aeg="3.7427"/>
        </ajakulud>
    </moodul_vvvprofileerija>
</html>
```

/*[1]//ajakulu[@operatsioon = ../*[2]/@operatsioon]/@aeg

```
<?xml version="1.0" encoding="utf-8"?>
<html tiitel="VabaVaraVeeb">
    <teated/>
    <moodul_vvvprofileerija>
        <ajakulud nimi="Ajakulu arvestus">
            <ajakulu moodul="init" operatsioon="init" aeg="0.0541"/>
            <ajakulu moodul="VVVNavi" operatsioon="init" aeg="0.0083"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="teeOperatsioonid"
              aeg="0.0004"/>
            <ajakulu moodul="VVVProfileerija" operatsioon="esita" aeg="0.237"/>
            <ajakulu moodul="VVVFilter" operatsioon="esita" aeg="0.022"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="esitaMoodulid"
              aeg="3.7427"/>
        </ajakulud>
    </moodul_vvvprofileerija>
</html>
```

/*[1]//ajakulu[@operatsioon = ../*[2]/@operatsioon]/@aeg

```
<?xml version="1.0" encoding="utf-8"?>
<html tiitel="VabaVaraVeeb">
    <teated/>
    <moodul_vvvprofileerija>
        <ajakulud nimi="Ajakulu arvestus">
            <ajakulu moodul="init" operatsioon="init" aeg="0.0541"/>
            <ajakulu moodul="VVVNavi" operatsioon="init" aeg="0.0083"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="teeOperatsioonid"
              aeg="0.0004"/>
            <ajakulu moodul="VVVProfileerija" operatsioon="esita" aeg="0.237"/>
            <ajakulu moodul="VVVFilter" operatsioon="esita" aeg="0.022"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="esitaMoodulid"
              aeg="3.7427"/>
        </ajakulud>
    </moodul_vvvprofileerija>
</html>
```

/*[1]//ajakulu[@operatsioon = ../*[2]/@operatsioon]/@aeg

```
<?xml version="1.0" encoding="utf-8"?>
<html tiitel="VabaVaraVeeb">
    <teated/>
    <moodul_vvvprofileerija>
        <ajakulud nimi="Ajakulu arvestus">
            <ajakulu moodul="init" operatsioon="init" aeg="0.0541"/>
            <ajakulu moodul="VVVNavi" operatsioon="init" aeg="0.0083"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="teeOperatsioonid"
              aeg="0.0004"/>
            <ajakulu moodul="VVVProfileerija" operatsioon="esita" aeg="0.237"/>
            <ajakulu moodul="VVVFilter" operatsioon="esita" aeg="0.022"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="esitaMoodulid"
              aeg="3.7427"/>
        </ajakulud>
    </moodul_vvvprofileerija>
</html>
```

/*[1]//ajakulu[@operatsioon = ../*[2]/@operatsioon]/@aeg

