



Interneto technologijos

XSLT
Instrukcijos



Instrukcijos

- Prisiminimui:
 - Šablono viduje gali būti:
 - Statinis tekstas,
 - Instrukcijos
- Toliau aptarsim šias instrukcijas:
 - `xsl:value-of`
 - `xsl:text`
 - `xsl:if`
 - `xsl:choose`
 - `xsl:for-each`
 - `xsl:sort`
 - `xsl:number`



Instrukcija `xsl:value-of`

- Skirta XML dokumento mazgų reikšmių spausdinimui

```
<xsl:value-of select=" <XPath išraiška> " />
```

- XPath išraiškos rezultatas paverčiamas į tekstą kviečiant XPath funkciją `string()`
 - Jei išraiška grąžina mazgų aibę, imamas tik pirmas aibės mazgas
 - Jei tas mazgas yra žymė, tai imama visų jos įpėdinių tekstinių mazgų konkatencija
 - Atributui imama jo reikšmė tarp kabučių

Instrukcija `xsl:text`

- Skirta teksto išvedimui:

```
<xsl:text>Kažkoks tekstas</xsl:text>
```

- Gali būti naudinga, jei tarp dviejų tekstinių mazgų jums reikia įterpti tarpą (tarpai yra automatiškai išmetami), pvz.:

```
<xsl:template match="asmuo">  
  <xsl:value-of select="vardas" />  
  <xsl:text> </xsl:text>  
  <xsl:value-of select="pavardė" />  
</xsl:template>
```

Jei šios eilutės nebūtų, tai tarp vardo ir pavardės nebūtų tarpo



Instrukcija `xsl:if`

■ Sąlyginis sakinyys

```
<xsl:if
  test = " <XPath išraiška> " >
  <!-- Turinys: šablonas -->
</xsl:if>
```

■ XPath išraiškos rezultatas verčiamas į loginį tipą su funkcija `boolean()`

- Jei rezultatas `true`, tai įterpiamas turinyje nurodytas šablonas

```
<xsl:for-each select="knyga">
  <xsl:value-of select="pavadinimas" />
  <xsl:if test="not(position()=last())">
    ,
  </xsl:if>
</xsl:for-each>
```



Instrukcija xsl:choose

- C/C++/Java switch analogas

```
<xsl:choose>
  <xsl:when test='...'>
    ...
  </xsl:when>
  <xsl:when test='...'>
    ...
  </xsl:when>
  <xsl:otherwise>
    ...
  </xsl:otherwise>
</xsl:choose>
```



Instrukcija `xsl:for-each`

- Skirta dirbti su ciklais

```
<xsl:for-each
  select = " <XPath išraiška> "
  <!-- Turinys: (xsl:sort*, šablonas) -->
</xsl:for-each>
```

- XPath išraiška privalo grąžinti mazgų aibę,
 - jos dydis tampa konteksto dydžiu (grąžinamas funkcija `last()`),
 - kiekvienas aibės mazgas paeiliui tampa einamuoju mazgu, kurio eilės numerį aibėje grąžina funkcija `position()`



Instrukcija xsl:for-each

■ Pavyzdys:

```
<xsl:template match="knyguSarašas">
  <xsl:for-each select="knyga">
    <xsl:value-of select="position()" />
    .
    <xsl:value-of select="pavadinimas" />
  </xsl:for-each>
</xsl:template>
```




Instrukcija `xsl:sort`

- Skirta mazgų aibės rūšiavimui

```
<xsl:sort
  select = " <XPath išraiška> "
  data-type = { "text" | "number" }
  lang = " <kalba> "
  order = { "ascending" | "descending" } />
```

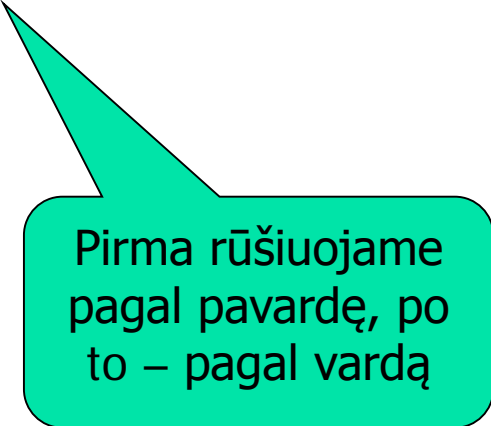
- Šią instrukciją leidžiama naudoti tik instrukcijų `xsl:apply-templates` arba `xsl:for-each` viduje
- XPath išraiškos rezultatas paverčiamas į eilutę kviečiant funkciją `string()`, rezultatas tampa *rūšiavimo raktu* (pagal ką bus rūšiuojama). Pagal nutylėjimą yra `select="."`
- `order` nurodo, ar rūšiavimas yra didėjimo, ar mažėjimo tvarka
- `data-type` nurodo, ar rūšiavimas yra leksikografinis (pagal nutylėjimą), ar skaitmeninis
- `lang` nurodo, pagal kokią kalbą yra vykdomas leksikografinis rūšiavimas (pvz.: `"lt-LT"`)



