ELEC-C8203 Automaatiojärjestelmät 2 - XML

XSLT muunnokset

Pekka Aarnio

Luennon sisältö

- OSA 1: XSLT-kieli
- OSA 2: AutomationML lyhyt kuvaus harjoitustehtävän tueksi

OSA1

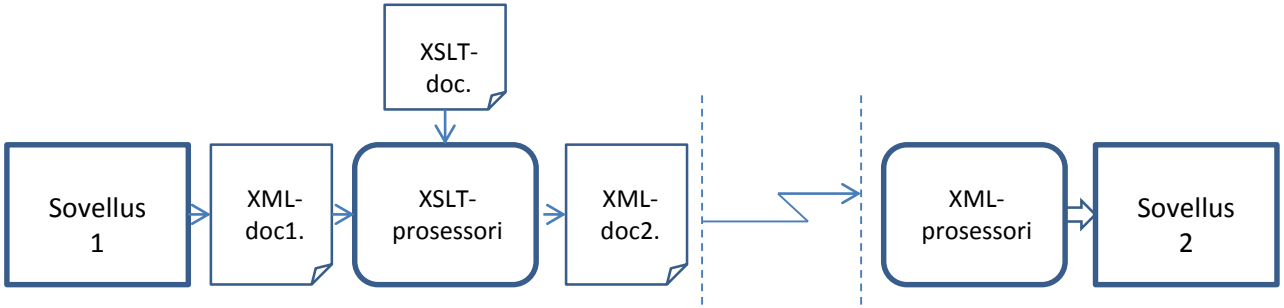
XSL-MUUNNOKSET

OSA 1-Sisältö

- XSLT:n käyttö tietointegroinnissa
- XSLT-prosessoinnin vaiheet
- Tietosisältöjen käsittelysäännöt
  - Value-of
  - For-each ja sort
  - ehtolauseet
- Rakenteiden luontisäännöt
  - Rakenteiden kopiointi
  - Uusien elementtien ja attribuuttien luonti
- Sapluunat eli templatet
- Edistyneet piirteet

XSLT:N KÄYTTÖSOVELLUKSET

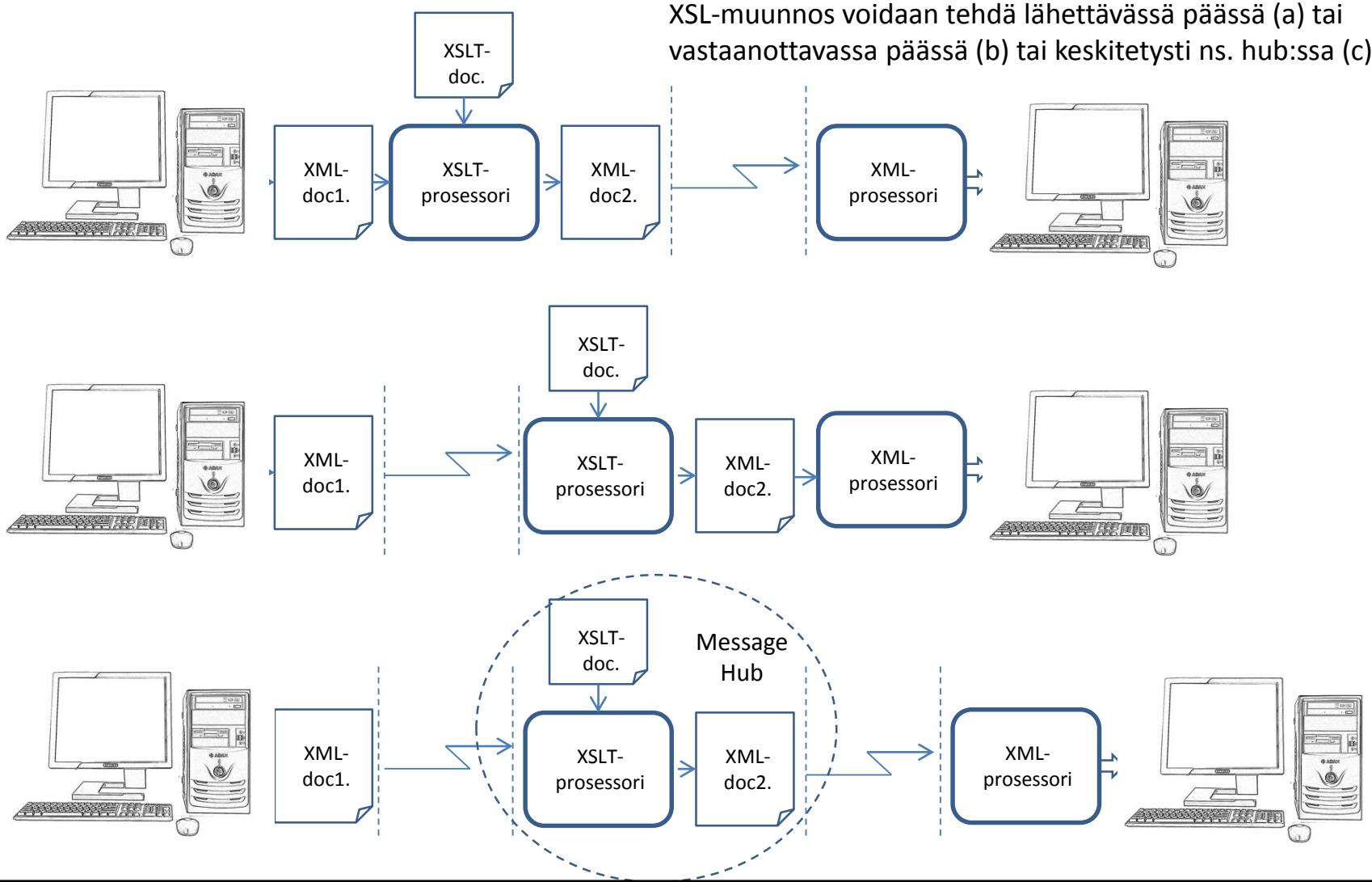
XML:n käyttösovellukset



- XML:a voidaan käyttää
  - perustuen valmiiseen kielioppiin (esim. AutomationML, B2MML, MathML)
  - käyttäen omaa sovelluskohtaista kielioppia
- Järjestelmien tietointegraatiossa tarvitaan muunnoksia kielestä toiseen
  - Eri organisaatioiden tietojärjestelmät käyttävät usein erilaista XML-esitystapaa samankin sovellusalueen tietojen esittämiseen
  - XSLT-prosessori muuntaa dokumentin (*doc1.xml)* toiseen muotoon (*doc2.xml)* lukemansa XSLT-dokumentin sääntöjen ohjaamana (ks. Kuva)

