Übungen zu Web- und XML-Technologien

BA-INF 133, Sommersemester 2018

Dr. Stefan Lüttringhaus-Kappel
Institut für Informatik III, Universität Bonn
stefan@iai.uni-bonn.de

13. Juli 2018

Inhaltsverzeichnis

1 20. April 2018

1.1 Organisatorisches

Termine, Inhalte und Methoden

Übungstermine

- Freitags *14:15 Uhr* bis ca. 15:45 Uhr,
- Globalübung

Übungsinhalte

- Vertiefung, Verständnisaufgaben
- *Selbststudium*, eigene Recherchen
- Kurzreferate
- Open-Source-Werkzeuge
- Programmieraufgaben (JavaScript, XSLT, Java, XQuery)
- Programmierprojekt: Web-Technologien (HTML5, CSS, JavaScript, Node.js)

Wichtig

- Praktische Java-Kenntnisse sind notwendig
- Die erfolgreiche Teilnahme an den Übungen ist Voraussetzung zur Zulassung zur Prüfung.

Zulassung zur Prüfung

Voraussetzungen für die Zulassung

1. Studium nach Bachelor-PO (oder im Nebenfach), *und*
2. erfolgreiche Übungsteilnahme
   - 50% der zu erreichenden Punkte in den Übungsaufgaben
   - Programmierprojekt erfolgreich bearbeitet, Präsentation der Ergebnisse
   - Abgabe und Bewertung im eCampus

1.2 eCampus

eCampus

Information und Kommunikation: eCampus

- Informationen, Termine, Literatur
- Skript, Übungsaufgaben
- Forum nutzen! *Forum als E-Mails abonnieren!* Diskussionsforum → Aktionen → Benachrichtigung für dieses Forum starten

eCampus Quickstart

- Account erstellen: https://ecampus.uni-bonn.de/
- Anmelden zu BA-INF 133 – Web- und XML-Technologien

Anmeldung im eCampus

- **Anmelden im eCampus bis zum 4. Mai 2018!**

Üben: Lösung hochladen

- Übung 0 sofort bearbeiten (1 Extrapunkt)

1.3 Aufgaben

Aufgabe 1

Aufgabe 1 (Forts.)

```
$ java -cp xercesImpl.jar:xercesSamples.jar sax.Counter fehler.xml
[Fatal Error] fehler.xml:2:6: The processing instruction target matching
"[xX][mM][lL]" is not allowed.
```

```
$ head -n3 fehler.xml
<?xml version="1.1" encoding="UTF-8"?>
<!-- Mein zweiter Versuch, XML zu schreiben :-) -->
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

```
$ java -cp xercesImpl.jar:xercesSamples.jar sax.Counter fehler.xml
[Fatal Error] fehler.xml:10:15: The entity "uuml" was referenced,
but not declared.
```

1.4 Beispiele

XML-Datenstrukturen

Beispiel 1.1 (Graph).

```
<graph>
  <vertex id="1"/>  <vertex id="2"/>  <vertex id="3"/>

  <edge type="undirected">
    <vertex-ref>1</vertex-ref>
    <vertex-ref>2</vertex-ref>
  </edge>

  <edge type="directed">
    <vertex-ref>2</vertex-ref>
    <vertex-ref>3</vertex-ref>
    <weight>3</weight>
  </edge>

  <edge type="bi-directional">
    <vertex-ref>1</vertex-ref>
    <vertex-ref>3</vertex-ref>
  </edge>
</graph>
```

XML-Datenstrukturen

Beispiel 1.2 (Konfigurationsdatei).

