ZENTRUM FÜR INFORMATIONSMODELLIERUNG AUSTRIAN CENTRE FOR DIGITAL HUMANITIES

i|d|e

KARL-FRANZENS-UNIVERSITÄT GRAZ
UNIVERSITY OF GRAZ

UNI GRAZ

Summer School “Digitale Edition”

Tag 4: Werkzeuge der Textpublikation

Roman Bleier und Carina Koch, ZIM -ACDH

INZ ÖAW ÖSTERREICHISCHE AKADEMIE DER WISSENSCHAFTEN

FWF Der Wissenschaftsfonds.

UNIVERSITÄT SALZBURG

Programm

- XSLT, was ist das und wie funktioniert es?
- Einfaches Transformationsszenario
- XSL Transformation mit oXygen?
- Übung 1: oXygen Übung
- XSLT Elemente: Grundlagen
- Übung 2: TEI zu XHTML am Beispiel einer Postkarte
- TEI Frameworks

XSLT, was ist das und wie funktioniert es?

Extensible Stylesheet Language Transformation

W3C recommendation: http://www.w3.org/Style/XSL/

W3C-Standard seit 1999, aktuell: XSLT 3.0

XSLT ist Teil der XSL Sprachfamilie:

1. XSLT: Transformationssprache für Text und XML Dokumente
2. XPath: Adressierung von Teilen eines XML Dokuments
3. XSL-FO: Darstellungs- und Formatierungssprache

XSLT, was ist das und wie funktioniert es?