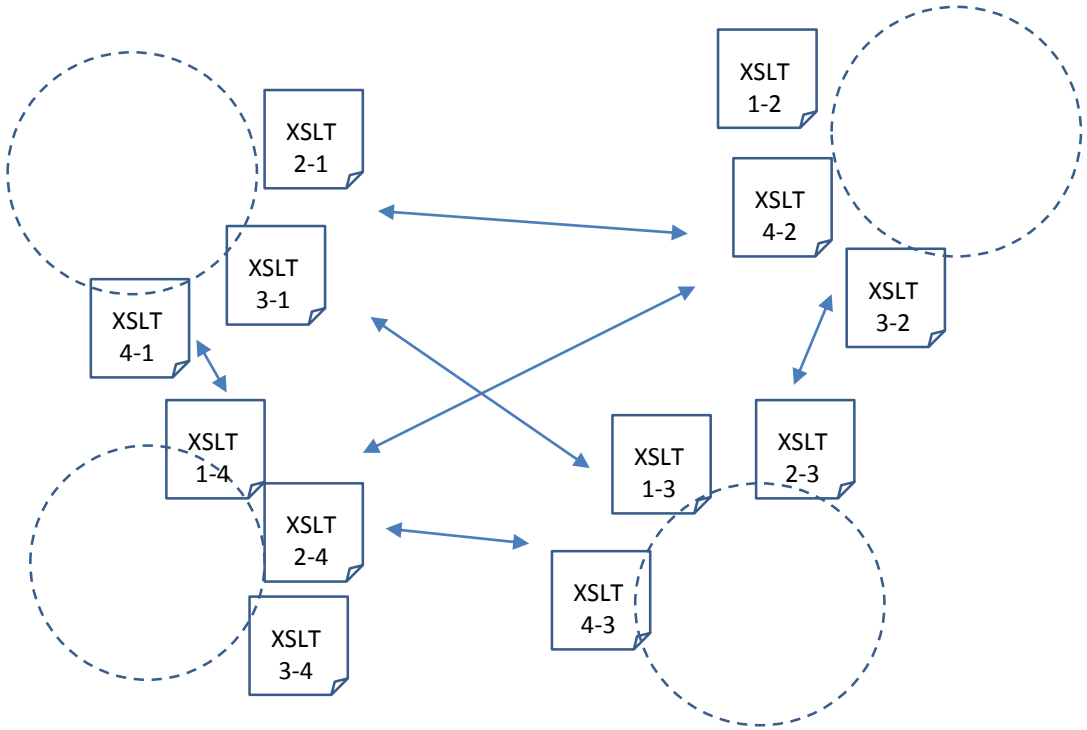
Tietointegraatio – XML tiedonsiirto

XML-tiedonsiirto kahden organisaation välillä:
XSL-muunnos voidaan tehdä lähettävässä päässä (a) tai vastaanottavassa päässä (b) tai keskitetysti ns. hub:ssa (c)