```
<channel name="xfwm4" version="1.0">
  <property name="general" type="empty">
    <property name="theme" type="string" value="Moheli"/>
    <property name="title_font" type="empty"/>
    <property name="double_click_time" type="empty"/>
```

```
    <property name="workspace_count" type="int" value="3"/>
    <property name="focus_delay" type="int" value="101"/>
    <property name="double_click_distance" type="int" value="5"/>
    <property name="scroll_workspaces" type="bool" value="true"/>
    <property name="shadow_delta_height" type="int" value="0"/>
    <property name="shadow_delta_width" type="int" value="0"/>
    <property name="shadow_delta_x" type="int" value="0"/>
    <property name="shadow_delta_y" type="int" value="-3"/>
    <property name="shadow_opacity" type="int" value="50"/>
    <property name="wrap_resistance" type="int" value="10"/>
    <property name="wrap_workspaces" type="bool" value="false"/>
    <!-- ... -->
    <property name="workspace_names" type="array">
      <value type="string" value="Workspace 1"/>
      <value type="string" value="Workspace 2"/>
      <value type="string" value="Workspace 3"/>
    </property>
    <property name="margin_left" type="int" value="0"/>
  </property>
</channel>
```

XML-Datenstrukturen

Beispiel 1.3 (Scalable Vector Graphics (SVG)).

```
<svg width="12cm" height="4cm" viewBox="0 0 1200 400"
     xmlns="http://www.w3.org/2000/svg" version="1.2"
     baseProfile="tiny">
  <desc>Example rect02 - rounded rectangles</desc>
  <!-- Show outline of canvas using 'rect' element -->
  <rect x="1" y="1" width="1198" height="398"
        fill="none" stroke="blue" stroke-width="2"/>
  <rect x="100" y="100" width="400" height="200" rx="50"
        fill="green" />
  <g transform="translate(700 210) rotate(-30)">
    <rect x="0" y="0" width="400" height="200" rx="50"
          fill="none" stroke="purple" stroke-width="30" />
  </g>
</svg>
```

XML-Datendefinition

Beispiel 1.4 (XML Schema).

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="internationalPrice">
    <xsd:complexType>
      <xsd:attribute name="currency" type="xsd:string"/>
```

```
      <xsd:attribute name="value"    type="xsd:decimal"/>
    </xsd:complexType>
  </xsd:element>

  <!-- ... -->
</xsd:schema>
```

XML-Datenverarbeitung

Beispiel 1.5 (XSLT).

```
<xsl:stylesheet version="2.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns="http://www.w3.org/1999/xhtml">
  <xsl:output method="html"/>
  <xsl:template match="slideshow">
    <html>
      <head>
        <title>
          <xsl:apply-templates select="title"/>
        </title>
      </head>
      <body>
        <h1><xsl:apply-templates select="title"/></h1>
        <xsl:apply-templates select="slides"/>
      </body>
    </html>
  </xsl:template>
  <!-- ... -->
</xsl:stylesheet>
```

XML-Datenstrukturen

```
<modul xmlns="http://cs.uni-bonn.de/modulhandbuch" last-modified="2012-01-30T15:48:22.952+01:00" l
  <aktiv>true</aktiv>  <editierbar>true</editierbar>
  <name>Web- und XML-Technologien</name>
  <name-englisch>Web and XML Technologies</name-englisch>
  <modulnummer>BA-INF 133</modulnummer>
  <dauer>1</dauer>  <turnus>jährlich</turnus>
  <modulverantwortlicher>Dr. Stefan Lüttringhaus-Kappel</modulverantwortlicher>
  <dozenten>
    <dozent>Dr. Stefan Lüttringhaus-Kappel</dozent>
  </dozenten>
  <studiengang>B. Sc. Informatik</studiengang>
  <modus>Wahlpflicht</modus>  <studiensemester>4,6</studiensemester>
  <fachlichekompetenzen>...</fachlichekompetenzen>
  <schluesselkompetenzen>...</schluesselkompetenzen>
  <inhalte>...</inhalte>
  <voraussetzungen>
    <empfohlen>
      <modul>BA-INF 024</modul>
    </empfohlen>
```

```
  </voraussetzungen>
  <modultyp>Kleine Vorlesung</modultyp>
  <leistungen>
    <leistung>
      <lehrform>Vorlesung</lehrform>
      <gruppengroesse>40</gruppengroesse>
    </leistung>
    <leistung>
      <lehrform>Übungen</lehrform>
      <gruppengroesse>20</gruppengroesse>
    </leistung>
  </leistungen>
  <pruefungsleistungen>Schriftliche Prüfung</pruefungsleistungen>
  <studienleistungen>Erfolgreiche Übungsteilnahme</studienleistungen>
  <literatur>...</literatur>
</modul>
```

2 27. April 2018

2.1 eCampus

Forum im eCampus

- Forum als E-Mails abonnieren: Diskussionsforum → Aktionen → Benachrichtigung für dieses Forum starten
- Forum nutzen!