Vorgesehener Workflow

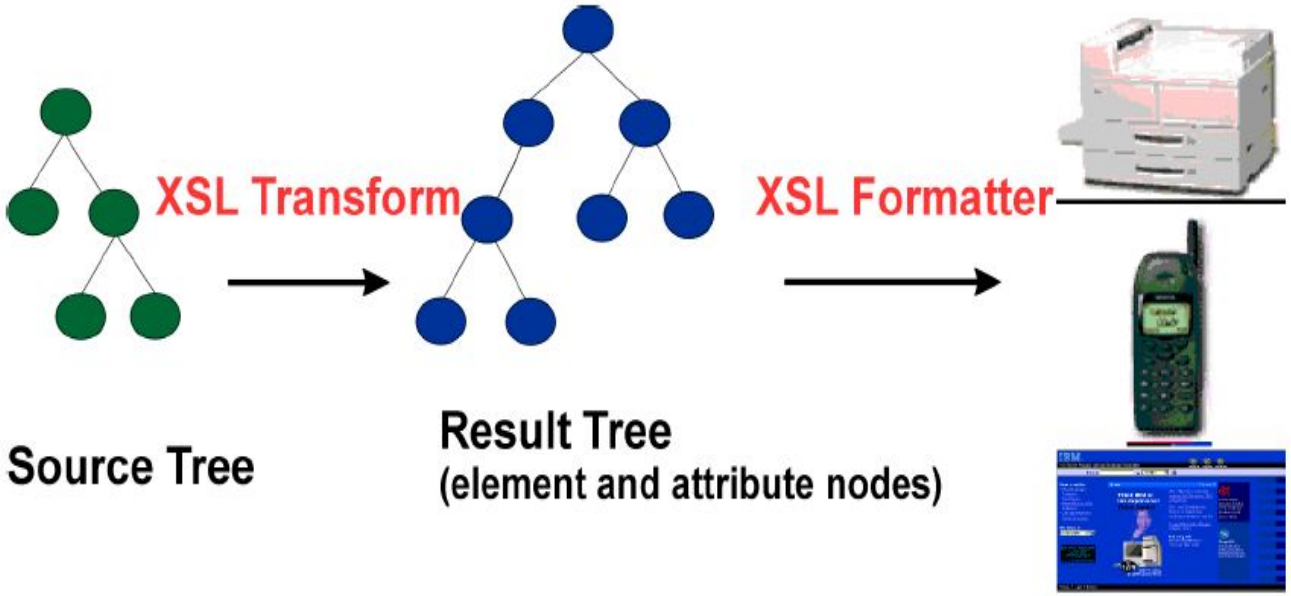
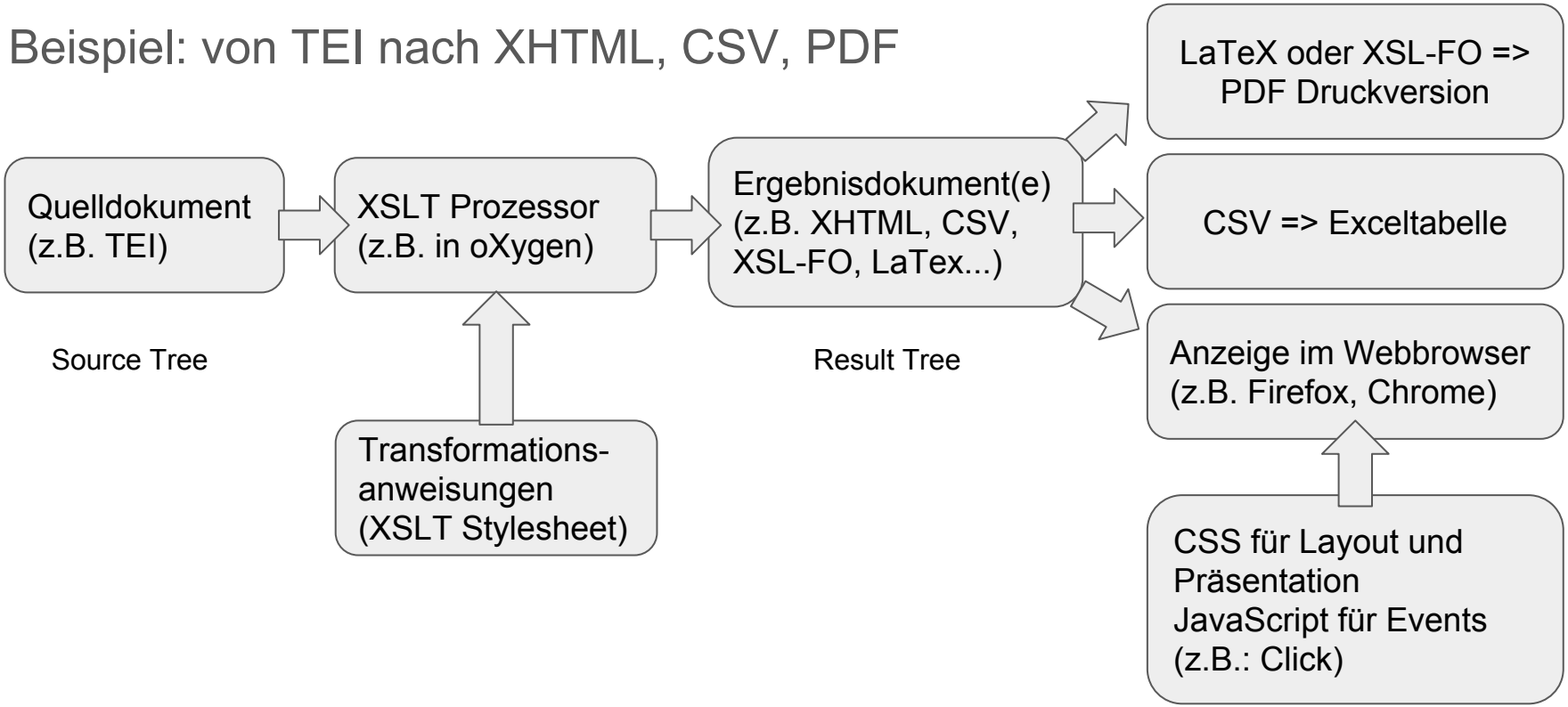


Image from the w3c recommendations: http://www.w3.org/TR/xsl/

XSLT, was ist das und wie funktioniert es?

Beispiel: von TEI nach XHTML, CSV, PDF



XSLT, was ist das und wie funktioniert es?

Umformung der XML Struktur
(Source Tree - Result Tree):

- Treffen einer Auswahl, nur Teile des Quelldokuments
- Einbinden von Daten aus anderen XML Dokumenten
- Änderung von Element- und Attributnamen
- Änderung der Reihenfolge
- Änderung der XML Verschachtelung