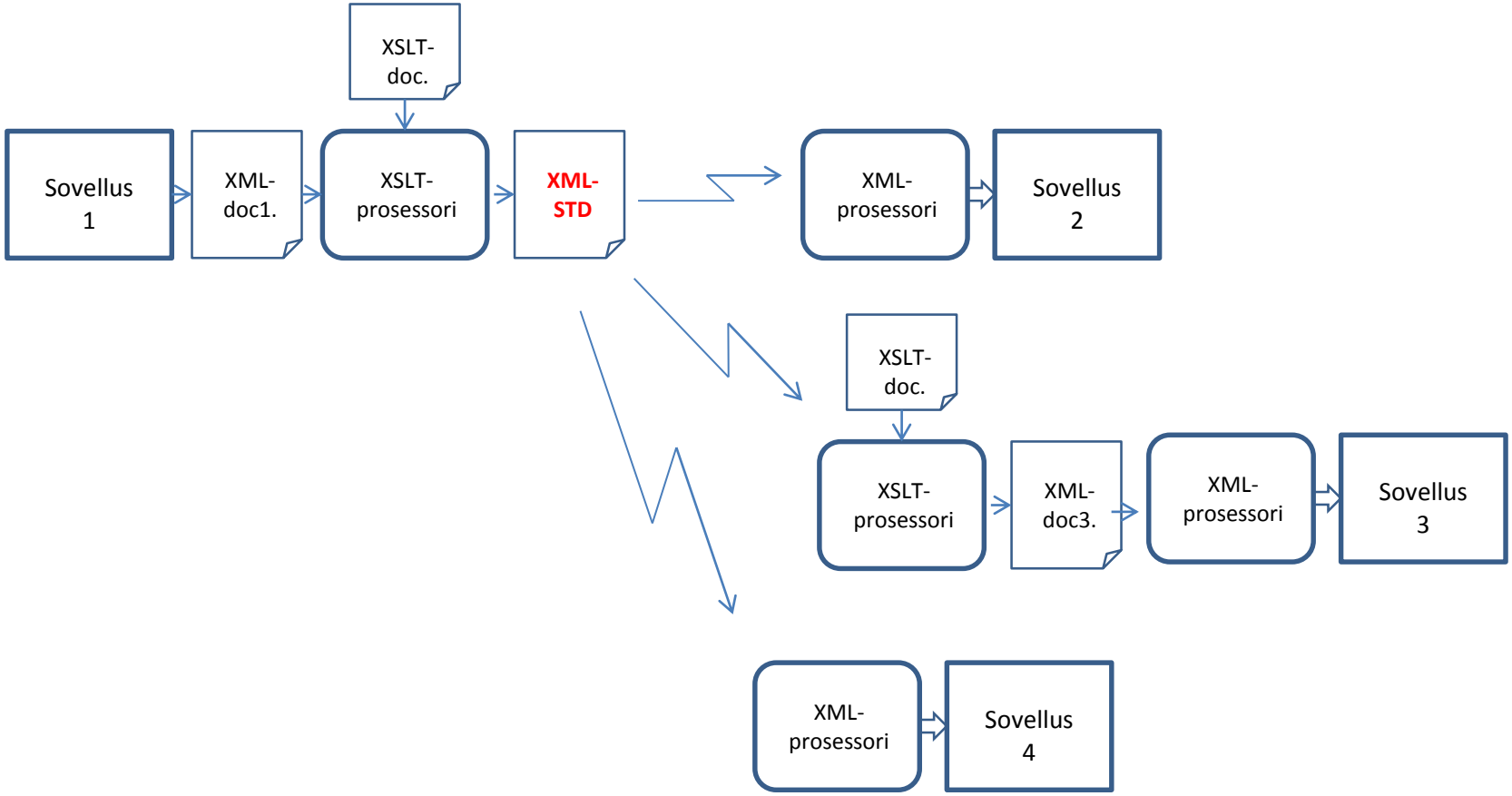


Integraatio-ongelma

- N erillistä organisaatiota, jotka kaikki käyttävät omaa erityistä XML-tiedon esitysmuotoa.
- Jokaisen organisaation täytyy tehdä N-1 erilaista muunnosta vastaanottaessaan muiden lähettämiä xml-dokumentteja (olettaen, että kaikki lähettävät omassa esitysmuodossaan)

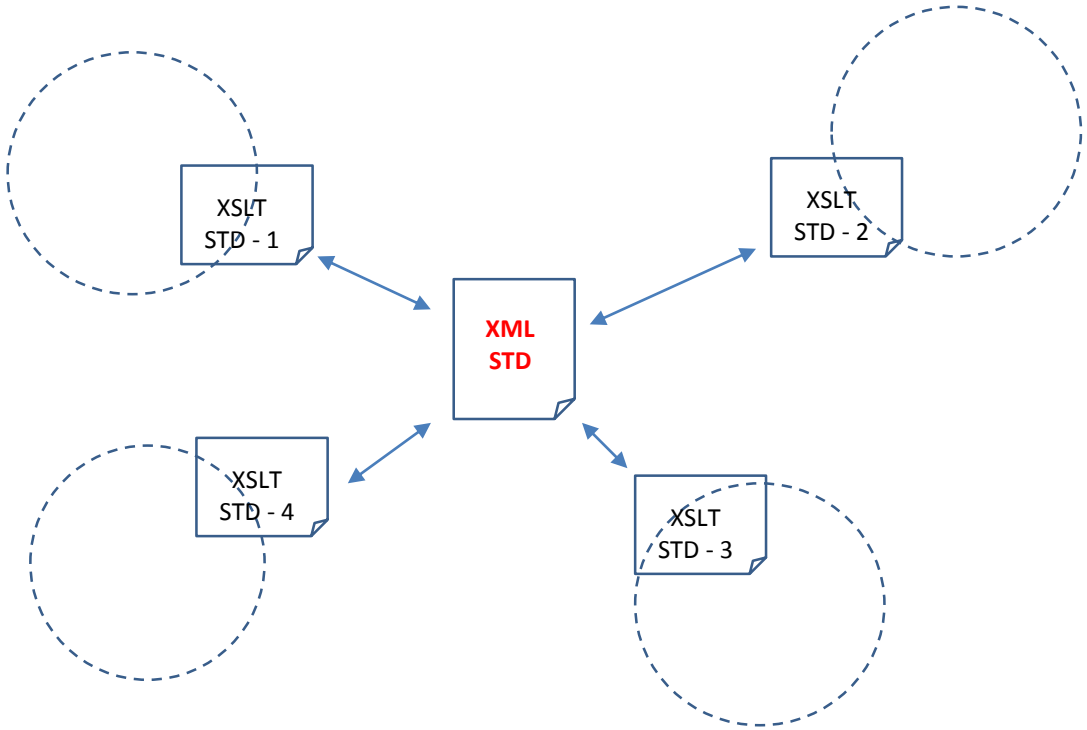


Tietointegraatio – Standardoitu tiedonsiirto



Tietointegraatio – Standardoitu tiedonsiirto

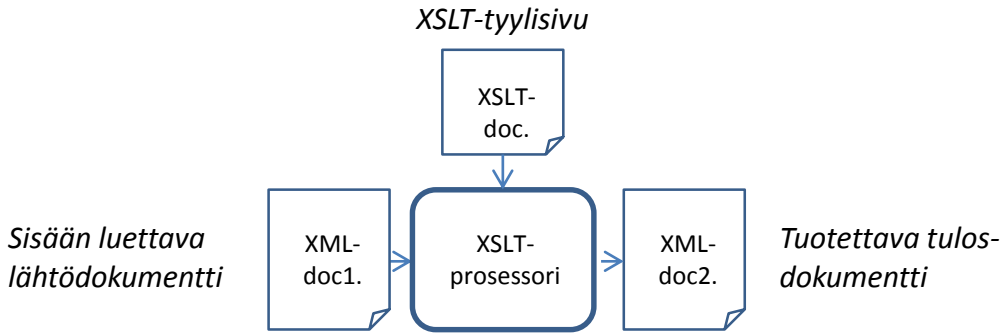
- N erillistä organisaatiota, jotka kaikki käyttävät tiedonsiirrossa standardia XML-esitysmuotoa.
- Jokaisen organisaation täytyy tehdä vain 0-1 erilaista muunnosta sekä vastaanottaessaan että lähettäessään standardeja xml-dokumentteja (muunnoksia ei tarvita lainkaan, jos std-muotoa käytetään myös organisaation sisällä)