2.2 Prüfungstermine

Prüfungstermine

Siehe eCampus!

2.3 Unicode

Unicode

- Unicode Home Page[1]
- Unicode Character Database[2]
  - `/Public/UCD/latest/ucd/UnicodeData.txt`[3]
- Unicode Character Code Charts[4]

[1] `http://www.unicode.org/`
[2] `http://www.unicode.org/ucd/`
[3] `http://www.unicode.org/Public/UCD/latest/ucd/UnicodeData.txt`
[4] `http://www.unicode.org/charts/index.html`

2.4 Fragen

Fragen? . . .

- Was ist ein Zeichensatz im Unterschied zu einer Kodierung?
- Welche Vorteile hat UTF-8 gegenüber anderen Kodierungen?
- Wie verhalten sich HTML und XML zueinander?
- Erläutere die logische Struktur der XML-Dokumente.
- Warum ist die Trennung von Inhalt und Layout wichtig?

2.5 Aufgaben

Aufgabe: Gültige Instanzen (bzgl. DTD)?

```
<!-- uni.dtd -->
<!ELEMENT uni (prof*, büro*)>
<!ELEMENT prof (#PCDATA)>
<!ATTLIST prof büro IDREF #IMPLIED>
<!ELEMENT büro (#PCDATA)>
<!ATTLIST büro gebäude (altbau|neubau) #REQUIRED>
<!ATTLIST büro id ID #IMPLIED>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <prof>Meier</prof>
  <büro id="a123" gebäude="altbau">123</büro>
  <prof>Müller</prof>
</uni>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <prof>Müller</prof>
  <büro id="n007">007</büro>
</uni>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <büro id="a123" gebäude="altbau">123</büro>
  <prof büro="a123">Meier</prof>
  <prof büro="a123">Schuhmacher</prof>
</uni>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <prof>Meier</prof>
  <prof büro="n077">Schulz</prof>
  <büro id="a123" gebäude="altbau">123</büro>
</uni>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <prof büro="a123">Meier</prof>
  <büro id="a123" gebäude="altbau">123</büro>
</uni>
```

```
<!DOCTYPE uni SYSTEM "uni.dtd">
<uni>
  <prof büro="n007">Fischer</prof>
  <büro id="n007" gebäude="neubau">007 (links)</büro>
  <büro id="n007" gebäude="neubau">007 (rechts)</büro>
</uni>
```

2.6 Entities

Experimente mit Entities

XML parsen und ausgeben (`SaxIdentity.java`, eCampus)

```
import java.util.Stack;
import org.xml.sax.helpers.XMLReaderFactory;
import org.xml.sax.XMLReader;
import com.sun.org.apache.xml.internal.serialize.XMLSerializer;
import com.sun.org.apache.xml.internal.serialize.OutputFormat;

public class SaxIdentity extends XMLSerializer {
  public static void main (String args[]) throws Exception {
    // Input side: parser
    XMLReader reader = XMLReaderFactory.createXMLReader();
    // Output side: serializer
    SaxIdentity handler = new SaxIdentity();
    OutputFormat format = new OutputFormat("xml", "UTF-8", true);
    handler.setOutputByteStream(System.out);
    handler.setOutputFormat(format);
    handler.setNamespaces(true);
    // Connect them
    reader.setContentHandler(handler);
    // Go
    reader.parse(args[0]);
  }
}
```

