## Übung Invoice

### Invoice XML

Mit dieser Übung soll das gegebene XML Dokument "invoice" via XSLT nach HTML transformiert werden. Erstellen Sie in der Eclipse IDE das Projekt "invoice" und die Datei invoice.xml. Kopiere Sie das XML Dokument aus dem folgenden Listing in die Datei invoice.xml:<?xml version="1.0" encoding="UTF-8"?>
<invoice nr="123">
 <head>
 <customer nr="789">
 <name>
 <firstname>Hans</firstname>
 <lastname>Muster</lastname>
 </name>
 <address>
 <street>Freiburgstrasse 20</street>
 <postalcode>3000</postalcode>
 <city>Bern</city>
 </address>
 </customer>
 <distributor>
 <company>Simtech AG</company>
 <address>
 <street>Marktgasse 6</street>
 <postalcode>3011</postalcode>
 <city>Bern</city>
 </address>
 </distributor>
 </head> 
 <positions>
 <position pos="3" price="600" currency="CHF">
 <description>PC Intel III 600 Mhz</description>
 </position>
 <position pos="2" price="700" currency="CHF">
 <description>21-Zoll Monitor</description>
 </position>
 <position pos="4" price="15" currency="CHF">
 <description>Maus</description>
 </position>
 <position pos="1" price="45" currency="EUR">
 <description>Tastatur</description>
 </position>
 </positions>
</invoice>

### Step 1

Programmieren Sie ein XSLT Stylesheet, so dass das folgende HTML Dokument transformiert wird: Das folgende XSLT Listing zeigt eine mögliche Vorlage:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 version="2.0">
 <xsl:template match="/">
 <html>
 <head>
 <title>Invoice</title>
 <style type="text/css">
 * {
 font-family: arial;
 }
 
 table {
 border:1px solid black;
 }
 
 th, td {
 padding: 2px;
 }

 span.label {
 display: inline-block;
 width: 5em;
 }
 
 </style>
 </head>
 <body>
 <h1>Rechnung Nr: ...</h1>
 <xsl:apply-templates select="invoice/head/customer"/>
 <xsl:apply-templates select="invoice/head/distributor"/>
 <p>
 <xsl:apply-templates select="invoice/positions"/>
 </p>
 </body>
 </html>
 </xsl:template>
 <!--
 template: customer
 -->
 <xsl:template match="customer">
 <h2>Kunde</h2>
 <span class="label">Kundennr:</span>...<br/>
 <span

class="label">Name:</span>...<br/>
 <xsl:apply-templates select="address"/>
 </xsl:template>
 <!--
 template: distributor
 -->
 <xsl:template match="distributor">
 <h2>Lieferant</h2>
 <span class="label">Firma:</span>...<br/>
 <xsl:apply-templates select="address"/>
 </xsl:template>
 <!-- 
 template: address
 -->
 <xsl:template match="address">
 <span class="label">Adresse:</span>..., 
 ...
 </xsl:template>
 <!--
 template: positions
 -->
 <xsl:template match="positions">
 <table>
 <tr>
 <th>Pos</th>
 <th>Artikel</th>
 <th>Preis</th>
 <th>Währung</th>
 </tr>
 ...
 </table>
 </xsl:template>
 <!-- 
 template: position
 -->
 <xsl:template match="position">
 <tr>
 <td>...</td>
 <td>...</td>
 <td>...</td> 
 <td>...</td> 
 </tr>
 </xsl:template>
</xsl:stylesheet>Erstellen Sie die Datei "invoice1.xsl" und kopieren Sie die obige Vorlage in die Datei. Ergänzen Sie das Script, so dass die Daten in das HTML Dokument korrekt gerendert werden.Das Saxon Script:java -jar ..\saxon\saxon-he-11.4.jar -s:invoice.xml -xsl:invoice1.xsl -o:invoice1.html

**Step 2**

Mit diesem Schritt wollen wir pro Währung eine separate Tabelle führen. Weiter sollen die geraden und ungeraden Zeilen unterschiedlich dargestellt werden.Definieren Sie ein Template, welches die Tabelle bezogen auf eine Währung anzeigt. Das gleiche Template soll für alle Währungen funktionieren. Steuern Sie das Template über Template Parameter (xsl:with-param). Der Parameter soll die gesuchte Währung (z.B. CHF) mitgeben, welche alsdann im XPath als Kriterium dienen kann.Integrieren Sie das Template in Ihre Transformation, so dass alle Währungen berücksichtigt werden. Es soll für jede Währung eine eigene Tabelle angezeigt werden.Definieren Sie für die geraden (even) und ungeraden (odd) Zeilen ein unterschiedliches Layout. Sie können zwischen geraden und ungeraden Zeilen über die XPath Methode "position()" unterscheiden: <xsl:if test="position() mod 2 = 0">
 // even 
</xsl:if>
<xsl:if test="position() mod 2 != 0">
 // odd
</xsl:if>Das folgende Bild zeigt ein mögliches Layout:Das Saxon Script:java -jar ..\saxon\saxon-he-11.4.jar -s:invoice.xml -xsl:invoice2.xsl -o:invoice2.html

**Step 3**

Jetzt möchten wir alle Preise in den Währungen CHF und Euro anzeigen lassen. Schreiben Sie ein Template, welches in einer weiteren Tabelle alle Artikel anzeigt und die Preise in CHF und Euro (je eine Spalte) anzeigt. Suchen Sie den aktuellen Umrechnungskurs im Internet. Fügen Sie in die Tabelle eine letze Zeile mit den Totalpreisen zu den jeweiligen Währungen hinzu. Schreiben Sie hierzu wiederum ein separates Template und setzen Sie für die Summenberechnung die XPath Methode "sum()" ein. Hierzu müssen Sie die Summe der einzelnen Währungen berechnen und alsdann das ganze zusammenzählen (+). Sortieren Sie die Preise aufsteigend. Das folgende Bild zeigt ein mögliches Layout:Das Saxon Script:java -jar ..\saxon\saxon-he-11.4.jar -s:invoice.xml -xsl:invoice3.xsl -o:invoice3.html

**Lösungen**

- Lösung zu Step 1
- invoice1.xsl

- Lösung zu Step 2
- invoice2.xsl
- Lösung zu Step 3
- invoice3.xsl

## Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

## Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

## Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

## https://www.simtech-ag.ch/online-schulung-xslt-kurse-kurs-xml-transformation-übungen