XSLT-TYYLISIVU JA PROSESSOINTI

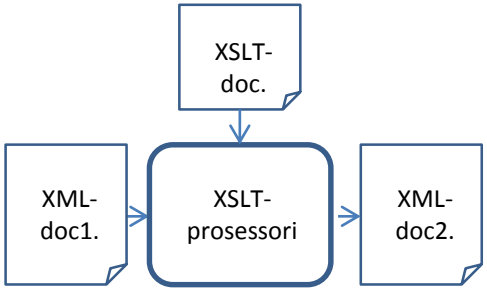
Online xslt tool: http://www.xpathtester.com/xslt

XSLT-prosessointi (1/2)



- XSLT-prosessori muuntaa lähtödokumentin (*doc1.xml*) uuteen muotoon tulosdokumentiksi (*doc2.xml*) XSLT-dokumentin sisältämien sääntöjen ohjaamana.
- Yleensä tavoitteena on lähtödokumentin tietosisällön esittäminen jossain uudessa rakennemuodossa
  - Uuteen dokumenttiin voidaan luoda aivan toisenlainen rakenne uusine elementteineen, attribuutteineen ja tekstisisältöineen
  - Toisaalta uuteen dokumenttiin voidaan myös kopioida lähdedokumentin elementtejä ja attribuutteja tietosisältöineen
- Uuden dokumentin rakenne ja tietosisältö voi siis olla (melkein) minkälainen tahansa
  - Sen ei tarvitse olla edes XML-dokumentti
  - Se voi olla html-dokumentti tai pelkkä tekstidokumentti

XSLT-prosessointi (2/2)



- XSLT-prosessoinnin vaiheet
  1. XSLT-prosessori lukee XML-dokumentin ja XSLT-tyylisivun (*XSLT-stylesheet*)
  2. Se analysoi ja jäsentää XML-dokumentin solmupuuksi (*node-tree*)
  3. Se käy läpi solmupuuta solmu kerrallaan aloittaen juurisolmusta (/)
  4. Ohjeet/säännöt ko. solmun käsittelyyn se hakee XSLT-tyylisivun ko. solmun käsittelyyn tarkoitetusta sapluunasta/mallinteesta (*template*)
  5. Sapluunan *match*-attribuutin XPath-lauseke määrittää sen solmun/solmujoukon, jonka käsittelyyn sapluunan säännöt on tarkoitettu
  6. Prosessori aloittaa sääntöjen lukemisen juuri-sapluunasta (root template), joka on oltava jokaisessa XSLT-dokumentissa: *<xsl:template match="/">*
  7. Säännöt voivat ohjata joko kirjoitusta tulosdokumenttiin tai käsiteltävän solmujoukon jatkoprosessointia ali-sapluunakutsuineen: *<xsl:apply-templates select="sub-node">*

Tyylisivun rakenne (1/2)

- XSLT-tyylisivu koostuu joukosta määrityksiä ja sapluunoita (template)
- Määrityksillä muokataan tyylisivun tuottamaan dokumenttiin liittyviä asioita tai määritellään muuttujia myöhempää käyttöä varten
- Sapluunat sisältävät dokumentin muunnossa käytettävät säännöt, sisään luetun dokumentin käsittelyohjeita ja tuotetun dokumentin elementtejä
  - sana ”sapluuna" kuvaa toimintaa melko hyvin:
    - osa tuotettavan dokumentin elementeistä on määritelty pysyvästi
    - näiden elementtien väliin tuotetaan lisää sisältöä käsittelyohjeiden pohjalta

Tyylisivun rakenne (2/2)

- Tyylisivun kaikki elementit ovat elementin <xsl:stylesheet> sisällä

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

- Tyylisivun tulostusmuoto määritellään <xsl:output> elementillä
  - Metodiksi voidaan määritellä: *xml, html* tai *text*
  - Indent-attribuutilla voidaan tulostus sisentää

```
< xsl:output method="html" indent="yes"/>
```

- Tyylisivujen sisältämät sapluunat määritellään <xsl:template > elementeillä

```
<xsl:template match="Xpath-expression">
```

- Muunnettavan XML-dokumentin alussa voidaan viitataan tyylisivuun prosessointi-ohjeella

```
<?xml-stylesheet type="text/xsl" href="ihmeet_lec2ex1.xsl"?>
```

Juurisapluuna - root template

- XSLT muunnos aloitetaan juurisapluunan (root template) prosessoinnilla.
  - <xsl:template match="/"> … muunnossääntöjä ….</xsl:template >
  - Se on oltava jokaisessa xslt-dokumentissa
  - Se määrittää miten xml-dokumentin juurisolmua prosessoidaan
  - Muunnossääntöjen prosessointi jatkuu viimeiseen juuri-sapluunan sääntöön asti
  - Se voi sisältää muiden sapluunoiden soveltamiskäskyjä
    - <xsl:apply-templates select="xpath-exp"/>
- (Huom: Sapluunoista lisää myöhemmissä kalvoissa)

Esimerkki1: XSLT tyylisivu

html-dokumentin luonti

Ihmeet_lec2ex1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
    <xsl:template match="/">
        <html><head><title>Wonders of the World</title></head>
         <body>
                <h1>Wonders of the World</h1>
         The <xsl:value-of select="ancient_wonders/wonder/name"/>
         is located in <xsl:value-of select="ancient_wonders/wonder/location"/>.
        </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

[http://www.kehogo.com/examples]