Erstes Beispiel

f1.xml

```
<!DOCTYPE greeting [
<!ENTITY nick "Harry">
<!ENTITY imprint SYSTEM "impressum.xml">
]>
<greeting>
  <p>Hello, world!</p>
  <p>&nick; greets you. Do you like &nick;?</p>
  &imprint;
</greeting>
```

Testlauf

```
$ java -cp .. SaxIdentity f1.xml
<?xml version="1.0" encoding="UTF-8"?>
<greeting>
  <p>Hello, world!</p>
  <p>Harry greets you. Do you like Harry?</p>
<p>© 2011. Das ist nur ein Test.</p>
<p>Jegliche Garantie wird abgelehnt!</p>
</greeting>
```

Erstes Beispiel (Forts.)

impressum.xml

```
<?xml version="1.1" encoding="UTF-8" ?>
<p>&#xA9; 2011. Das ist nur ein Test.</p>
<p>Jegliche Garantie wird abgelehnt!</p>
```

Zweites Beispiel

f2.xml

```
<!DOCTYPE greeting [
<!ENTITY imprint SYSTEM "impressum.xml">
<!ENTITY wobmotd SYSTEM "http://wobdoc.iai.uni-bonn.de/motd/hi.xml">
]>
<greeting>
  <p>Hello, universe!</p>
  <p>Let's see what's on the web:</p>
  &wobmotd;
  &imprint;
</greeting>
```

Testlauf

```
$ java -cp .. SaxIdentity f2.xml
<greeting>
  <p>Hello, universe!</p>
  <p>Let's see what's on the web:</p>
  <p>Have a lot of fun!</p>
<p>© 2011. Das ist nur ein Test.</p>
<p>Jegliche Garantie wird abgelehnt!</p>
</greeting>
```

2.7 Fragen

3 4. Mai 2018

3.1 Aktuelles

Aktuelle Informationen

Am Freitag, 11. Mai, fällt die Übung aus!

3.2 Aufgaben

XML Information Set — Begriffe

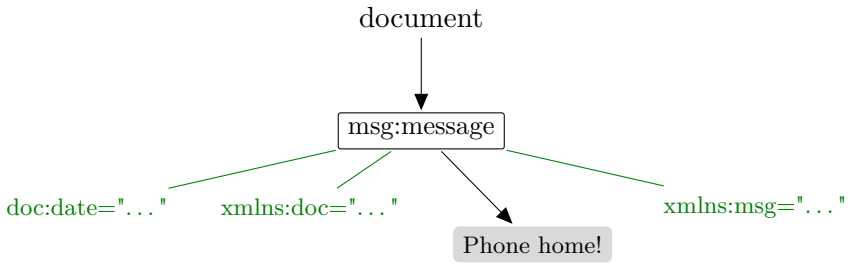
- Ein wohlgeformtes XML-Dokument, das die Namespace-Constraints erfüllt, hat ein *Information Set* (kurz *Infoset*).
- Das Dokument braucht dazu nicht gültig (bzgl. einer DTD) zu sein.
- Ein XML Information Set besteht aus einer Menge von *Information Items* (kurz *InfoItem*)
- Ein Information Item besitzt eine Menge von *Eigenschaften* (*Properties*, Name-Wert-Paare).
- Ein Information Set entspricht einem *Baum*, ein Information Item einem *Knoten* oder anderen Informationen darin.

Übersicht: Information Items

1. Document
2. Element
3. Attribute
4. Processing Instruction

5. Unexpanded Entity Reference
6. Character
7. Comment
8. Document Type Declaration
9. Unparsed Entity
10. Notation
11. Namespace

Information Items — Beispiel



Information Items

- Ein *Document*-InfoItem
- Ein *Element*-InfoItem
- Drei *Attribute*-InfoItems
- Drei *Namespace*-InfoItems
- Elf *Character*-InfoItems

Information Item: Element

Eigenschaften

namespace name Namespace-URI oder *No Value*

local name lokaler Teil des Elementnamens

prefix Präfix-Anteil des Elementnamens oder *No Value*

children Liste der Kind-InfoItems; möglich: Element, Processing Instruction, Unexpanded Entity Reference, Character, Comment, *aber nicht Attribute*.