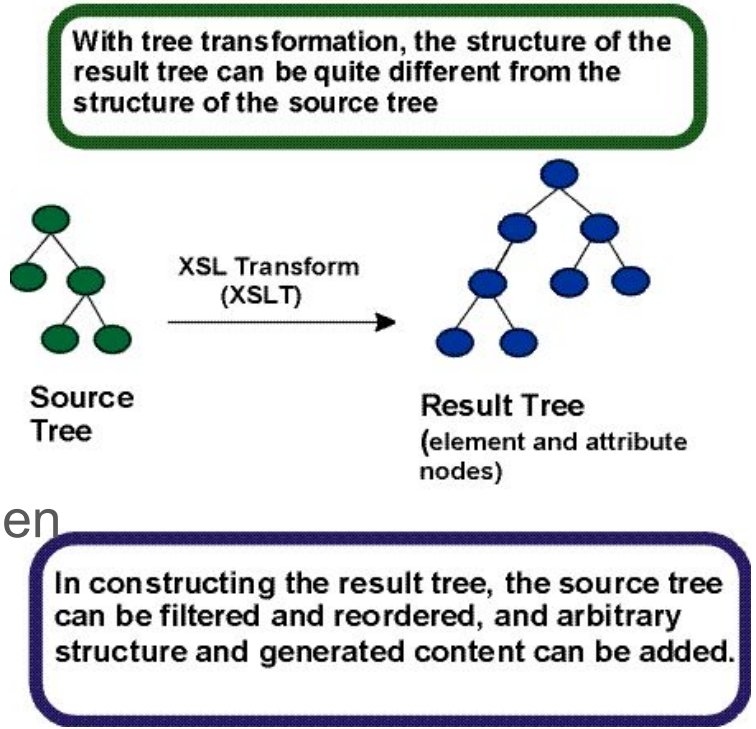


Image from the w3c recommendations:
http://www.w3.org/TR/xsl/

XSLT, was ist das und wie funktioniert es?

Umformung der XML Struktur und Einbindung von weiteren Ressourcen:

- Umformung der XML Struktur: TEI => XHTML
- Einbauen von Hyperlinks, Grafiken, Audio
- Präsentation: Einhängen von CSS
- Events: Einhängen von JavaScript

XSLT, was ist das und wie funktioniert es?

Anwendungsbeispiele:

- Hilfe beim eigenen Arbeiten mit XML (z.B. automatisierte Überarbeitung einer Transkription, Erstellung einer Leseansicht, nachträgliches Einfügen von Elementen, Attributen, Daten)
- Webpublikation von XML Inhalten (z.B. GAMS, TEI Boilerplate, Versioning Machine)
- Generierung von CSV für weitere statistische Verarbeitung
- Generierung von PDF für Druckversionen

Einfaches Transformationsszenario

“The basic operation can be summarized as follows: when a node in the source matches a rule's pattern, the content of that rule is created in the result tree. Once you grasp this basic operation, the overall XSLT processing model is easy to understand. Given a source tree and a stylesheet, the XSLT processor carries out the transformation described by rules in the stylesheet by following a sequence of steps...”

http://oreilly.com/catalog/orxmlapp/chapter/ch07.html

Einfaches Transformationsszenario

XML Quelldokument	XML Struktur: Element-, Attribut-, Textknoten
XSLT Dokument	Anweisungen wie die Knoten im XML Dokument weiterverarbeitet werden sollen
XSLT Prozessor	Liest Knoten für Knoten im Quelldokument und transformiert sie basierend auf den Anweisungen im XSLT Dokument

Einfaches Transformationsszenario

- XSLT Dokument ist XML und muss valide sein!!!
- XSLT Dokumente werden üblicherweise in Datein mit *.xsl gespeichert:
  - beispieldatei.xsl
- Alle XSLT Elemente gehören zum XSLT Namensraum:
  - XSLT Namensraum: http://www.w3.org/1999/XSL/Transform
  - Namensraum und xsl-Kürzel werden am Wurzelelement angegeben
  - Andere Namensräume müssen auch angegeben werden

Einfaches Transformationsszenario

Aufbau eines einfachen XSLT Dokuments (auf nächster Folie):

