Heutiges Tutorium

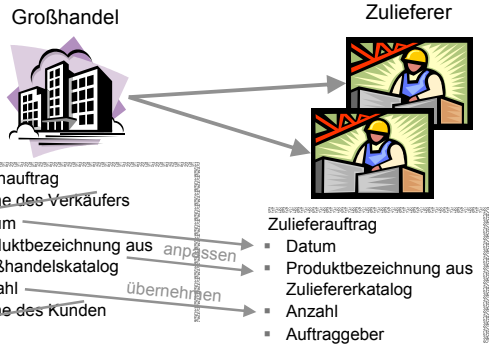
- Übungsblatt 3
- Beispiel eines XSLT-Stylesheet
  - aus Hunter et al., Beginning XML, 2001
  - XML-Inhalt ➔XML-Inhalt
- XSLT und Namensräume

Übungsblatt 3: mögliches Vorgehen

1. XSLT-Stylesheet für Instanz der Übung 2
2. Mit XMLSpy (Enterprise Edition) aus XML-Schema eine Instanz generieren und damit testen:

3. Aus XML-Schema eine *minimale* Instanz generieren und XSLT-Stylesheet damit testen:

Beispiel



XML-Inhalt ➔ XML-Inhalt

Kundenauftrag

```
<?xml version="1.0"?>
<order>
 <salesperson>John Doe</salesperson>
 <item>Production-Class Widget</item>
 <quantity>16</quantity>
 <date>
  <month>1</month>
  <day>13</day>
  <year>2000</year>
 </date>
 <customer>Sally Finkelstein</custome
</order>
```

XSLT

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
 <date>2000/1/13</date>
 <customer>Company A</customer>
 <item>
  <part-number>E16-25A</part-number>
  <description>Production-Class Widget</description>
  <quantity>16</quantity>
 </item>
</order>
```


Das Stylesheet für das Beispiel

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" encoding="UTF-8"/>
  <xsl:template match="/">
    …
  </xsl:template>
  <xsl:template match="item">
    …
  </xsl:template>
</xsl:stylesheet>
```

- method="xml": XML-Deklaration wird erzeugt, Wohlgeformtheit wird aber nicht garantiert.
- indent="yes": Prozessor darf Kind-Elemente einrücken.

Datum transformieren

Kundenauftrag

```
<?xml version="1.0"?>
<order>
 <salesperson>John Doe</salesperson>
 <item>Production-Class Widget</item>
 <quantity>16</quantity>
 <date>
  <month>1</month>
  <day>13</day>
  <year>2000</year>
 </date>
 <customer>Sally Finkelstein</custome
</order>
```

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
 <date>2000/1/13</date>
 <customer>Company A</customer>
 <item>
  <part-number>E16-25A</part-number>
  <description>Production-Class Widget</description>
  <quantity>16</quantity>
 </item>
</order>
```

Datum transformieren

```
<xsl:template match="/">
  <order>
    <date>
      <xsl:value-of select="order/date/year"/> /
      <xsl:value-of select="order/date/month"/> /
      <xsl:value-of select="order/date/day"/>
    </date>
    …
  </order>
</xsl:template>
```

Ergebnis

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  …
</order>
```


Kunden-Element erzeugen

Kundenauftrag

```
<?xml version="1.0"?>
<order>
 <salesperson>John Doe</salesperson>
 <item>Production-Class Widget</item>
 <quantity>16</quantity>
 <date>
   <month>1</month>
   <day>13</day>
   <year>2000</year>
 </date>
 <customer>Sally Finkelstein</custome
</order>
```

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
 <date>2000/1/13</date>
 <customer>Company A</customer>
 <item>
   <part-number>E16-25A</part-number>
   <description>Production-Class Widget</description>
   <quantity>16</quantity>
 </item>
</order>
```


Kunden-Element erzeugen

```
<xsl:template match="/">
  <order>
    <date>
      <xsl:value-of select="order/date/year"/>/
      <xsl:value-of select="order/date/month"/>/
      <xsl:value-of select="order/date/day"/>
    </date>
    <customer>Company A</customer>
    …
  </order>
</xsl:template>
```

Ergebnis

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  <customer>Company A</customer>
  …
</order>
```


Anzahl übernehmen

Kundenauftrag

```
<?xml version="1.0"?>
<order>
 <salesperson>John Doe</salesperson>
 <item>Production-Class Widget</item>
 <quantity>16</quantity>
 <date>
   <month>1</month>
   <day>13</day>
   <year>2000</year>
 </date>
 <customer>Sally Finkelstein</custome
</order>
```

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
 <date>2000/1/13</date>
 <customer>Company A</customer>
 <item>
   <part-number>E16-25A</part-number>
   <description>Production-Class Widget</description>
   <quantity>16</quantity>
 </item>
</order>
```


Anzahl übernehmen

