Extensible Stylesheet Language Transformations

XSLT

Grundlagen der Stylesheets

- XSLT
  - Extensible Stylesheet Language Transformation
- Transformation
  - Überführt XML-Daten in ein neues Format
    - wieder XML
    - HTML
    - WML
    - ...
- XSLT wird benötigt, um XML-Daten zu visualisieren
  - XML-Daten selbst besitzen keine präsentationsbezogenen Informationen

Transformation

- Transformation geschieht dadurch, dass
  - „Schablonen“ (templates) gesucht werden, die auf ein Element passen
  - der Inhalt der Schablonen dann „produziert“ (ausgegeben) wird
- In den Schablonen kann man weitere Schablonen anwenden
- Damit wird der Dokumentbaum sukzessive depth-first-artig durchwandert und jedes Element durch den Inhalt der Schablone ersetzt
- Ähnlichkeit zu kontextfreier Grammatik

Schablonen definieren

- Schablonen werden über **<xsl:template match="ABC">** definiert
  - diese Schablone wird immer dann aktiviert, wenn ein Element mit Namen „<ABC>“ auftaucht
- Anwendung
  - <xsl:apply-templates/> : alle Schablonen anwenden, je nach Element
  - <xsl:apply-templates select="ABC"/>: nur Schablonen anwenden für ABC-Elemente

Schablonen definieren

- Beispiel

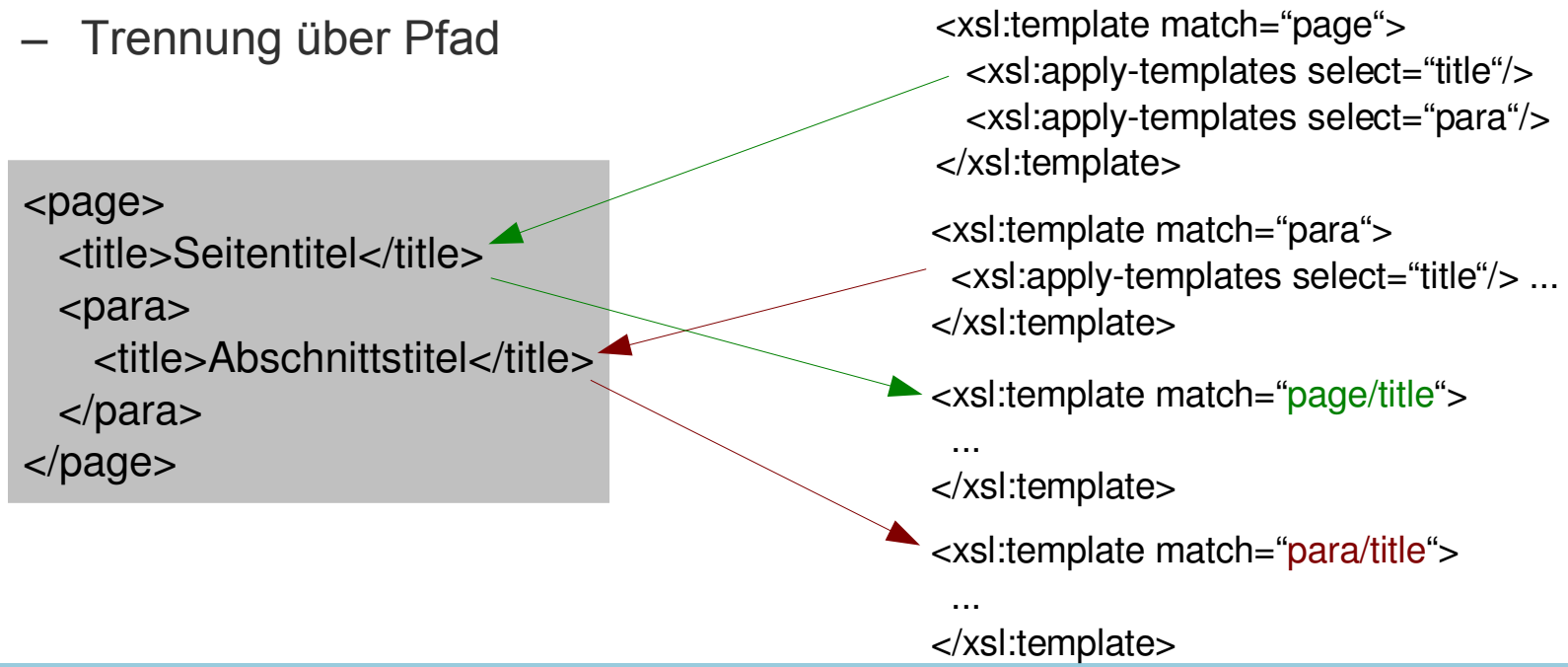
```
<html>
<body>
  <xsl:apply-templates select="para"/>
</body>
</html>
```

```
<xsl:template match="para">
  <p class="meinParagraph"><xsl:apply-templates/></p>
</xsl:template>
```