attributes Menge von *Attribut*-InfoItems

namespace attributes Menge von *Attribut*-InfoItems, eins für jede Namespace-Deklaration der Form `xmlns="..."` oder `xmlns:name="..."`.

in-scope namespaces Menge von *namespace*-InfoItems, eins für jeden Namespace, der hier gültig ist.

base URI die *Base URI* des Elements

parent *document*- oder *element*-InfoItem

Information Item: Attribute

Eigenschaften

namespace name Namespace-URI oder *No Value*, falls nicht vorhanden

local name lokaler Teil des Attributnamens

prefix Präfix-Anteil des Attributnamens oder *No Value*, falls nicht vorhanden

normalized value normalisierter Attributwert

specified (Boolean) Kommt der Wert aus dem Start-Tag? (`false`: Default-Wert aus der DTD)

attribute type Typ des Attributs aus der DTD, *No Value*, falls kein Typ deklariert oder *Unknown*, falls DTD nicht (komplett) gelesen.

references für die Typen `IDREF`, `IDREFS`, `ENTITY`, `ENTITIES`, or `NOTATION`: Liste der *element*-, *unparsed entity*- oder *notation*-InfoItems, auf die der Wert verweist.

owner element *element*-InfoItem, das dieses Attribut enthält

Information Item: Document

Eigenschaften

children Liste der Kind-InfoItems, darunter genau ein *Element*-InfoItem

document element ein *Element*-InfoItem

notations Menge von *Notation*-InfoItems

unparsed entities Menge von *Unparsed Entity*-InfoItems

base URI die *Base URI* des Document-Entities

character encoding scheme Name der Zeichensatz-Kodierung

standalone „Standalone Status“ des Dokuments, aus der XML-Deklaration (*No Value* falls nicht vorhanden)

version Version aus der XML-Deklaration (*No Value* falls nicht vorhanden)

all declarations processed (Boolean) Hat der Prozessor die komplette DTD gelesen? Falls `false`, bekommen einige der anderen Eigenschaften den Wert *Unknown*.

Weitere Information Items

- ***Character***

  Eigenschaften: **character code**, **element content whitespace**, **parent**

- ***Namespace***

  Eigenschaften: **prefix**, **namespace name**

- ...

Aufgabe: Gültige Instanzen (bzgl. XML Schema)?

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="p" type="type1"/>
  <xsd:element name="q" type="type2"/>
  <xsd:complexType name="type1">
    <xsd:sequence>
      <xsd:element name="a" type="xsd:string"/>
      <xsd:element ref="q"/>
      <xsd:element name="c" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="type2">
    <xsd:choice>
      <xsd:element name="a" type="xsd:boolean"/>
      <xsd:element name="b" type="xsd:int" minOccurs="2" maxOccurs="9"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:schema>
```

```
<p>
  <a>a test</a>
  <q><b>2</b></q>
</p>
```

```
<p>
  <a>a test</a>
  <q><a>true</a></q>
  <c>51</c>
</p>
```

```
<p>
  <a>a test</a>
  <q><a>true</a><b>2</b><b>3</b><b>4</b><b>5</b></q>
  <c>a</c>
</p>
```