```
<?xml version="1.0" encoding="utf-8"?>
<html tiitel="VabaVaraVeeb">
    <teated/>
    <moodul_vvvprofileerija>
        <ajakulud nimi="Ajakulu arvestus">
            <ajakulu moodul="init" operatsioon="init" aeg="0.0541"/>
            <ajakulu moodul="VVVNavi" operatsioon="init" aeg="0.0083"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="teeOperatsioonid"
              aeg="0.0004"/>
            <ajakulu moodul="VVVProfileerija" operatsioon="esita" aeg="0.237"/>
            <ajakulu moodul="VVVFilter" operatsioon="esita" aeg="0.022"/>
            <ajakulu moodul="VVVTeenusteHaldur" operatsioon="esitaMoodulid"
              aeg="3.7427"/>
        </ajakulud>
    </moodul_vvvprofileerija>
</html>
```

XSLT-Extensible Stylesheet Language Transformation

XSLT-Extensible Stylesheet Language Transformation

- Dokumentide teisenduste ja muutmiste kirjeldamise keel;
- Mõeldud dokumentide esituste teisendamiseks;
- 1.0 W3C Recommendation 1999;
- 2.0 W3C Recommendation 2007;
- Sisesed (*embedded*) ja välised (*external*);

Kasutusvaldkond

- B2B ja EAI rakendustes sõnumite teisendamiseks;
  - XSL teisendused on jõudsamad kui objekt-teisendused;
- Veebiteenustes päringute vastuste kasutajale sobival kujul esitamiseks;
- Veebiportaalides andmete kasutajale sobival kujul esitamiseks;
- XML andmete kasutajale esitamiseks (nt. Windows Live Messenger ajalugu);

XSLT sisendid

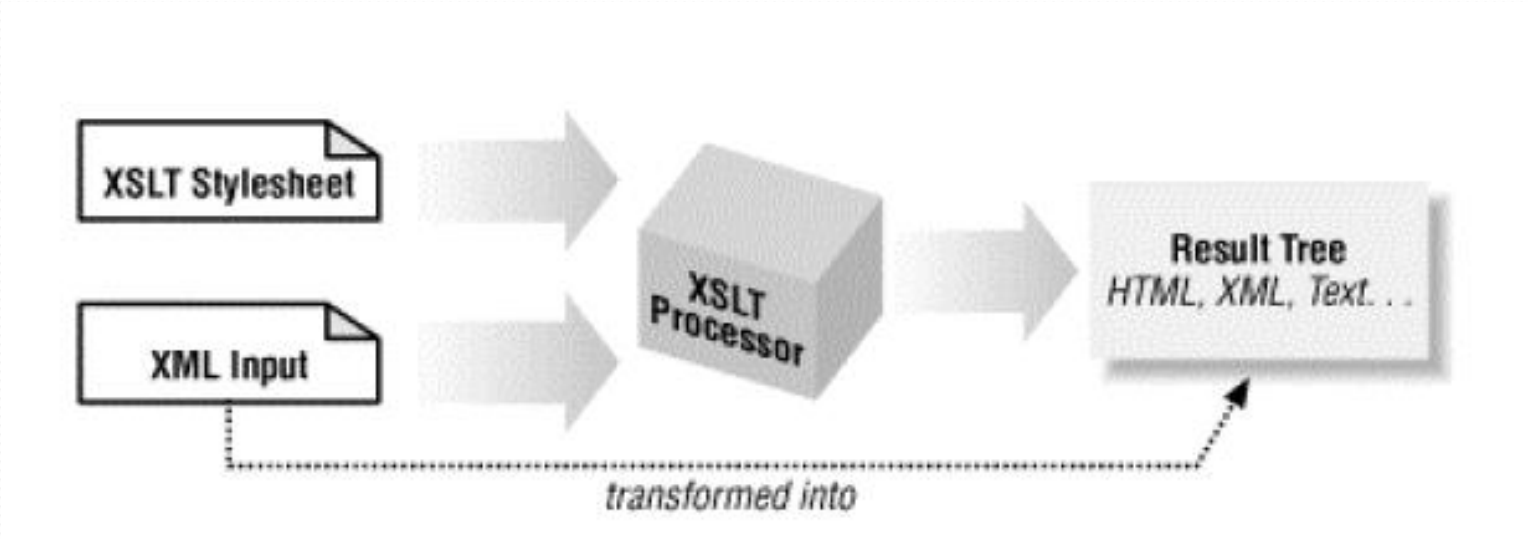
Kõik, mida saab adresseerida Xpath-iga:

- XML failid
- Andmehoidlad
- O-O keele teegid
- Failid

XSLT väljundid

Kõik failiformaadid:

- XML
- HTML
- XSL-FO
- Andmefail


XSLT Stylesheet
XML Input
XSLT Processor
Result Tree
HTML, XML, Text. . .
transformed into

XML “ilusaks väljanäitamiseks” on kolm põhiviisi:

- **CSS**: *cascading style sheets* fail, sarnane html kujundamisele;
  CSS failis saab öelda, kuidas mingit elementi visuaalselt näidata: värvid, font, paigutus. Tugi Mozillas ja IE-s.

- **XSLT**: XML teisendamine HTML-ks
  XSLT failis olevate teisendusreeglitega teisendatakse XML harilikuks html-ks ja näidatakse seda. Tugi Mozillas ja IE-s.

- **XSL-FO**: XML teisendamine spetsiaalsesse kujunduskeelde. XSL-FO on spetsiaalne XML süntaksiga kujunduskeel, võimas ja keeruline.

CSS

raamatud.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="raamatud.css"?>
<raamatud>
   <raamat>
     <autor>Amy Shuen</autor>
     <pealkiri>Web 2.0: A Strategy Guide</pealkiri>
     <hind currency="EEK">325.20</hind>
   </raamat>
   <raamat>
     <autor>Tom Brinck</autor>
     <pealkiri>Usability for the web</pealkiri>
     <hind currency="EUR">41.10</hind>
   </raamat>
</raamatud>
```

raamatud.css

```
<STYLE TYPE="text/css">
    BODY { margin: 10px; color: black; font-family: Arial,

Helvetica,sans-serif; }
    raamat { color:blue; padding: 10px; }
    autor { color:black; font-weight:bold;}
</STYLE>
```