Esimerkki1: XSLT tyylisivu

html-dokumentin luonti

XML1: Ihmeet_lec2ex1.xml

```
<?xml version="1.0"?>
<!-- ihmeet_lec2ex1.xml -->
<?xml-stylesheet type="text/xsl" href="ihmeet_lec2ex1.xsl"?>

<ancient_wonders>
  <wonder>
    <name language="English">Colossus of Rhodes</name>
    <location>Greece</location>
  </wonder>
</ancient_wonders>
```

XSLT: Ihmeet_lec2ex1.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
  <xsl:template match="/">
      <html><head><title>Wonders of the World</title></head>
       <body>
            <h1>Wonders of the World</h1>
        The <xsl:value-of select="ancient_wonders/wonder/name"/>
        is located in <xsl:value-of select="ancient_wonders/wonder/location"/>.
       </body>
       </html>
  </xsl:template>
</xsl:stylesheet>
```

tulostusmuoto

template

XSLT-prosessori

XML2: Ihmeet_lec2ex1.html

Esimerkki1: XSLT tyylisivu

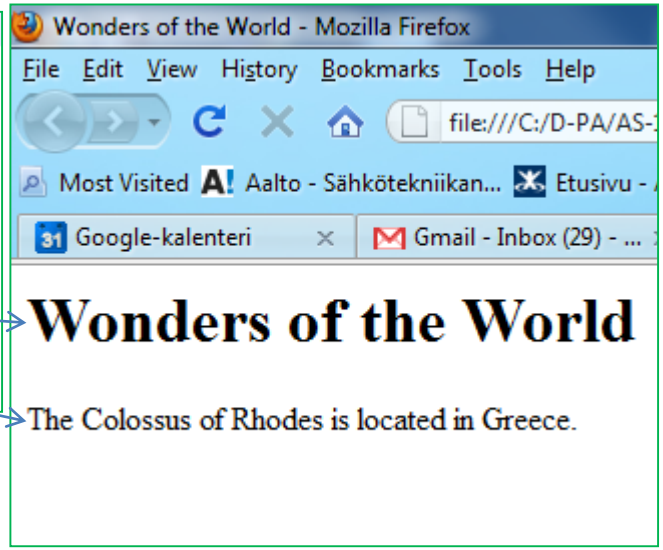
html-dokumentin luonti

- XSL-muunnoksen voi ajaa XML-editorilla, jolloin se tuottaa tuloksena html-tiedoston (ihmeet_lec2ex1.html), jonka voi avata katseltavaksi internet-selaimella
- Internet-selaimet (Firefox, Chrome, IE, jne.) osaavat myös suoraan tehdä XSL-muunnoksen, jos avattavan XML-tiedoston alussa on viittaus XSLT-tiedostoon

Tulosdokumentti
Ihmeet_lec2ex1.html

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Wonders of the World</title>
  </head>
  <body>
    <h1>Wonders of the World</h1>
    The Colossus of Rhodes is located in Greece.
  </body>
</html>
```

Tulosdokumentin
selainnäkymä



http://www.w3schools.com/xsl/default.asp

TEKSTISISÄLTÖJEN KÄSITTELYSÄÄNNÖT

Arvojen poimiminen

- XSLT-tyylisivuissa yleensä poimitaan suuri joukko arvoja sisään luettavasta dokumentista ja sijoitetaan tuotettavaan dokumenttiin
  - Elementtien tekstisisältöjä ja attribuuttien arvoja siirretään tai kirjoitetaan näkyviin
- Arvojen poiminta tapahtuu elementillä
  <xsl:value-of select="xpath-lauseke"/>
  - elementti poimii XPath-lausekkeen mukaisen arvon ja sijoittaa sen tuotettavaan dokumenttiin
  - Huomaa, että Xpath-lausekkeen määrittämän solmujoukon ensimmäisen solmun arvo tulostetaan
  - mikäli arvoa ei ole, elementti ei tuota mitään

Online xpath tool: http://www.xpathtester.com/xpath

Toisto

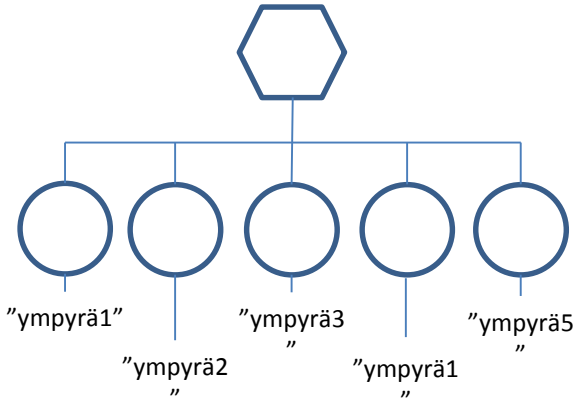
- XSLT sisältää hyvin yksinkertaisen toistorakenteen
  - se voi toistaa tietyn sapluunan osan kaikille tietyllä XPath-lausekkeella valituille solmuille
  - ei ole mahdollisuutta toistaa asioita esimerkiksi viittä kertaa
    - toistojen määrä riippuu aina sisään luettavasta dokumentista
- Toisto toteutaan elementillä <xsl:for-each>
  - elementissä on XPath-lauseke sääntönä ja elementin sisällä oleva XSLT-koodi toistetaan kaikille säännön valitsemille elementeille:

```
<xsl:for-each select="name">
   <xsl:value-of select="@first"/>.
  <xsl:value-of select="@last"/>@aalto.fi
   <br/>
</xsl:for-each>
```

Esimerkki: XML-dokumentti

- ”Polygons_with_size.xml”

