

XML-Praktikum SS 2015

Belinda Zahra

Thea Heim

XML-Praktikum SS 2015

Belinda Zahra
Thea Heim

Table of Contents

1. Einleitung	1
Beschreibung und Motivation von CalendarX	1
Ziel und Organisation des XML-Praktikums	1
2. Das Kalender-System CalendarX	2
XML Schema: Die Architektur von CalendarX	2
Umsetzung der Tages-, Wochen und Monatssicht	2
SVG zur Erstellung der Templates	2
XSLT zur Generierung der Event-Elemente	3
Wiederverwendung der Templates	4
Uhrzeit	4
Einbindung der FuncX Bibliothek	4
Benutzeroberfläche	4
Protokoll zwischen Client und Server	5
Server-Komponente	5
3. Reflexion	6
Arbeit im Team	6
Thematik CalendarX	6
Organisation und Betreuung im Praktikum	6

List of Figures

2.1. Template Tagessicht	4
--------------------------------	---

Chapter 1. Einleitung

Beschreibung und Motivation von CalendarX

Text

Ziel und Organisation des XML-Praktikums

Text

Chapter 2. Das Kalender-System

CalendarX

XML Schema: Die Architektur von CalendarX

Das konzeptuelle Schema für Kalender-Daten

Umsetzung der Tages-, Wochen und Monatssicht

SVG zur Erstellung der Templates

Die Templates der Tages-, Wochen und Monatssicht wurden mit Hilfe von scalable Vector Graphics (SVG) erstellt und mit Hilfe der Tages-, Wochen- und Monatsstylesheets aufgerufen. Im ersten Schritt erhält das Template einen Namen, um es später eindeutig zuordnen zu können. Danach wird die SVG-Fläche mit den einleitenden Höhen- und Breitenangaben definiert. Innerhalb der SVG-Tags wird die entsprechende Graphik gezeichnet.

Nachfolgend wird beispielhaft das Template "tagessicht" beschrieben. Dieses enthält 25 horizontale Linien, innerhalb zweier Linien steht jeweils die volle Stunde von 0:00 Uhr bis 23:00 Uhr eingrenzen. Zudem enthält das Template eine vertikale Linie, um die Uhrzeiten von den Kalendereinträgen abzugrenzen.

```
<!-- Schreibt jede volle Stunde auf eine horizontale Linie -->
<xsl:template name="tagessicht">
  <svg width="1300" height="1000" xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink">
    <!-- Im defs-Container werden die Objekte definiert -->
    <defs>
      <!-- Linie horizontal-->
      <line x1="50" y1="50" x2="1150" y2="50"
        stroke-width="2" stroke="grey" id="li"/>
      <!-- Linie vertikal-->
      <line x1="100" y1="50" x2="100" y2="800"
        stroke-width="2" stroke="grey" id="li2"/>
    </defs>
    <!-- Zeichne die oben definierte Linie 25 mal mit Abstand 30 -->
    <use xlink:href="#li" y="30"/>
    <use xlink:href="#li" y="60"/>
    <use xlink:href="#li" y="90"/>
    <use xlink:href="#li" y="120"/>
    <use xlink:href="#li" y="150"/>
    ...
    <!-- Schreibe bestimmte Uhrzeit auf jede Linie -->
    <text x="55" y="100">00:00</text>
    <text x="55" y="130">01:00</text>
    <text x="55" y="160">02:00</text>
    <text x="55" y="190">03:00</text>
    ...
    <!-- Zeichne vertikale Linie neben die Uhrzeiten -->
    <use xlink:href="#li2"/>
  </svg>
```

XSLT zur Generierung der Event-Elemente

Um die Event-Elemente eines Tages, einer Woche oder eines Monats in das entsprechende Template einzutragen wird die XSL Transformation (XSLT), eine Programmiersprache zur Transformation von XML Dokumenten, angewendet. Im ersten Schritt wird der Datei, welche die Beispielergebnisse enthält (sampleEvents.xml) ein Stylesheet zugeordnet, das diese Events absteigend nach Datum und Startzeitpunkt sortiert (sampleEvents.xsl). Das Ergebnis ist eine XML-Datei mit den sortierten Event-Elementen (events_sortiert.xml). Aus dieser Zwischendatei heraus werden dann die entsprechenden Stylesheets, entweder für die Tages-, die Wochen-, oder die Monatssicht aufgerufen (tagessicht.xsl, wochensicht.xsl, monatssicht.xsl).