XSL põhiidee

- XSL on nn. mallipõhine (*template*) keel, nagu PHP ja JSP:
  - XSL faili sisu ongi teisenduse tulemus
  - XSL faili sisse saab panna `xsl`-keelseid elemente, mis asendatakse lähtefaili juppidega (alltoodud näites ühtegi sellist elementi pole).
  - Ükskõik, millisele `xml` failile järgmist `xsl`-faili rakendada, tulemus on sama.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0“
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
   <body>Hello</body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

`xsl` keele nimi (url)

See tekst on teisenduse tulemus

`esimene.xsl`

Väga lihtne näide

Fail `data.xml`

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="render.xsl"?>
<message>Hei!</message>
```

Fail `render.xsl`

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template match="/">
  <html><body>
   <h1><xsl:value-of select="message"/></h1> </body></html>
 </xsl:template>
</xsl:stylesheet>
```

Kuidas XSLT töötab

- XML dokument loetakse sisse ja luuakse tippude puu

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

malli abil valitakse terve puu

- Malli sees olevaid reegleid rakendatakse sobivatele tippudele, st muudetakse XML puu struktuuri
  - Kui on veel malle, siis neid tuleb välja kutsuda peamallist
- Osa XML puust võib jääda muutmata (kui ei sobi kriteeriumiga)
- Kui kogu mall on rakendatud, siis kirjutatakse puu välja (tekstidokumenti)

