XRG
XML Web Engineering research group

XML Technologies

XSLT

An Empty XSLT Script

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

  <xsl:output method="xml" indent="yes"/>

</xsl:stylesheet>
```

XSLT Templates

- Unnamed

```
<xsl:template match="xpath_expression(s)">
   ...
</xsl:template>
```

- Named

```
<xsl:template name="template_name">
   ...
</xsl:template>
```

Implicit XSLT Templates

- Matching root node

```
<xsl:template match="/">
  <xsl:apply-templates />
</xsl:template>
```

- Matching element nodes

```
<xsl:template match="*">
  <xsl:apply-templates />
</xsl:template>
```

Implicit XSLT Templates

- Matching text nodes

```
<xsl:template match="text()">
  <xsl:value-of select="." />
</xsl:template>
```

- Matching attribute nodes

```
<xsl:template match="@*">
  <xsl:value-of select="." />
</xsl:template>
```

`<xsl:apply-templates select="xpath" />`	Application of templates on nodes selected using `xpath` (if `select` is not specified, we apply the templates on child nodes)
`<xsl:value-of select="xpath"/>`	Output of values specified using `xpath`
`<xsl:text>…</xsl:text>`	Construction of a text node
`<xsl:attribute name="str">` … `</xsl:attribute>`	Construction of an attribute
`<xsl:element name="str">` … `</xsl:element>`	Construction of an element
`<element-name atr="str">` … `</element-name>`	Construction of an element `element-name`, having attribute `atr` with value `str`
`<xsl:for-each select="xpath">` … `</xsl:for-each>`	Iteration
`<xsl:if test="xpath">` … `</xsl:if>`	Condition (without else branch)

`<xsl:choose>`
`<xsl:when test="xpath">…</xsl:when>`
…
`<xsl:otherwise>` … `</xsl:otherwise>`
`</xsl:choose>`
	Generalization of condition (with else branch)

`<xsl:variable name="var" select="xpath" />`	Variable, accessible via `$var`.
`<xsl:param name="var" select="xpath" />`	Parameter

Modes of Templates

- Template definition

```
<xsl:template match="aaa" mode="overview">
...</xsl:template>

<xsl:template match="aaa" mode="detail">
...</xsl:template>
```

- Template calling

```
<xsl:template match="/">
 <xsl:apply-templates select="aaa" mode="overview" />
</xsl:template>
```

```
<xsl:template match="xyz">
<xsl:apply-templates select="aaa" mode="detail" />
</xsl:template>
```

Named Templates with Parameters

- Template definition

```
<xsl:template name="template">
  <xsl:param name="parameter" />
  ...
  <!-- example of usage of parameter -->
  <xsl:value-of select="$parameter" />
</xsl:template>
```

- Template calling

```
<xsl:call-template name="template">
  <xsl:with-param name="parameter"
                  select="xpath expression" />
</xsl:call-template>
```

Common Errors

- Don’t forget that we have implicit templates

```
<xsl:stylesheet>
  <xsl:template match="aaa">
    <xsl:value-of select="@attr" />
  </xsl:template>
</xsl:stylesheet>
```

- It does not print only the values of attribute `attr` of element `aaa`!!!

```
<xsl:template match="node()" />
```

Common Errors

- Don’t forget how `<xsl:apply-templates />` works
  - It moves the processing to child nodes, but not attributes
    - XPath axis child::
- To move to attributes we must use:

```
<xsl:apply-templates select="@*"/>
```