```
<p>
  <a>a test</a>
  <q><b>0</b><b>-1</b><b>2</b><b>3</b><b>4</b><b>5</b></q>
</p>
```

Aufgabe: Gültige Instanzen?

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="t" type="type1">
    <xsd:key name="key1">
      <xsd:selector xpath=".//n" />
      <xsd:field xpath="@i" />
    </xsd:key>
    <xsd:keyref name="keyref1" refer="key1">
      <xsd:selector xpath=".//n"/>
      <xsd:field xpath="@p" />
    </xsd:keyref>
  </xsd:element>

  <xsd:complexType name="type1">
    <xsd:choice>
      <xsd:element name="n" type="type1" maxOccurs="unbounded" />
      <xsd:element name="i" type="xsd:integer" />
    </xsd:choice>
    <xsd:attribute name="i" type="xsd:nonNegativeInteger" use="optional" />
    <xsd:attribute name="p" type="xsd:nonNegativeInteger" use="optional" />
  </xsd:complexType>

</xsd:schema>
```

```
<t>
  <n i="1">
    <n i="2" p="3">
      <i>2</i>
    </n>
  </n>
  <n i="3" p="1">
    <i>2</i>
  </n>
</t>
```

```
<t>
  <n i="12" p='12'>
    <n i="3">
      <i>4</i>
    </n>
  </n>
  <n i="3" p="12">
    <i>8</i>
  </n>
</t>
```

```
<t>
  <n i="6">
```

```
    <i>4</i>
  </n>
  <n i="12" p="6">
    <n i="15" p="4">
      <i>12</i>
    </n>
  </n>
</t>
```

```
<t>
  <n i="23">
    <n i="22" p="55">
      <i>-1</i>
    </n>
    <n i="4" />
  </n>
</t>
```

3.3 Beispiele

XML-Schema-Beispiele

XML Schema für ...

- XHTML 1.0[5]
- SVG[6]
- XML Schema[7]

4 18. Mai 2018

4.1 Aufgaben

4.2 Programmierprojekt

Unser Programmierprojekt

Aufgaben des Projekts

1. HTML5 und CSS

[5] http://www.w3.org/2002/08/xhtml/xhtml1-strict.xsd
[6] http://www.w3.org/TR/2002/WD-SVG11-20020108/SVG.xsd
[7] http://www.w3.org/TR/xmlschema-1/#normative-schemaSchema

2. Javascript
3. Server

Werkzeuge

Serverseitig verwenden wir

- Node.js[8] und
- Express[9], ein einfaches Web-Framework für Node.

Meilensteine

- *Zwischenabgabe* von Aufg. 1 und 2: Bestehen notwendig für die Zulassung. Feedback, ggf. Nachbesserung innerhalb einer Woche
- *Endabgabe* von Aufg. 1, 2 und 3: Bestehen notwendig für die Zulassung. Keine Nachbesserung!
- *Präsentation*: Bestehen notwendig für die Zulassung.

Beispiel: HTTP

Was macht ein Browser mit folgender URL? https://www.w3.org/Consortium/siteindex.html#technologies

- DNS-Auflösung / TCP-Verbindung
- Geben Sie einen Request in HTTP 1.1 an, den ein Browser erzeugen würde.
- Geben Sie auch den Header einer möglichen HTTP-Response hierzu an.

```
$ host www.w3.org
www.w3.org has address 128.30.52.100

$ telnet 128.30.52.100 80
Trying 128.30.52.37...
Connected to 128.30.52.37.
Escape character is '^]'.
GET /Consortium/siteindex.html HTTP/1.1
Host: www.w3.org


HTTP/1.1 200 OK
Date: Wed, 17 May 2017 11:26:20 GMT
Last-Modified: Thu, 16 Jun 2016 15:24:24 GMT
ETag: "8abf-53566d53c7a00"
Accept-Ranges: bytes
Content-Length: 35519
```

[8] http://nodejs.org/
[9] http://expressjs.com/

```
Vary: Accept-Encoding,upgrade-insecure-requests
Cache-Control: max-age=21600
Expires: Wed, 17 May 2017 17:26:20 GMT
P3P: policyref="http://www.w3.org/2014/08/p3p.xml"
Content-Type: text/html; charset=utf-8

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
...
<h2 id="technologies">W3C A to Z</h2>
...
</html>
```

```
GET /Consortium/siteindex.html HTTP/1.1
Host: www.w3.org
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.96 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
DNT: 1
Referer: https://www.w3.org/
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: en-US,en;q=0.8
Cookie: authorstyle=no
If-None-Match: "8abf-53566d53c7a00-gzip"
If-Modified-Since: Thu, 16 Jun 2016 15:24:24 GMT
```

(Tatsächlicher Request eines realen Browsers)