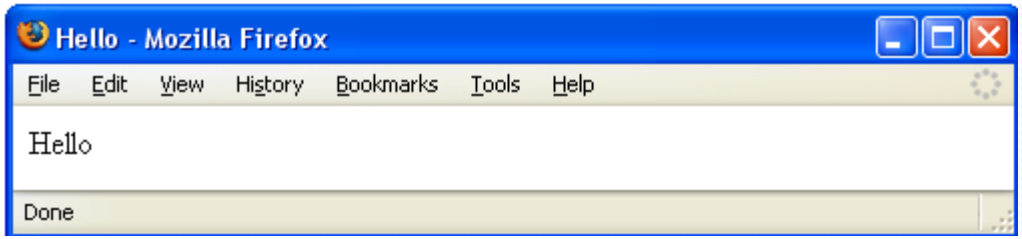
Näide

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href=esimene.xsl"?>
<fitnesscenter>
 <member level="platinum">
  <name>Jon</name>
  <phone type="home">6464-1234</phone>
  <phone type="work">6464-4321</phone>
  <favoritecolor>lightgrey</favoritecolor>
 </member>
</fitnesscenter>
```

Töötluskäsk brauseri jaoks

esimene.xml

Kui see fail avada brauseriga (Firefox või MS IE), siis kuvatakse `xsl` failis olev `html`, st tekst "Hello!"



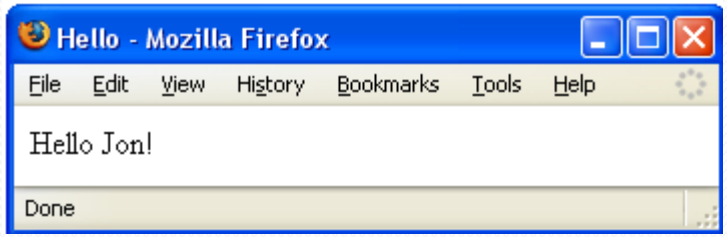
Täiendame näidet

teine.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                            version="1.0"
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
    <body>Hello
     <xsl:value-of select="/fitnesscenter/member/name"/>!
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

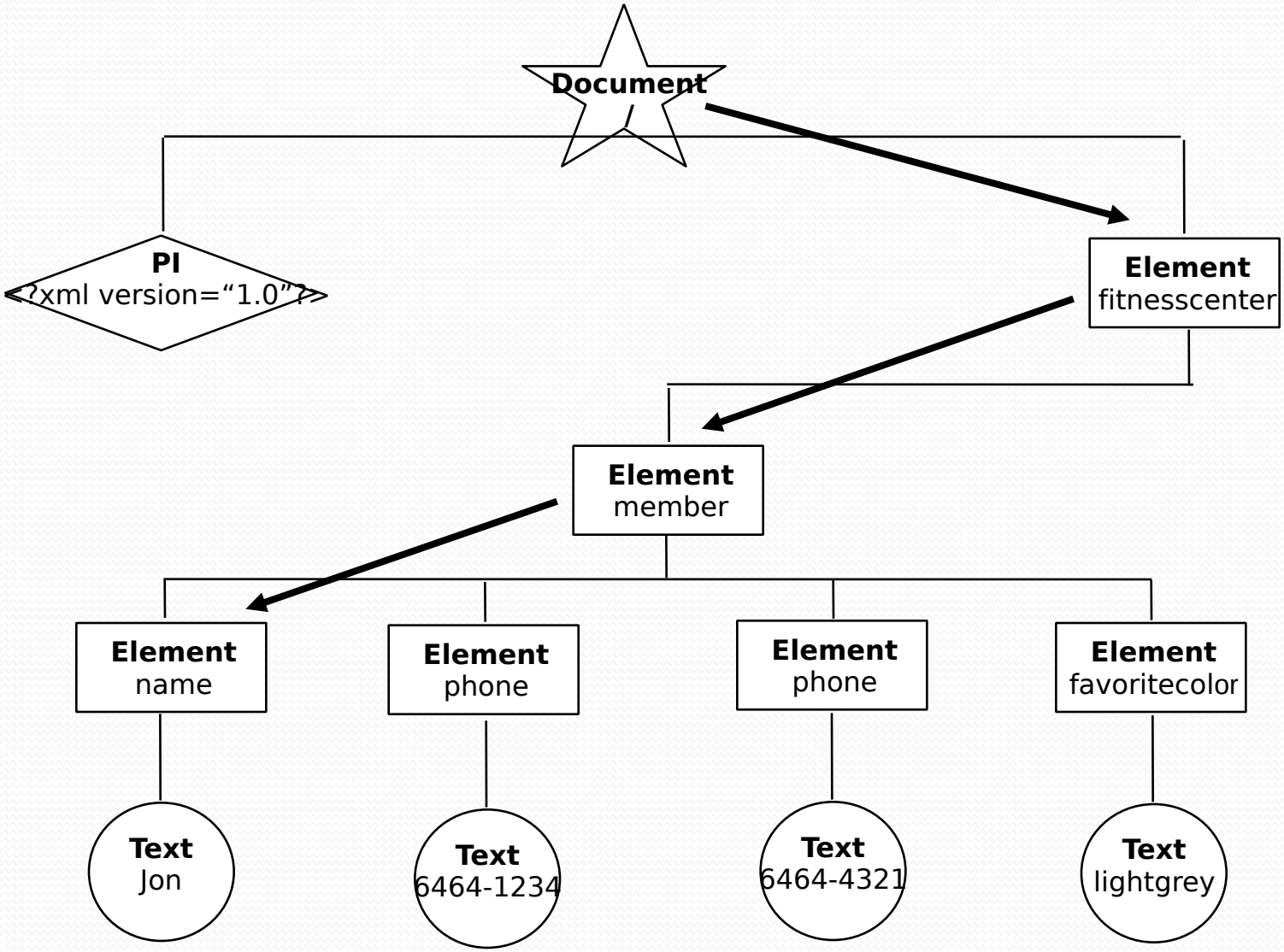
Sealt võetakse asenduse tulemus (XPATH keele avaldis)

asenduskäsk `xsl` keeles.



Tee `/FitnessCenter/Member/Name` tähendab:

Alusta dokumendi juurelemendist (algusest), otsi kohe selle alt element `fitnesscenter`, siis selle alt `member`, siis selle alt `name`, siis võta selle elemendi väärtus

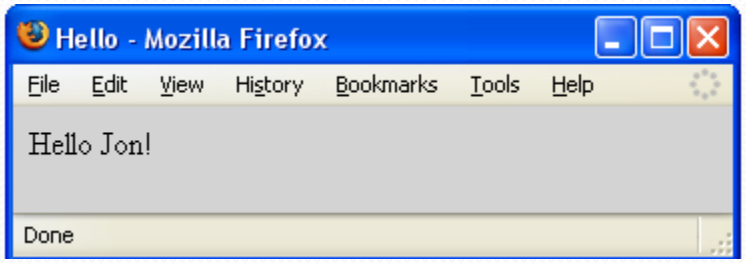

Document
PI
<?xml version="1.0"?>
Element
fitnesscenter
Element
member
Element
name
Element
phone
Element
phone
Element
favoritecolor
Text
Jon
Text
6464-1234
Text
6464-4321
Text
lightgrey

Teine asendus

kolmas.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                          version="1.0“
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
    <body bgcolor="{/fitnesscenter/member/favoritecolor}">
      Hello
     <xsl:value-of select="/fitnesscenter/member/name"/>!
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

Loogelistes sulgudes atribuudi **bgcolor** sisu asendatakse.

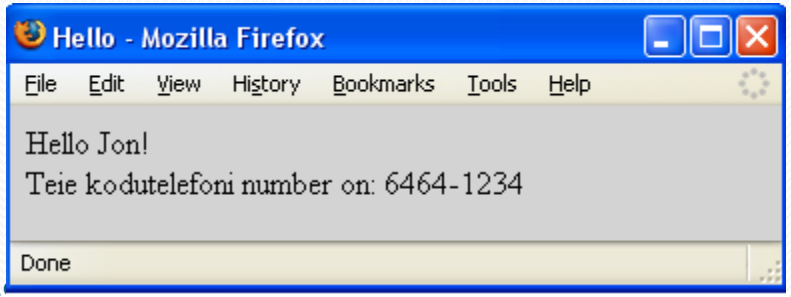


Kolmas asendus

neljas.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                          version="1.0"