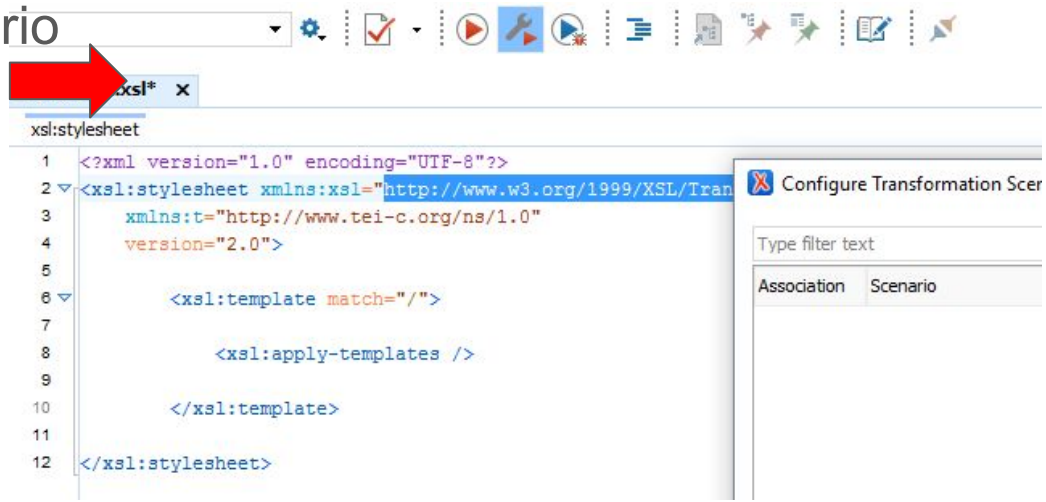
- Zeile 1: Processing Instruction, gibt XML Version und Zeichenkodierung an
- Zeile 2 + 13: Wurzelelement ‘`xsl:stylesheet`’
  - In Attributen des Wurzelelements werden die Namensräume (xslt - Zeile 3 und TEI - Zeile 4) & die XSLT Version (Zeile 5) angegeben
- Zeile 7 + 11: Template (Schablone/Vorlage), Anweisung für die Transformation des Wurzelelements (`@match` + XPath Selector)
- Zeile 9: XSLT Anweisung die Kindelemente weiterzubearbeiten