Instrukcija xsl:sort

■ Pavyzdys:

```
<xsl:template match="knyguSarašas">
  <xsl:apply-templates select="knyga">
    <xsl:sort select="autorius/pavardė"/>
    <xsl:sort select="autorius/vardas"/>
  </xsl:apply-templates>
</xsl:template>
```



Pirma rūšiuojame
pagal pavardę, po
to – pagal vardą



Instrukcija `xsl:number`

- Skirta sąrašo numeravimui

```
<xsl:number  
  value = " <XPath išraiška> "  
  format = " <formatas> " />
```

- XPath išraiškos rezultatas paverčiamas į skaičių su XPath funkcija `number()`
 - Jei `value` atributas nenurodytas, tai įterpiamas skaičius, atitinkantis einamojo mazgo poziciją dokumente
 - Galimi formatai: 1, a, A, i, I

```
<xsl:template match="knygųSąrašas">  
  <xsl:for-each select="knyga">  
    <xsl:sort select="." />  
    <p>  
      <xsl:number value="position()" format="1. " />  
      <xsl:value-of select="pavadinimas" />  
    </p>  
  </xsl:for-each>  
</xsl:template>
```



Papildomos XPath funkcijos

- *node-set* **current()**
 - grąžina mazgų aibę iš vieno elemento – šiuo metu šablone apdorojamo einamojo mazgo
- Pavyzdys: einamasis mazgas yra žymė su atributu `ref`, kuris turi nuorodą į žymės `autorius` atributo `id` reikšmę. Norime surasti tą autorių, į kurį rodo `ref`:
 - `//autorius[@id=current()]/@ref`
- Taip neveiks:
 - `//autorius[@id=../@ref]`
 - čia `"."` ims ne einamosios žymės atributą `ref`, o žymės `autorius` atributą `ref`

XML dokumento susiejimas su transformacija

■ XML dokumentas:

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

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

```
<knyga>
```

```
  <pavadinimas>XSLT</pavadinimas>
```

```
  <autorius>John Smith</autorius>
```

```
</knyga>
```

Kelias iki failo, kuriame yra transformacija (praplėtimas nebūtinai turi būti "xslt").

Dabar užtenka atidaryti šitą XML dokumentą su naršykle, ir transformacija bus įvykdyta automatiškai.

XML dokumentų su vardų sritimis transformacijos

- Tarkim turim XML dokumentą:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="transf.xslt" type="text/xsl"?>
<knyga xmlns="http://www.mif.vu.lt/~donatas">
  <pavadinimas>XSLT</pavadinimas>
  <autorius>John Smith</autorius>
</knyga>
```

Vardų sritis pagal
nutylėjimą

- Transformacijoje visos XPath išraiškos privalo naudoti vardų sritis!

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:b="http://www.mif.vu.lt/~donatas">
  <xsl:template match="b:knyga">
    <h1><xsl:value-of select="b:pavadinimas"/></h1>
  </xsl:template>
</xsl:stylesheet>
```

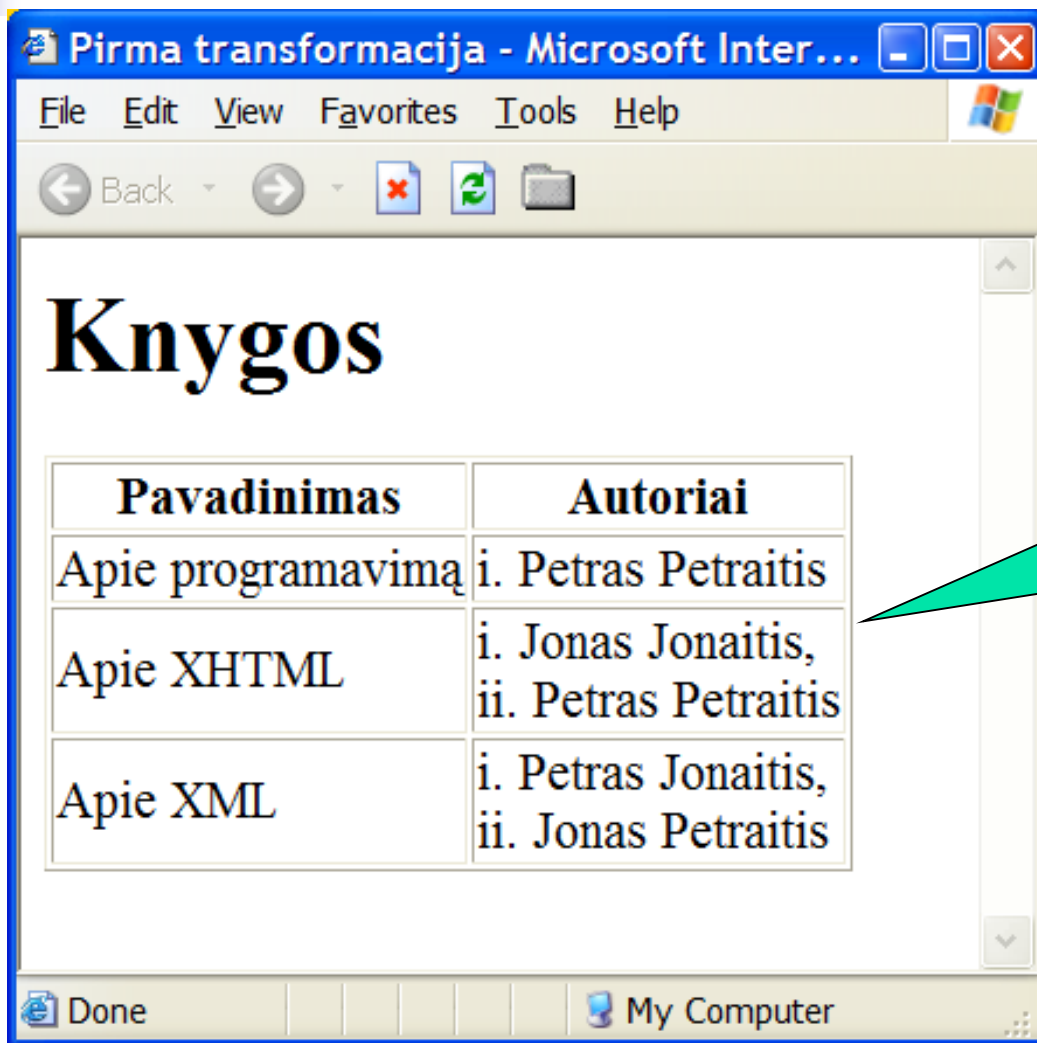
Transformacijos į XHTML pavyzdys – XML dokumentas