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
   <body bgcolor="{/fitnesscenter/member/favoritecolor}">
     Hello
    <xsl:value-of select="/fitnesscenter/member/name"/>!
    <br/>
      Teie kodutelefoni number on:
        <xsl:value-of
    select="/fitnesscenter/member/phone[@type='home']"/>
   </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

Valib elemendi **phone**, millel on atribuut **type** väärtusega "**home**"

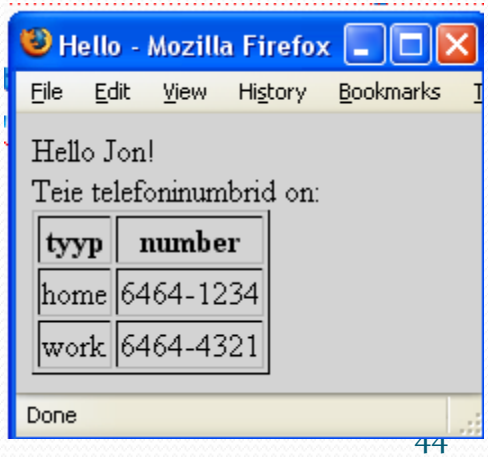


Tsükkel

viies.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                          version="1.0"
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
   <body bgcolor="{/fitnesscenter/member/favoritecolor}">
     Hello
    <xsl:value-of select="/fitnesscenter/member/name"/>!
    <br/> Teie telefoninumbrid on:
    <table border="1" width="25%">
     <tr><th>tyyp</th><th>number</th></tr>
     <xsl:for-each select="/fitnesscenter/member/phone">
     <tr>
      <td><xsl:value-of select="@type"/></td>
      <td><xsl:value-of select="."/></td>
     </tr>
     </xsl:for-each>
    </table>
   </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

Tsükli sees suhtelised teed (*relative path*)

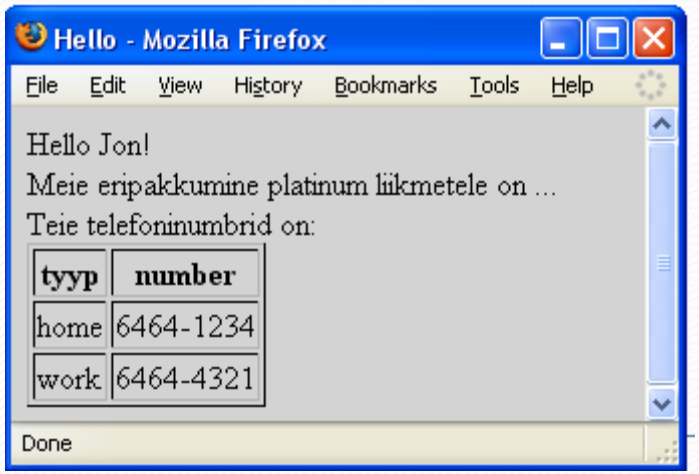


Tingimus/valik

kuues.xsl

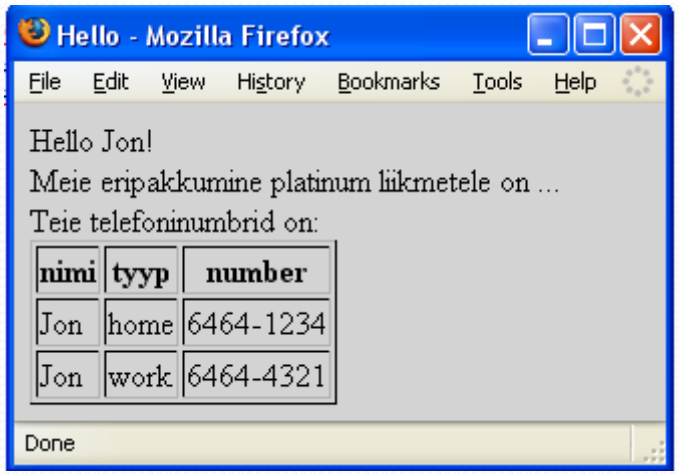
Täiendame oma `xsl` faili nii, et ainult plaatina-tasemel liikmetele teeme html-s pakkumise (st, siis kui `<Member level="platinum">`)

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                        version="1.0"
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
   <body bgcolor="{/fitnesscenter/member/favoritecolor}">
     Hello
      <xsl:value-of select="/fitnesscenter/member/name"/>!
     <xsl:if test="/fitnesscenter/member/@level='platinum'">
      Meie eripakkumine platinum liikmetele on ...
      <br/>
     </xsl:if>
    <br/> Teie telefoninumbrid on:
   <table border="1" width="25%">
    <tr><th>tyyp</th><th>number</th></tr>