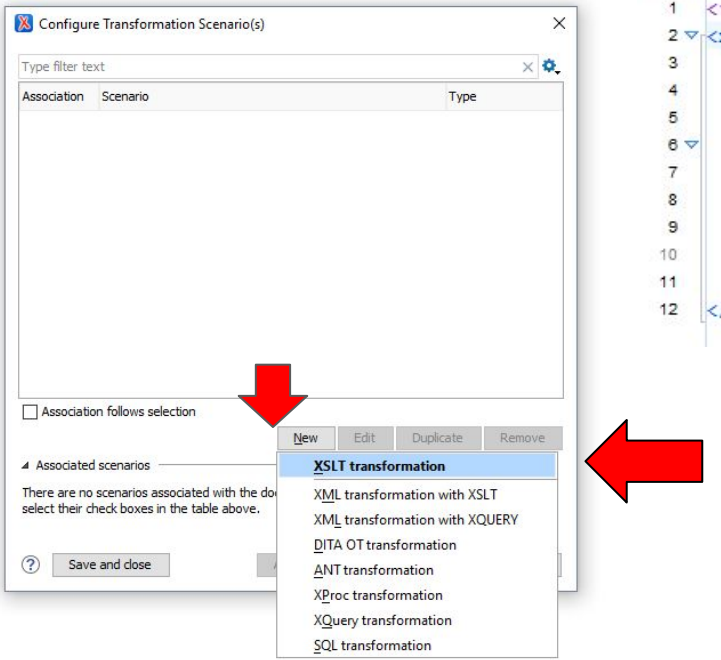
Einfaches Transformationsszenario

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet
3      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
4      xmlns:t="http://www.tei-c.org/ns/1.0"
5      version="2.0">
6
7          <xsl:template match="/">
8
9              <xsl:apply-templates />
10
11         </xsl:template>
12
13 </xsl:stylesheet>
```

XSL Transformation mit oXygen?

Toolbar: Transformationsszenario Konfigurieren



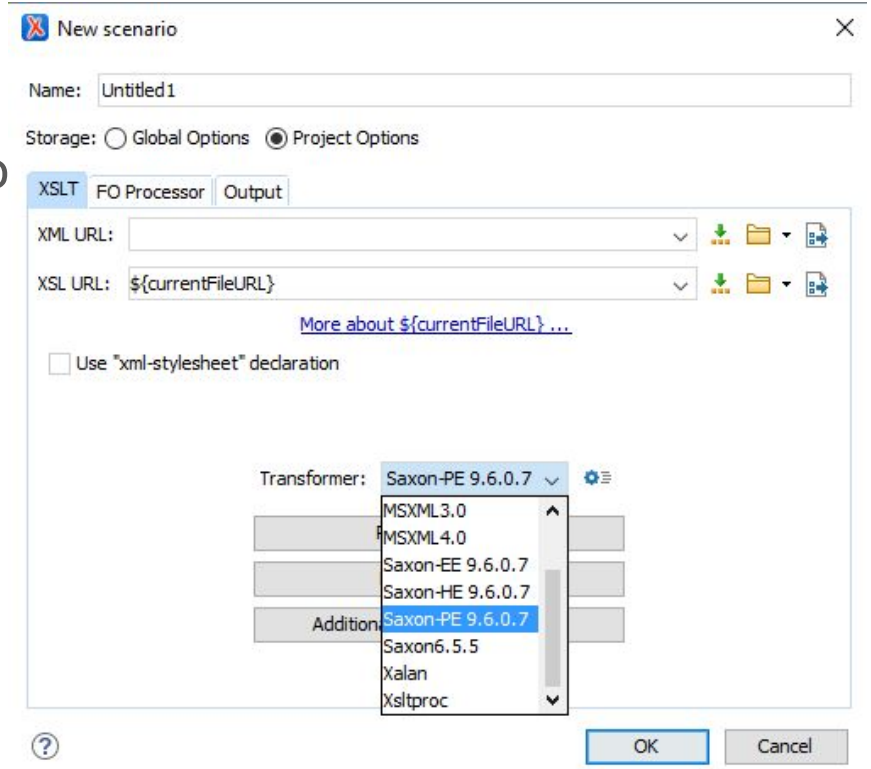


Neues XSLT Transformationsszenario anlegen

XSL Transformation mit oXygen?

XSLT Reiter:

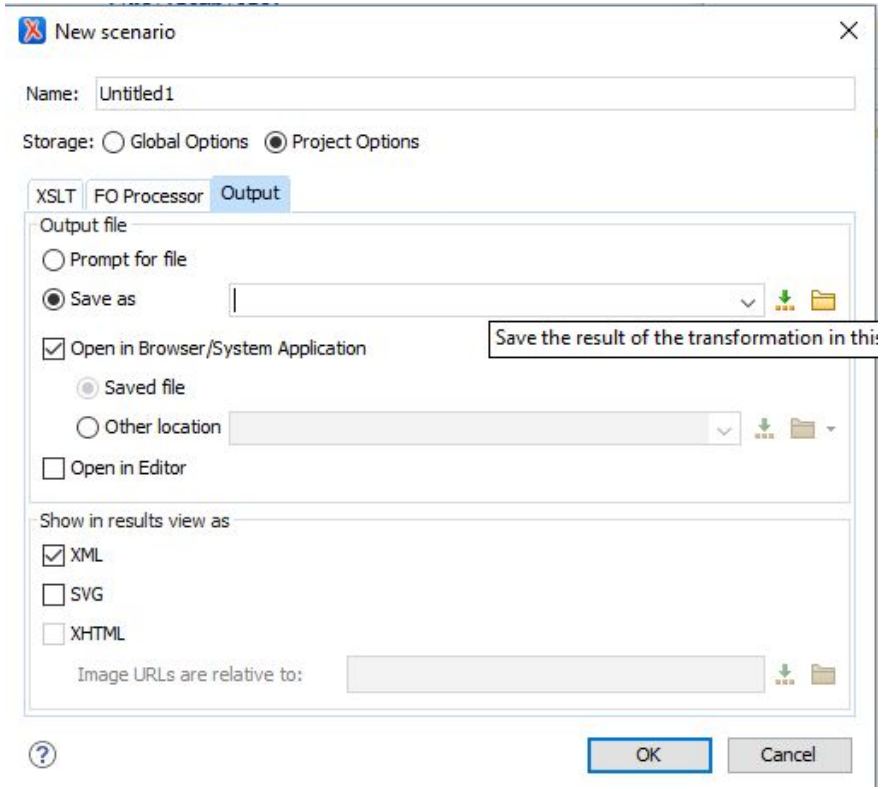
- Name des Transformationszenario
- Die URL zur XML Quelldokument
- Die URL zum XSLT Dokument
- Auswahl eines XSLT Prozessors



XSL Transformation mit oXygen?

Output Reiter:

- Angabe des Speicherorts des Ergebnisdokumentes
- Soll das Ergebnisdokument vom Webbrowser oder oXygen Editor geöffnet werden?
- Wie soll das Ergebnisdokument angezeigt werden?



Übung 1: oXygen Übung

Im Donnerstag Ordner auf http://ginko.uni-graz.at/de2017 folgende Übungsdateien herunterladen und speichern:

- Übung_booklist.xml
- Die Anleitung befindet sich in: Übung1_Anleitung.pdf

XSLT Elemente: Grundlagen

Warum wird bei diesem einfachen Beispiel Text ausgegeben?

=> XSLT Def

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet
3      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
4      xmlns:t="http://www.tei-c.org/ns/1.0"
5      version="2.0">
6
7          <xsl:template match="/">
8
9              <xsl:apply-templates />
10
11         </xsl:template>
12
13 </xsl:stylesheet>
```