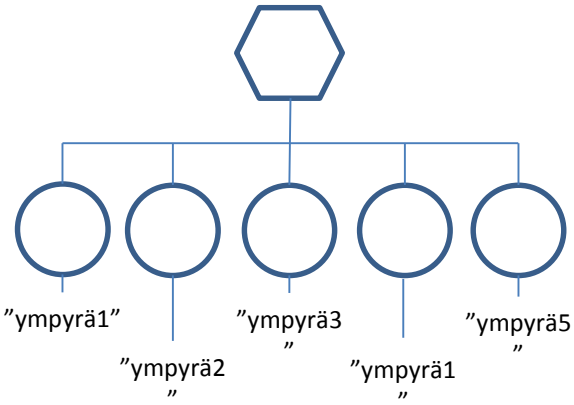
```
<?xml version="1.0" encoding="UTF-8" ?>
<?xml-stylesheet type="text/xsl" href=”to_square.xsl"?>
<polygons>
    <circle color="blue" size=”5">ympyra1</circle>
    <circle color="blue" size=”5">ympyra2</circle>
    <circle color="blue" size="5">ympyra3</circle>
    <circle color="blue" size=”5">ympyra4</circle>
    <circle color="blue" size=”5">ympyra5</circle>
</polygons>
```



Kuva: Lähtödokumentin polygons_doc1.xml rakenne.

xsl:value-of & xsl:for-each

- xsl:value-of
  - Huom: palauttaa vain XPath-solmujoukon ensimmäisen solmun arvon*
- xsl:for-each
  - Mahdollistaa kaikkien solmujen läpikäynnin luupissa




```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!-- XSL Ex1-1: to_square_value-of.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Tr
    <xsl:output method="xml" indent="yes"/>
    <xsl:template match="/">
        <xsl:value-of select="/polygons/circle/@size"/>
    </xsl:template>
</xsl:stylesheet>
```

```xml
<?xml version="1.0" encoding="utf-8"?>5
```

```xml
<?xml version="1.0" encoding="utf-8"?>55555
```

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!-- XSL Ex1-2: to_square_for-each_value-of.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="xml" indent="yes"/>
    <xsl:template match="/">
        <xsl:for-each select="/polygons/circle">
            <xsl:value-of select="@size"/>
        </xsl:for-each>
    </xsl:template>
</xsl:stylesheet>
```

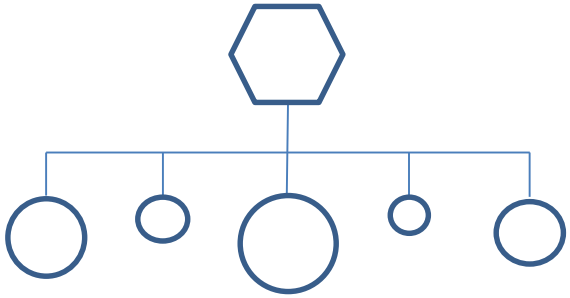
Järjestäminen

- XSLT tarjoaa mahdollisuuden järjestää sisään luettavan dokumentin elementit uuteen järjestykseen ennen sapluunan valintaa
  - käytetään elementtiä <xsl:sort select="..."\>
  - järjestäminen tapahtuu select-lausekkeen tuottamien arvojen pohjalta
    - elementille voi antaa useita lisäattribuutteja järjestämisen ohjaamiseksi
- Järjestäminen tapahtuu joko <xsl:apply-templates\> tai <xsl:for-each\> elementtien sisällä

```
<xsl:apply-templates select="sect1/title">
  <xsl:sort select="text()"/>
 </xsl:apply-templates>
```

- Järjestämiselementtejä voi olla useita peräkkäin

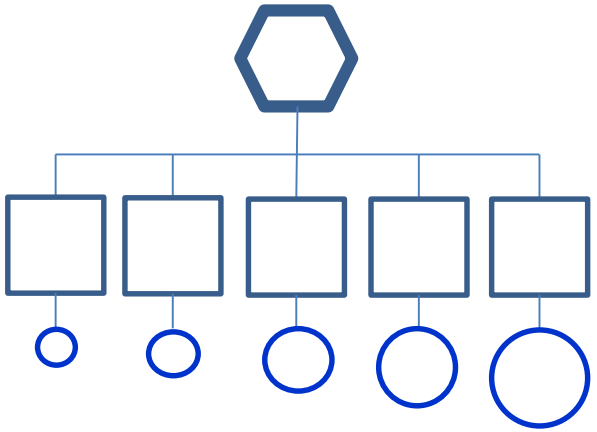
For-each & sort



Kuva 1: Lähtödokumentin polygons_doc1.xml rakenne.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- XSL Ex1: to_square_sort_value-of.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="xml" indent="yes"/>
    <xsl:template match="/">
        <xsl:element name="POLYGONS">
            <xsl:for-each select="/polygons/circle">
                <xsl:sort select="@size" data-type="number" order="ascending"/>
                <xsl:element name="square">
                    <xsl:copy-of select="."/>
                </xsl:element>
            </xsl:for-each>
        </xsl:element>
    </xsl:template>
</xsl:stylesheet>
```

Kuva 2: XSLT-tyylisivu.xsl.



Kuva 3: Tulosdokumentin POLYGONS_doc2.xml rakenne.

Ehdolliset osiot

- XSLT:ssa on kaksi rakennetta ehdollisten osioiden laatimiseksi (IF ja CHOOSE)
  - molemmissa käytetään samaa testirakennetta
  - testinä on XPath-lauseke ja se läpäistään, jos
    - lausekkeessa oleva ehto toteutuu
    - lauseke evaluoituu ei-tyhjäksi joukoksi (ei erillistä ehtoa)
  - ehtoina ovat normaalit =, <, > ja !=
    - HUOM: yleensä vertailumerkit **<** ja **>** pitää kirjoittaa XML-syntaksin takia vastaavilla entiteeteillä < ja > (huom. Kuitenkin Online testerillä* täytyy käyttää < ja >)

*Online xpath tool: http://www.xpathtester.com/xpath

If-lauseke

- If-lauseke on yksinkertaisempi ehdoista
  - sen sisällä oleva mallinteen osa suoritetaan ehdon toteutuessa

```
<xsl:if test="@target &gt; 10">
  <xsl:attribute name="target">
    <xsl:value-of select="@target"/>
  </xsl:attribute>
</xsl:if>
```