    <xsl:for-each select="/fitnesscenter/member/phone">
    <tr>
     <td><xsl:value-of select="@type"/></td>
     <td><xsl:value-of select="."/></td>
    </tr>
    </xsl:for-each>
   </table>
   </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```



Elementide võtmine dokumendist mujalt

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                        version="1.0"
 <xsl:output method="html"/>
 <xsl:template match="/">
  <html>
   <head><title>Hello</title></head>
   <body bgcolor="{/fitnesscenter/member/favoritecolor}">
     Hello
    <xsl:value-of select="/fitnesscenter/member/name"/>!
    <xsl:if test="/fitnesscenter/member/@level='platinum'">
    Meie eripakkumine platinum liikmetele on ...
    <br/>
   </xsl:if>
    <br/> Teie telefoninumbrid on:
   <table border="1" width="25%">
    <tr><th>nimi</th><th>tyyp</th><th>number</th></tr>
    <xsl:for-each select="/fitnesscenter/member/phone">
    <tr>
     <td><xsl:value-of select="../name"/></td>
     <td><xsl:value-of select="@type"/></td>
     <td><xsl:value-of select="."/></td>
    </tr>
    </xsl:for-each>
   </table>
   </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```



Mallide kasutamine

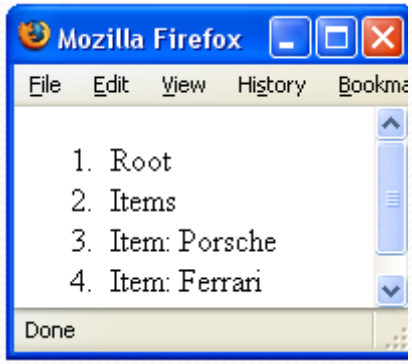
```
<?xml version="1.0"?>
<?xml-stylesheet href="example1.xsl" type="text/xsl"?>
<items>
 <item partNum="123-AB">
  <productName>Porsche</productName>
  <quantity>1</quantity>
 </item>
 <item>
  <productName>Ferrari</productName>
  <quantity>2</quantity>
 </item>
</items>
```

Mallide kasutamine

example1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="html"/>
<xsl:template match="/">
<html>
 <head></head>
 <body>
  <ol>
   <li>Root</li>
   <xsl:apply-templates/>
  </ol>
 </body>
</html>
</xsl:template>
<xsl:template match="items">
  <li>Items</li>
   <xsl:apply-templates/>
</xsl:template>

<xsl:template match="item">
  <li>Item: <xsl:value-of select="productName"/></li>
</xsl:template>
</xsl:stylesheet>
```