XSLT Elemente: Grundlagen

XSLT Default-Verhalten:

- Element- und Dokumentknoten werden aufgerufen:

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

- Kommentare und Processing-instructions werden nicht ausgegeben: `<xsl:template match="processing-instruction()|comment()"/>`
- Text- und Attributknoten werden ausgegeben:

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

XSLT Elemente: Grundlagen

Template Element: `<xsl:template>`

- Über das Attribute **`@match`** kann ein Knoten über einen XPath-Ausdruck ausgewählt werden: **`<xsl:template match="//tei:persName">`**
  - wird mit **`<xsl:apply-templates />`** aufgerufen
- Oder über **`@name`** ein Name vergeben werden: **`<xsl:template name="footer">`**
  - wird mit **`<xsl:call-template name="footer">`** aufgerufen
  - Mit **`<xsl:with-param name="websiteURL">`** können beim Aufruf auch Parameter übergeben werden

XSLT Elemente: Grundlagen

`<xsl:value-of>` wird zur Ausgabe von Textknoten verwendet
Mit **`@select`** und einem XPath Ausdruck wird ein Textknoten ausgewählt
Beispiele:

- `<xsl:value-of select="."/>`
- `<xsl:value-of select="$variable"/>`
- `<xsl:value-of select="tei:persName"/>`
- `<xsl:value-of select="text()"/>`
- `<xsl:value-of select="@type"/>`

XSLT Elemente: Grundlagen

Schleifen: **`<xsl:for-each>`** mit **`@select:`** iterierte über eine mit **`@select`** ausgewählten Gruppe von Knoten

```
<xsl:template match="/bookList">
    <ul>
        <xsl:for-each select="bibl[@type='literature']">
            <li>

                <xsl:apply-templates select="author"/>
                ,
                <xsl:value-of select="title"/>

            </li>
        </xsl:for-each>
    </ul>
</xsl:template>
```

XSLT Elemente: Grundlagen

Bedingungen: **<xsl:if>** und **@test** (enthält die Bedingung)
Wird die Bedingung erfüllt, werden die Anweisungen innerhalb **<xsl:if>** ausgeführt

```
<ul>
    <xsl:for-each select="bibl[@type='literature']">
        <li>
            <xsl:apply-templates select="author"/>
            <xsl:if test="author and title">
                <xsl:text>, </xsl:text>
            </xsl:if>
            <xsl:value-of select="title"/>
        </li>
    </xsl:for-each>
</ul>
```

XSLT Elemente: Grundlagen

Komplexere Bedingungen: **<xsl:choose>\<xsl:when> <xsl:otherwise>**

```
<ul>
    <xsl:for-each select="bibl[@type='literature']">
        <li>
            <xsl:choose>
                <xsl:when test="author">
                    <xsl:apply-templates select="author"/>
                        <xsl:text>, </xsl:text>
                  </xsl:when>
                <xsl:when test="translator">
                    <xsl:apply-templates select="translator"/>
                    <xsl:text>(trans.), </xsl:text>
                </xsl:when>
                <xsl:otherwise><!-- do nothing --></xsl:otherwise>
            </xsl:choose>
            <xsl:value-of select="title"/>
        </li>
    </xsl:for-each>
</ul>
```

XSLT Elemente: Grundlagen

Ausgabe eines Elements: **`<xsl:element>`**
Ausgabe eines Attributs: **`<xsl:attribute>`**
Ausgabe von Text: **`<xsl:text>`**

```
<xsl:element name="li">

    <xsl:attribute name="class" select="@type"></xsl:attribute>
    <xsl:attribute name="style">color:green</xsl:attribute>

    <xsl:text>Some text</xsl:text>

</xsl:element>
```