HUOM: IF lausekkeessa ei ole else if tai edes else –osaa (toteutetaan käytännössä uusilla if-lausekkeilla)

Choose-lauseke

- Choose-lauseke mahdollistaa usean ehtolausekkeen yhdistämisen
  - vain yksi ehtolausekkeista suoritetaan
    - ensimmäisen onnistuneen testin jälkeen jatketaan choose-lausekkeen jälkeisistä elementeistä
  - lausekkeessa voi olla myös erillinen vaihtoehto tilanteella, jossa yksikään testi ei toteutunut

Choose-esimerkki

Choose

When

Otherwise

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

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <xsl:choose>
        <xsl:when test="price &gt; 10">
          <td bgcolor="#ff00ff">
          <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:when test="price &gt; 9">
          <td bgcolor="#cccccc">
          <xsl:value-of select="artist"/></td>
        </xsl:when>
        <xsl:otherwise>
          <td><xsl:value-of select="artist"/></td>
        </xsl:otherwise>
      </xsl:choose>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
```

http://www.w3schools.com/xsl/default.asp

RAKENTEIDEN LUONTISÄÄNNÖT

Elementtien Kopiointi

- XSLT mahdollistaa myös kopioinnin sisään luettavasta dokumentista tuotettavaan dokumenttiin
  - kopioinnissa dokumenttisirpale (document fragment) kopioidaan sellaisenaan
  - ei tarvita erillisiä sääntöjä jokaiselle osaselle
  - toisaalta ei voida myöskään vaikuttaa lopputulokseen
- Kopiointi on osa sapluunaa
  - kopiointi suoritetaan elementillä <xsl:copy-of>
    - elementti voi määritellä myös XPath-lausekkeen, joka määrää kopioitavat elementit

      <xsl:copy-of select="..."/>
    - jos lauseketta ei määritellä, kopioidaan nykyinen solmu
  - elementillä <xsl:copy> kopioidaan vain nykyinen elementti ilman sen attribuutteja ja lapsia

Elementtien ja attribuuttien luonti

- XSLT mahdollistaa elementtien luonnin dynaamisesti
  - käytetään elementtiä <xsl:element>

```
<xsl:element name=”new-element-name">
  <xsl:apply-templates/>
</xsl:element>
```

- Vastaavasti myös attribuutteja voidaan luoda lennossa
  - Käyttäen elementtiä <xsl:attribute>
  - attribuutti liitetään viimeksi määritellyn elementin sisällä
  - ESIM1:

```
<a>
  <xsl:attribute name="href">
    <xsl:value-of select="substring(@src, 2)”/>
  </xsl:attribute>
  <xsl:value-of select="@title"/>
</a>
```

ESIM2:

```
<xsl:template match="circle“>
   <xsl:element name="square">
      <xsl:attribute name="color">red</xsl:attribute>
   </xsl:element>
</xsl:template>
```

http://www.w3schools.com/xsl/default.asp

SAPLUUNAT (TEMPLATES)

Sapluuna (Template)

- Sapluuna koostuu kahdesta osasta
  1. sapluunan laukaiseva sääntö
  2. sapluunan sisältö
- Sääntö määritetään XPath-lausekkeella
- Sapluunan sisällön rakenteen määrää tuotettavan dokumentin rakenne ja sisältö
  - sapluuna voi tuottaa uusia elementtejä ja attribuutteja (ja tekstisisältöä) tuotettuun dokumenttiin
  - sapluuna voi valita sisään luetun dokumentin elementtejä
    - kutsua uusia mallinteita/templateja elementtien pohjalta
    - käsitellä elementit itse
  - sapluuna voi sisältää ehtoja ja toistoja

http://www.w3schools.com/xsl/default.asp

Sapluunoiden valinnan ohjaus

- Sapluuna voi käynnistää sapluunan haun mille tahansa XML-dokumentin osalle (solmulle tai solmujoukolle), joka määritellään Xpath-lausekkeella

  <xsl:apply-templates select=”Xpath-lauseke”/>
- Prosessori käsittelee solmut dokumenttijärjestyksessä (document order)
- Myös attribuutti, tekstisirpale tai muu XML-dokumentin osa voidaan valita
  - yleisin tapaus on käsitellä nykyisen solmun alla (sisällä) olevat solmut:
    <xsl:apply-templates/> (*)
  - voidaan myös valita vain tietyt solmut käsittelyyn:
    <xsl:apply-templates select="name|street|city|postal-code"/>
  - valinta voi kohdistua muualla oleviin solmuihin:
    <xsl:apply-templates select="//distant"/>
  - Valinta voidaan tehdä myös ehdolliseksi

    <xsl:apply-templates select="/ancient_wonders/wonder[@build &lt 800]"/>

http://www.w3schools.com/xsl/xsl_apply_templates.asp

(*) Huom: Oletus-sapluuna:
Jos kutsutaan sapluunoita ilman select-valintaa, prosessori hakee kullekin solmulle sopivimman löytämänsä sapluunan. Jos sapluunaa ei löydy käytetään oletus-sapluunaa, joka tulostaa elementtien tekstisisällöt.

Sapluunoiden valinnan ohjaus

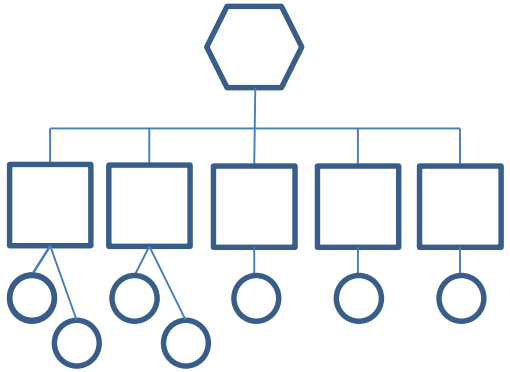
- Huomaa, että xslt-prosesori käy läpi kaikki Xpath-lausekkeella (select) valitut solmut kutsuen niihin sopivaa sapluunaa (match) ja toteuttaa sapluunan sisältämät toiminnot (tulosdokumenttiin kirjoittamisen) jokaiselle solmujoukon solmulle
  - Eli *xsl:for-each* luuppeja ei tarvitse erikseen kirjoittaa
- Esimerkki:
  - Select-lauseke valitsee sapluunan <xsl:template match=”circle”>
  - Tämän templaten sääntöjä sovelletaan kaikkien *square*-elementtien kaikkiin *circle*-elementteihin (7 kpl)