Väljundpuu koostamine

```
<element name=string>
<attribute name=string>
<processing-instruction name=string>
<comment>
<text>
```

Väljundpuu koostamine

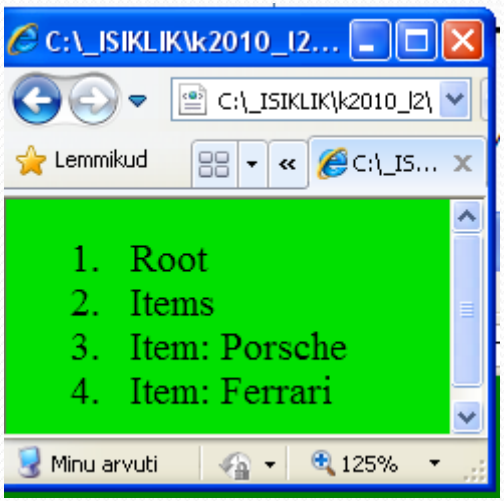
example2.xsl

XSL fragment:

```
…
<xsl:template match="/">
  <html>
    <head></head>
    <xsl:comment>
      taust roheliseks
    </xsl:comment>
    <xsl:element name="body">
    <xsl:attribute
        name="bgcolor">
     green
    </xsl:attribute>
      <ol>
      <li>Root</li>
      <xsl:apply-templates/>
     </ol>
    </xsl:element>
  </html>
</xsl:template>
…
```

XSL väljund:

```
<html>
<head></head>
<!—taust roheliseks-->
<body bgcolor="green">
<ol>
<li>Root</li>
<li>Items</li>
<li>Item: Porsche</li>
<li>Item: Ferrari</li>
</ol>
</body>
</html>
```



Struktuur

```
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:include href="..."/>
  <xsl:import href="..."/>
  <xsl:strip-space elements="..."/>
  <xsl:preserve-space elements="..."/>
  <xsl:variable name="...">...</xsl:variable>
  <xsl:param name="...">...</xsl:param>
  <xsl:output method="..." />
  <xsl:key name="..." match="..." use="..."/>
  <xsl:decimal-format name="..."/>
  <xsl:namespace-alias stylesheet-prefix="..." result-prefix="..."/>
  <xsl:attribute-set name="...">
    ...
  </xsl:attribute-set>

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

Käsud (stiilileht)

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
   <xsl:import href="import_uri"/>
   <xsl:include href="include_uri"/>
   <xsl:output
       method = "xml" | "html" | "text"
       version = string
       encoding = string
       omit-xml-declaration = "yes" | "no"
       standalone = "yes" | "no"
       doctype-public = string
       doctype-system = string
       cdata-section-elements = qnames
       indent = "yes" | "no"
       media-type = string />
```

Näide

```
...
<xsl:output method="html" version="4.0" encoding="iso-8859-1"
indent="yes"/>
```

Käsud (mall)

```
<xsl:template
  match = pattern
  name = qname
  priority = number
  mode = qname>
  <xsl:param
     name = qname
     select = expression>
      <!-- Content -->
  </xsl:param>
  <!-- Content -->
  </xsl:template>
```

Puu läbimine

```
<xsl:apply-templates
   select = node-set-expression
   mode = qname>
   <!-- Content: (xsl:sort | xsl:with-param)* -->
</xsl:apply-templates>
<xsl:for-each
   select = node-set-expression>
   <!-- Content: (xsl:sort*, template) -->
</xsl:for-each>
```

Muutuja ja parameeter

```
<xsl:variable
   name = qname
   select = expression>
   <!-- Sisu -->
</xsl:variable>
<xsl:call-template
   name = string>
   <xsl:with-param
     name = qname
     select = expression>
     <!-- Sisu -->
   </xsl:with-param>
</xsl:call-template>
```