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl"
  href="Transformacija.xslt" ?>

<knyguSarasas
xmlns="http://www.mif.vu.lt/~donatas">
  < autoriai>
    < autorius id="a1">
      < vardas>Jonas</vardas>
      < pavardė>Jonaitis</pavardė>
    </ autorius>
    < autorius id="a2">
      < vardas>Petras</vardas>
      < pavardė>Petraitis</pavardė>
    </ autorius>
    < autorius id="a3">
      < vardas>Petras</vardas>
      < pavardė>Jonaitis</pavardė>
    </ autorius>
  </ autoriai>
```

```
<knygos>
  <knyga>
    < autorius ref="a2"/>
    < pavadinimas>Apie
programavimą</pavadinimas>
  </knyga>
  <knyga>
    < autorius ref="a2"/>
    < autorius ref="a1"/>
    < pavadinimas>Apie
XHTML</pavadinimas>
  </knyga>
  <knyga>
    < autorius ref="a3"/>
    < autorius ref="a1"/>
    < pavadinimas>Apie
XML</pavadinimas>
  </knyga>
</knygos>
</knyguSarasas>
```

Norimas rezultatas



Norime kad būtų:
Romėniška numeracija,
 autorių rūšiavimas

Transformacija - antraštė

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:d="http://www.mif.vu.lt/~donatas">

  <xsl:template match="/">
    <html>
      <head>
        <meta http-equiv="Content-Type"
          content="text/html; charset=UTF-8" />
        <title>Pirma transformacija</title>
      </head>
      <body>
        <xsl:apply-templates select="d:knygųSarašas/d:knygos"/>
      </body>
    </html>
  </xsl:template>
```

Į šią vietą bus
įterptas kitų
šablonų darbo
rezultatas

Einam tiesiai prie
knygų



Transformacija – lentelės antraštė

```
<xsl:template match="d:knygos">
  <h1>Knygos</h1>

  <table border="1">
    <tr>
      <th>Pavadinimas</th>
      <th>Autoriai</th>
    </tr>
    <xsl:apply-templates />
  </table>

</xsl:template>
```



Transformacija – knygos

```
<xsl:template match="d:knyga">
  <tr>
    <td><xsl:value-of select="d:pavadinimas" /></td>
    <td>
      <xsl:for-each select="d:autorius">
        ...
        <xsl:value-of
          select="//d:autorius[@id=current()/@ref]/d:vardas"/>

        <xsl:text> </xsl:text>

        <xsl:value-of
          select="//d:autorius[@id=current()/@ref]/d:pavardė"/>
        ...
        <br />
      </xsl:for-each>
    </td>
  </tr>
</xsl:template>
```



Priedam rūšiavimą, numeravimą

```
<xsl:for-each select="d:autorius">
  <xsl:sort select="//d:autorius[@id=current()/@ref]/d:pavardė"/>
  <xsl:sort select="//d:autorius[@id=current()/@ref]/d:vardas" />

  <xsl:number value="position()" format="i. " />

  <xsl:value-of
    select="//d:autorius[@id=current()/@ref]/d:vardas" />
  <xsl:text> </xsl:text>
  <xsl:value-of
    select="//d:autorius[@id=current()/@ref]/d:pavardė" />

  <xsl:if test="position() != last()">
    <xsl:text>,</xsl:text>
  </xsl:if>
  <br />
</xsl:for-each>
```