Übung 2: grundlegende XSL Elemente

Im Donnerstag Ordner auf http://ginko.uni-graz.at/de2017 folgende Übungsdateien herunterladen und speichern:

- Verwende die XML Übungsdatei von vorhin: Übung_booklist.xml
- Die Anleitung befindet sich in: Übung2_Anleitung.pdf

Übung 3: Postkarten TEI Übung

Im Donnerstag Ordner auf http://ginko.uni-graz.at/de2017 folgende Übungsdateien herunterladen und speichern:

- Verwende die XML Übungsdatei: Übung_Postkarte.xml
- Die Anleitung befindet sich in: Übung3_Anleitung.pdf

TEI Frameworks

TEI frameworks haben oft ein XSLT Stylesheet

für die grundlegende Transformation (TEI => HTML)

Grundverständnis von XSLT ist notwendig um mit solchen Frameworks zu arbeiten

Beispiele:

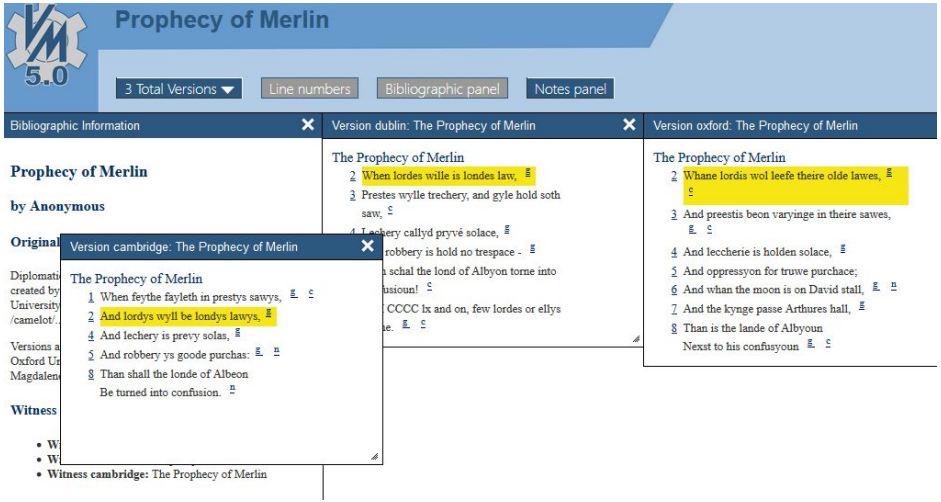
TEI Boilerplate: http://dcl.ils.indiana.edu/teibp/

DEMO: http://dcl.slis.indiana.edu/teibp/content/demo.xml

TEI Frameworks

Versioning Machine: http://v-machine.org/

DEMO: http://v-machine.org/samples/



TEI Frameworks

TEI TAPAS: http://tapasproject.org/

Ziel: Kostengünstige Langzeitarchivierungslösung für TEI Dokumente

DEMO:
http://www.tapasproject.org/tapas-commons/files/letter-written-thomas-monroe-franklin-and-polly-tanner-south-granville-new-york

=> Präsentationsansichten: TEI Boilerplate und TAPAS Generic

Weiterführende Literatur

XML-Kurzreferenzen:
https://www.i-d-e.de/publikationen/weitereschriften/xml-kurzreferenzen/

w3school.com: http://www.w3schools.com/xsl/

IBMs XSLT Tutorial:
http://www.ibm.com/developerworks/xml/tutorials/x-introxslt/

Jeni Tennison, Beginning XSLT 2.0: From Novice to Professional (2. Ed. 2005).

Michael Kay, XSLT 2.0: Programmer's Reference (3. Ed. 2004).

ZENTRUM FÜR INFORMATIONSMODELLIERUNG AUSTRIAN CENTRE FOR DIGITAL HUMANITIES

i|d|e

KARL-FRANZENS-UNIVERSITÄT GRAZ
UNIVERSITY OF GRAZ

Fragen? Einfach melden...

Roman Bleier, ZIM - ACDH
roman.bleier@uni-graz.at

Carina Koch, ZIM - ACDH
carina.koch@uni-graz.at

INZ ÖAW ÖSTERREICHISCHE AKADEMIE DER WISSENSCHAFTEN

FWF Der Wissenschaftsfonds.

UNIVERSITÄT SALZBURG