Tingimused

```
<xsl:if
   test = boolean-expression>
   <!-- Content: template -->
</xsl:if>
<xsl:choose>
   <xsl:when
     test = boolean-expression>
     <!-- Content: template -->
   </xsl:when>
   <xsl:otherwise>
     <!-- Content: template -->
   </xsl:otherwise>
</xsl:choose>
```

XML spetsiifilised teisendused

```
<xsl:element
   name = qname
   namespace = uri-reference
   use-attribute-sets = qnames>
   <xsl:attribute
     name = qname
     namespace = uri-reference>
     <!-- Content: template -->
   </xsl:attribute>
   <!-- Content: template -->
</xsl:element>
<xsl:attribute-set
   name = qname
   use-attribute-sets = qnames>
   <!-- Content: xsl:attribute* -->
</xsl:attribute-set>
```

XML spetsiifilised teisendused

```
<xsl:text
   disable-output-escaping = "yes" | "no">
   <!-- Content: #PCDATA -->
</xsl:text>
<xsl:processing-instruction
   name = ncname >
   <!-- Content: template -->
</xsl:processing-instruction>
<xsl:comment>
   <!-- Content: template -->
</xsl:comment>
```

Väärtuste kopeerimine

Ainult tipp ilma atribuutide ja alamelementideta

```
<xsl:copy
  use-attribute-sets = qnames>
  <!-- Content: template -->
</xsl:copy>
```

Avaldise või tipu väärtus

```
<xsl:value-of
  select = string-expression
  disable-output-escaping = "yes" | "no" />
```

Atribuudi mallina

```
<img src="{$image-dir}/{href}" width="{size/@width}"/>
```

Puu koopia

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

Veel võimalusi

- Sorteerimine
- Võtmed ja id-d (vastavalt xml-i id ja idref atribuutidele)
- Nummerdamine
- Tühikute käitlemine
- Numbrite vormindamine

Näide

```
<!-- tegevuse nupu loomine -->
<xsl:template name="t_esita_tegevuse_nupp"
  match="tegevus" priority="1">
    <xsl:param name="tegevus" select="."/>
    <xsl:variable name="tegevuse_url">
      <xsl:call-template name="t_pane_kokku_tegevuse_url">
        <xsl:with-param name="tegevus" select="$tegevus"/>
      </xsl:call-template>
    </xsl:variable>
    <a href="{$tegevuse_url}"
      title="{$tegevus/@kirjeldus}" class="nupp">
      <xsl:value-of select="$tegevus/@nimetus"/>
    </a>
</xsl:template>
```

XSL-i sidumine XML failiga

```
<?xml-stylesheet type = "text/xml" href = uri ?>
```

URI võib viidata

- eraldi failis olevale stiililehele
- samas failis defineeritud stiililehele id järgi

Silumine

Logimine

```
<xsl:message
  terminate = "yes" | "no">
  <!-- Content: template -->
</xsl:message>
```

Siluriga

- Microsoft Visual Studio
- Altova XMLSpy
- Stylus Studio
- Oxygen
- …

Silumine käsurealt

1. Laadida alla `xalan-j_2_7_1`
   Lahtipakitud `.jar` failid peavad olema kättesaadavad `CLASSPASS` muutujas

```
public class xalanProcess {
  public static void main(String[] args)  {
     com.sun.org.apache.xalan.internal.xslt.Process
       process = new
       com.sun.org.apache.xalan.internal.xslt.Process();
     process._main(args);
  }
}
```

2. Kompileerida
3. Tulemus HTML on näha peale programmi käivitamist

```
>java xalanProcess -in fail.xml -xsl fail.xsl
```

Viited

- XPath lessons:
  - http://www.w3schools.com/xpath/default.asp
- Tutorials:
  - http://www.w3schools.com/xsl/default.asp
  - http://www.zvon.org/xxl/XSLTutorial/Output/index.html
  - http://www.w3schools.com/xslfo/xslfo_documents.asp

- XML-related tools:
  - Apache Xerces: http://xerces.apache.org
  - XML Spy http://www.xmlspy.com
  - MS XML parsing and processing tools: http://msdn.microsoft.com/XML/XMLDownloads

65