```
<xsl:template match="/">
  <order>
    <date>…</date>
    <customer>Company A</customer>
    <item>
      …
      <quantity>
        <xsl:value-of select="order/quantity"/>
      </quantity>
    </item>
  </order>
</xsl:template>
```

Ergebnis

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  <customer>Company A</customer>
  <item> …
    <quantity>16</quantity>
  </item>
</order>
```


Artikelnummer erstellen

Kundenauftrag

```
<?xml version="1.0"?>
<order>
 <salesperson>John Doe</salesperson>
 <item>Production-Class Widget</item>
 <quantity>16</quantity>
 <date>
   <month>1</month>
   <day>13</day>
   <year>2000</year>
 </date>
 <customer>Sally Finkelstein</custome
</order>
```

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
 <date>2000/1/13</date>
 <customer>Company A</customer>
 <item>
   <part-number>E16-25A</part-number>
   <description>Production-Class Widget</description>
   <quantity>16</quantity>
 </item>
</order>
```

Artikelnummer erstellen

```
<xsl:template match="/">
  <order>
    <date>…</date>
    <customer>Company A</customer>
    <item>
      <xsl:apply-templates select="order/item"/>
      <quantity>
        <xsl:value-of select="order/quantity"/>
      </quantity>
    </item>
  </order>
</xsl:template>
```


Artikelnummer erstellen

```
<xsl:template match="item">
  <part-number>
    <xsl:choose>
      <xsl:when test=". = 'Production-Class Widget'">E16-25A</xsl:when>
      <xsl:when test=". = 'Economy-Class Widget'">E16-25B</xsl:when>
      <xsl:otherwise>00</xsl:otherwise>
    </xsl:choose>
  </part-number>
  …
</xsl:template>
```

Ergebnis

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  <customer>Company A</customer>
  <item>
    <part-number>E16-25A</part-number>
    …
    <quantity>16</quantity>
  </item>
</order>
```


Zugriff auf externe Datenbanken

```
<xsl:template match="item">
  <part-number>
    <xsl:choose>
      <xsl:when test=". = 'Production-Class Widget'">E16-25A</xsl:when>
      <xsl:when test=". = 'Economy-Class Widget'">E16-25B</xsl:when>
      <xsl:otherwise>00</xsl:otherwise>
    </xsl:choose>
  </part-number>
  …
</xsl:template>
```

- Kein direkter Zugriff auf eine Datenbank möglich.
- xsl:import erlaubt jedoch importieren externer Templates.
- Lösung: Datenbank ➔ Templates ➔ xsl:import

Artikelbeschreibung übernehmen

Kundenauftrag

```
<?xml version="1.0"?>
<order>
  <salesperson>John Doe</salesperson>
  <item>Production-Class Widget</item>
  <quantity>16</quantity>
  <date>
    <month>1</month>
    <day>13</day>
    <year>2000</year>
  </date>
  <customer>Sally Finkelstein</customer>
</order>
```

Zulieferauftrag

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  <customer>Company A</customer>
  <item>
    <part-number>E16-25A</part-number>
    <description>Production-Class Widget</description>
    <quantity>16</quantity>
  </item>
</order>
```


Artikelbeschreibung übernehmen

```
<xsl:template match="item">
  <part-number>…</part-number>
  <description>
    <xsl:value-of select="."/>
  </description>
</xsl:template>
```

- Beachte: "." bezeichnet *aktuellen* Knoten, hier den Knoten, auf den das Template angewandt wird ("item").

Artikelbeschreibung übernehmen

```
<xsl:template match="item">
  <part-number>…</part-number>
  <description>
    <xsl:value-of select="."/>
  </description>
</xsl:template>
```

Ergebnis

```
<?xml version="1.0" encoding="UTF-8"?>
<order>
  <date>2000/1/13</date>
  <customer>Company A</customer>
  <item>
    <part-number>E16-25A</part-number>
    <description>Production-Class Widget</description>
    <quantity>16</quantity>
  </item>
</order>
```

Fazit aus dem Beispiel

- Beispiel-Stylesheet nicht gut strukturiert:
  - besser für jede Teiltransformation eigenes Template
  - besser viele kleine Templates statt eines großen Templates
- Anbindung von Datenbanken umständlich:

  Datenbank ➔ Templates ➔ xsl:import
- Beispiel-Transformation sollte auf Server durchgeführt werden, da Kundenauftrag extern nicht sichtbar sein sollte.
- Auf dem Server kann aber für die Transformation eine beliebige andere Programmiersprache benutzt werden.

XSLT 1.0 und Namensräume

- XSLT 1.0 / XPath kennt *keine* Namensräume
- In XSLT 1.0 / XPath dürfen aber Namen mit ":" benutzt werden.
- bei namensraumeingeschränkten Elementen:
  - Namensraum-Präfix definieren
  - In XPath-Pfaden Präfix voranstellen:

    <xsl:value-of select="ns:order/ns:date/ns:year"/>
  - Standard-Namensraum statt Präfix nicht möglich!