```
<xsl:template match=”/">
…
   <xsl:apply-templates select="/polygons/square/circle"/>
</xsl:template>
```

```
<xsl:template match=”square">
…
</xsl:template>
```

```
<xsl:template match=”circle">
…
</xsl:template>
```

match



http://www.w3schools.com/xsl/default.asp

EDISTYNEET PIIRTEET

Edistyneemmät piirteet

- Edellä esitelyillä toiminnoilla pystyy muuntamaan XML-dokumentin muodosta toiseen
  - keinot kuitenkin loppuvat, jos sisään luetun dokumentin elementtejä pitää käsitellä useasti tai lopputulos riippuu luetun dokumentin attribuuttien arvoista
  - vastaavasti XSLT-tyylisivun toimintaa saatetaan haluta ohjata ulkopuolelta
- Ratkaisuja on useita erilaisia
  - toisto
  - muuttujien käyttö
  - ehdolliset osiot sapluunoissa
  - erilaisten moodien käyttö
  - sapluunoiden nimeäminen ja kutsuminen ilman säännön laukeamista

[Frosterus]

Muuttujat (1/2)

- Tyylisivulla voi olla käytössä muuttujia
  - muuttujat asetetaan elementillä <xsl:variable name="..."> ja niiden arvoa ei voi muuttaa myöhemmin (ks. alla)
  - muuttujat ovat käytössä koko tyylisivulla
  - muuttujiin viitataan syntaksilla $nimi
    - muuttujiin voidaan viitata myös {$nimi} sapluunan elementtien sisällä, vrt. attribuutit
- Muuttujat ovat näppäriä laadittaessa helposti säädettäviä tyylisivuja
  - muokattavat tiedot sijoitetaan muuttujiin, joita säätämällä tyylisivun toiminta tai tuotettava dokumentti muuttuu
- Vaikka muuttujan arvoa ei voi muuttaa, se voidaan määrittää uudelleen alemmalla tasolla
  - uusi arvo piilottaa aiemman arvon
  - aiempi arvo on käytössä palattaessa takaisin ylemmälle tasolla

Muuttujat (2/2)

- Example:
  - <xsl:variable name="domain">aalto.fi</xsl:variable>
  - ...
  - <xsl:attribute name="href">http://www.<xsl:value-of select="$domain"/>/</xsl:attribute>
  - ...
  - <info domain="{$domain}">Aalto</info>

- Example2:
  - <xsl:variable name="bodyTextSize">10pt</xsl:variable>
  - …
  - <xsl:template match="product">
  -
  - <xsl:apply-templates/>
  - </xsl:template>

Harjoitustehtävä 2 Vinkkejä

- Suositus: käytä *template*-tekniikkaa mahdollisimman paljon mieluummin kuin *for-each*-luuppeja, koska se on XSLT:n keskeinen tekniikka
- Ensimmäinen juuri-template kutsuu (xsl:apply-templates) ali-templatea, joka ’match:ää’ ’*assembly_line*’ elementtiin.
- Tämä ali-template kutsuu edelleen ali-templateja, jotka sisältävät säännöt ’*assembly_line*’-elementin lapsielementtien käsittelyyn ja niin edespäin…

```
<?xml version="1.0" encoding="utf-8"?>
<!-- U2E2_1: Students' version of aml_transform.xsl. Extend this stub file -->
<!-- U2E2_1: Transforming source file production_line.xml to AML format. -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="xml" indent="yes" encoding="utf-8" media-type="text/xml"/>
    <xsl:variable name="classlibpath">ProdLineSystemUnitClassLib</xsl:variable>

    <!-- Main entry point -->
    <xsl:template match="/">
        <xsl:element name="AutomationMLLibrary">
            <xsl:apply-templates select=”assembly_line” />
        </xsl:element>
    </xsl:template>

    <!-- TOTEUTA TÄHÄN MUUNNOKSEN TEMPLATET OHJEIDEN MUKAAN -->

</xsl:stylesheet>
```

Lähdeviitteet

- Kurssikirja:
  - XSLT-kieleen liittyvät luvut 2, 12 ja 13 kirjasta: Goldberg, K.,H.2009. XML-Visual QuickStart Guide. 2.e. Peachpit Press XML. : Introduction, Ch1, Ch2, Ch3, Ch4 ja Ch9-11, Ch12-13, Ch14-15.
  - Goldberg kirjan esimerkit: http://www.kehogo.com/examples
  - XSLT-kieleen liittyvät luvut 3, 8 kirjasta: E-book: Beginning XML, (5th ed.) by Fawcett, J., et al., John Wiley & Sons, Inc. Ch 1-2, **Ch3**, Ch5, Ch7, **Ch8**.
- Muita lähteitä:
  - Tutoriaalit: http://www.w3schools.com/
  - W3C:n Määrittelyt:
    - http://www.w3.org/XML/Core/
    - http://www.w3.org/Style/XSL/
    - http://www.w3.org/XML/Schema#dev
- Tools
  - Online tool: xpath, xslt, schema validate: http://www.xpathtester.com
  - Online tool: Regular Expression Tester: http://www.freeformatter.com/regex-tester.html
  - Command line tool: Xmlstarlet command line XML toolkit:
    - Download for windows: http://xmlstar.sourceforge.net/download.php
  - XML editors
    - MindFusion XML Viewer on Ilmainen editori. Sopii ainakin XSLT-muunnoksien harjoitteluun.