

# Basic XSL-Transformation Details

- syntax
- xsl tags
- output methods

# XSLT Syntax

XSLT is an XML document, as such, it begins with the XML prolog:

```
<?xml version="1.0"?>
```

# XSLT Syntax

The root element of the style sheet is:

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

# XSLT Syntax

The document root is represented with a slash symbol. The template example matching the root element is :

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

# `<xsl:apply-templates>`

**With `<xsl:apply-templates>` it is possible to define a set of nodes of the source document to be processed. This element, which is an instruction, selects a template rule for each node.**

`<xsl:value-of>`

To write text (the string value of an expression) to the result tree, use the instruction `<xsl:value-of>`.

`<xsl:value-of>` is the most common way to write text to the result tree and is used in most style sheets.

# `<xsl:for-each-instruction>`

To repeat an operation several times, use `<xsl:for-each-instruction>`. This for-each loop allows use to manipulate sets and their members.

# Output Methods

**Output methods determine the output tree type.**

**The three most common settings are:**

**"html", "xml", and "text".**

**XML is the default output method.**

# Output Methods-HTML

```
<?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="/PLANETS">  
  <HEAD>  
    <TITLE>  
      The Planets Table  
    </TITLE>  
  </HEAD>
```

# Output Methods-XML

```
<?xml version="1.0"?>  
<xsl:stylesheet version="1.0"  
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  
  <xsl:output method="xml"/>  
  <xsl:template match="*">  
    <xsl:copy>  
      <xsl:apply-templates/>  
    </xsl:copy>  
  </xsl:template>  
</xsl:stylesheet>
```

# Output Methods-text

```
<?xml version="1.0"?>  
<xsl:stylesheet version="1.0"  
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  
  <xsl:output method="text"/>  
  <xsl:strip-space elements="*" />  
<xsl:template match="/PLANETS">
```