Nachfolgender Programmcode zeigt beispielhaft, wie die Events aus der sortierten Zwischendatei gefiltert und jeweils als Rechteck, dessen Höhe von der Dauer des Events abhängt, in das oben beschriebene SVG-Template der Tagessicht hineingezeichnet werden.

```
<xsl:template match="/">
  <svg height="200%" width="100%">
    <!-- Template "tagessicht" wird aufgerufen -->
    <xsl:call-template name="tagessicht"/>

    <!-- Globale Variablen -->
    <xsl:variable name="aktuellesDatum" as="xs:date"
      select="document('aktuellesDatum.xml')/datum"/>

    <!-- Schreibe das Datum des Tages in die obere Mitte
    der Seite (festgelegt in aktuellesDatum.xml) -->
    <text x="600" y="25" text-anchor="middle"
    font-size="20" fill="red">
      <xsl:value-of select="$aktuellesDatum"/>
    </text>

    <!-- Falls vorhanden, trage die Termine für den
    Tag in das Template ein -->
    <xsl:for-each
      select="document('events_sortiert.xml')/events/
      event[datum = $aktuellesDatum]">
      <xsl:variable name="startRechteck" select="80 +
      (startZeitInMin div 2)"/>
      <xsl:variable name="endeRechteck" select="80 +
      (endZeitInMin div 2)"/>

      <!-- Zeichne ein Rechteck für die Zeitspanne,
      in der ein Termin stattfindet -->
      <rect x="105" y="{ $startRechteck }" width="1040"
      height="{ ($endeRechteck) - $startRechteck }"
      fill="gainsboro"/>

      <!-- Schreibe Startzeit, Endzeit und die
      Beschreibung in das Rechteck -->
      <text x="115" y="{ $startRechteck + 15 }">
        <xsl:value-of select="startZeit"/> -
        <xsl:value-of select="endZeit"/> :
        <xsl:value-of select="beschreibung"/>
      </text>
    </xsl:for-each>
  </svg>
</xsl:template>
```

Das Ergebnis enthält nun das Datum des aktuellen Tages, das Template der Tagessicht und die Rechtecke für die Events an dem betrachteten Tag. Folgendes Bild zeigt die Tagessicht nach der XSL Transformation:

Figure 2.1. Template Tagessicht

2015-05-07

00:00	
01:00	
02:00	
03:00	
04:00	
05:00	
06:00	
07:00	
08:00	
09:00	
10:00	10:15:00 - 12:00:00 : XML-Praktikum
11:00	
12:00	12:30:00 - 16:30:00 : Team Meeting
13:00	
14:00	
15:00	
16:00	

Wiederverwendung der Templates

Bei der Erstellung der Templates wurde darauf geachtet, dass sie nicht nur für eine Sicht verwendet werden können, sondern auch für die Erstellung der anderen Sichten brauchbar sind.

Mit Hilfe des Befehls `<xsl:include href="tagessicht.xsl"/>` wird das Stylesheet der Tagessicht, dessen Template in der Wochensicht benötigt wird, in diese mit eingebunden. Genauso wird in der Monatssicht das Stylesheet der Tagessicht eingebunden. Mit Hilfe des Befehls `<xsl:call-template name="tagessicht"/>` können einzelne Templates aufgerufen werden.

Uhrzeit

Die Uhrzeit wird durch `xs:time` dargestellt. Die voreingestellte Darstellung eines `xs:time` ist Stunden:Minuten:Sekunden (hh:mm:ss). Dank der Formatierungsfunktion `format-time()` in XSLT 2.0 kann man `xs:time` Werte verwenden und dann das Format dieser ändern. Somit können wir die Sekunden in den Sichten weglassen und erhalten stattdessen die Darstellung Stunden:Minuten (hh:mm).

Einbindung der FuncX Bibliothek

Für die Manipulation von Datums- und Zeitangaben wird die XSLT-Bibliothek FuncX (<http://www.xsltfunctions.com/>) verwendet. Die Bibliothek wird v.a. dazu benötigt, um die Wochen und Monate mit den richtigen Daten zu füllen. Mit Hilfe ihrer Funktionen können so beispielsweise der erste Tag im Monat, die Anzahl der Tage eines Monats, die Nummer eines Tages im Jahr, u.v.m. bestimmt werden.

Benutzeroberfläche

Beschreibung der unterstützten Seiten, den darauf angebotenen Interaktionsmöglichkeiten und dem Layout, umgesetzt in XHTML, CSS und XForms

Protokoll zwischen Client und Server

HTTP-Nachrichten mit URLs und XML-Daten

Server-Komponente

Services, umgesetzt mit XQuery und XSLT

Chapter 3. Reflexion

Arbeit im Team

Text

Thematik CalendarX

Text

Organisation und Betreuung im Praktikum

Text