Alle eingebetteten Elemente auswerten

Schablonen

- Wichtig
  - Das Beispiel würde auch mit <xsl:apply-templates/> funktionieren, wenn <para> die einzigen Kindelemente sind
- Schablone gefunden
  - Aktueller Knoten ist relativer Startpunkt für weitere Schablonen
  - Trennung über Pfad



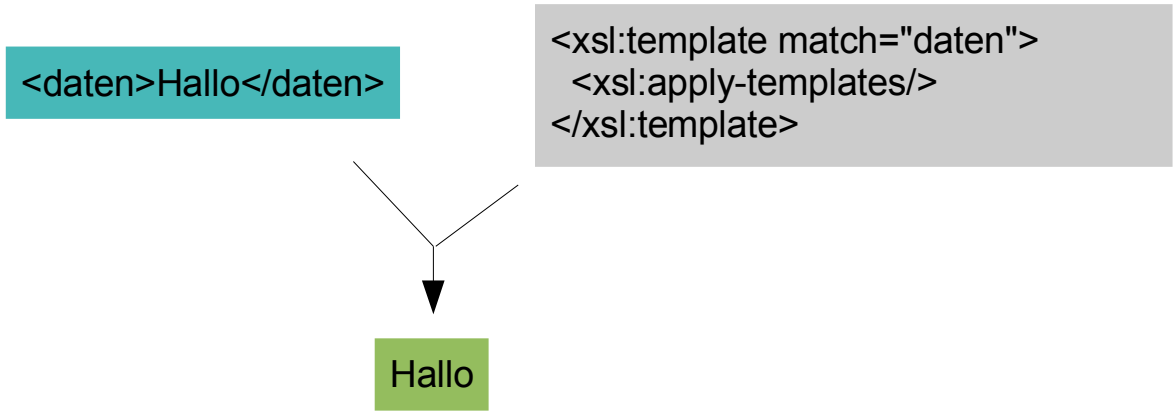
Textknoten verarbeiten

- Für Textknoten im XML gibt es eine implizite Schablone, die deren Inhalt ausgibt (daher werden sie bei <xsl:apply-templates/> ausgegeben)

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

implizit, muss nicht definiert werden!

text() trifft jeden Textknoten
„.“ ist das aktuelle Element (XPath)

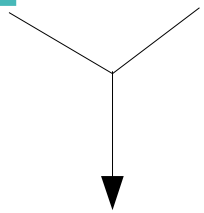


Textknoten

- Erzeugen eines Textknotens
  - <xsl:value-of select="element"> oder
  - <xsl:value-of select="@attribut">
- Beispiel

```
<person titel="Dr.">
   <vorname>Michael</vorname>
   <nachname>Zapf</nachname>
</person>
```

```
<xsl:template match="person">
  <p>
   <xsl:value-of select="@titel"/>
   <xsl:text> </xsl:text>
   <xsl:value-of select="vorname"/>
   <xsl:text> </xsl:text>
   <xsl:value-of select="nachname"/>
  </p>
</xsl:template>
```



```
<p>Dr. Michael Zapf</p>
```

Die fünf Textknoten verschmelzen zu einem einzigen Textknoten (innerhalb von <p>)

Alternativen

- Ähnlich wie in Java
  - <xsl:choose>...<xsl:when> entspricht switch in Java
  - <xsl:if> entspricht if in Java

<section title="Überschrift" level="2">

Auf Anführungszeichen achten!

```
<xsl:template match="section">
<xsl:choose>
    <xsl:when test="@level='1'">
      <h1><xsl:value-of select="@title"/></h1>
    </xsl:when>
    <xsl:when test="@level='2'">
      <h2><xsl:value-of select="@title"/></h2>
    </xsl:when>
    <xsl:otherwise>
      <h3><xsl:value-of select="@title"/></h3>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

Transformation

- Wurzelelement
  - heißt in unserem Beispiel <page title="...">

```
<xsl:template match="page">
<html>
  <head>
    <title>Mein Dokument: <xsl:value-of select="@title"/></title>
    <meta name="GENERATOR" content="Handmade" />
    <meta name="AUTHOR" content="ich selbst" />
    <link rel="stylesheet" type="text/css" href="mein_css_falls_vorhanden.css" />
  </head>
  <body>
    <xsl:apply-templates/>
  </body>
</html>
</xsl:template>
```

Konversion in HTML

- Kopfzeile ganz zu Beginn der XSL-Datei
  - produziert entsprechenden Kopf für HTML-Datei

```
<xsl:stylesheet version="1.1"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="html"
            version="4.01"
            encoding="iso-8859-1"
            doctype-public="-//W3C//DTD HTML 4.01 Strict//EN"
            doctype-system="http://www.w3.org/TR/html401/strict.dtd"
            media-type="text/html"/>

... hier kommen die Schablonen hin ...

</xsl:stylesheet>
```

→ <!DOCTYPE...>

Anwendungen

- XSL-Transformatoren, z.B. Saxon (saxon.sourceforge.net)
  - Java-Anwendung
  - **java -classpath saxon8.jar net.sf.saxon.Transform meinedatei.xml meinstil.xsl**
- Spezifikation
  - http://www.w3.org/TR/xslt