Request mit `If-Modified-Since`

```
$ telnet www.w3.org 80
Trying 128.30.52.100...
Connected to www.w3.org.
Escape character is '^]'.
GET /Consortium/siteindex.html HTTP/1.1
Host: www.w3.org
If-Modified-Since: Thu, 16 Jun 2016 15:24:24 GMT

HTTP/1.1 304 Not Modified
Date: Wed, 17 May 2017 11:44:52 GMT
ETag: "8abf-53566d53c7a00"
Expires: Wed, 17 May 2017 17:44:52 GMT
Cache-Control: max-age=21600
Vary: upgrade-insecure-requests

Connection closed by foreign host.
```

Caching verhindern

```
$ telnet www.w3.org 80
Trying 128.30.52.100...
Connected to www.w3.org.
Escape character is '^]'.
GET /Consortium/siteindex.html HTTP/1.1
Host: www.w3.org
Pragma: no-cache
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Wed, 17 May 2017 11:46:42 GMT
Last-Modified: Thu, 16 Jun 2016 15:24:24 GMT
ETag: "8abf-53566d53c7a00"
```

```
Accept-Ranges: bytes
Content-Length: 35519
Vary: Accept-Encoding,upgrade-insecure-requests
Cache-Control: max-age=21600
Expires: Wed, 17 May 2017 17:46:42 GMT
P3P: policyref="http://www.w3.org/2014/08/p3p.xml"
Content-Type: text/html; charset=utf-8

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Übungen zu TCP/IP und HTTP

Zum Auffrischen der Java-Kenntnisse und zum besseren Verständnis des Internets und der WWW-Transportebene:

1. (TCP) Stelle eine TCP-Verbindung zwischen zwei Java-Prozessen her, siehe Java TCP Sockets and Swing Tutorial[10]
2. (WWW-Client) Lade Webseiten aus einem Java-Prozess herunter, siehe The Java Tutorials: Connecting to a URL[11]
3. (WWW-Server) Installiere und konfiguriere den
   - Apache HTTP Server[12] und/oder den
   - nginx HTTP Server[13] und/oder
   - Node.js[14]

   für eine einfache Website mit statischen Inhalten.

Die Aufgabenteile 1 und 2 können auch sinnvoll in anderen Programmiersprachen (Python, Javascript, ...) gelöst werden.

4.3 Werkzeuge im Web-Browser

Werkzeuge im Web-Browser

Mozilla Firefox

- F12, Ctrl-Shift-I → Developer Tools (built-in)

Google Chrome / Chromium

- F12, Ctrl-Shift-I → DevTools (built-in)

[10] http://www.cise.ufl.edu/~amyles/tcpchat/
[11] http://java.sun.com/docs/books/tutorial/networking/urls/connecting.html
[12] http://httpd.apache.org/
[13] http://nginx.org/en/
[14] http://nodejs.org/

- Dokumentation[15]

- https://developer.mozilla.org/en-US/docs/Tools/Page_Inspector
- http://getfirebug.com/
- https://developer.chrome.com/devtools

4.4 Fragen

Fragen

- Was ist das Internet? Wie werden Daten transportiert? Wie werden Computer adressiert? Wozu dient das Schichtenmodell?
- Was ist ein Hypertext? Wo gibt es Hypertexte?
- Nenne die drei wichtigen Komponenten des WWW!
- Welche Aspekte könnten beim Design von URLs ein Rolle gespielt haben?
- Wo werden URLs verwendet?
- Wie funktioniert HTTP?
- Was sind persistente Verbindungen in HTTP und wozu braucht man sie?
- Wozu benötigt man einen HTTP-Proxy?
- Kann man HTML ohne HTTP einsetzen?
- Kann man HTTP ohne HTML einsetzen?

5 1. Juni 2018

5.1 Programmierprojekt

5.2 Termine

Prüfungsanmeldung

Prüfungsanmeldung

- Elektronisch in Basis, 1. bis 21. Juni 2018

[15] https://developers.google.com/web/tools/chrome-devtools/

5.3 Beispiele und Werkzeuge

Browser-Unterstützung / Tools

Welcher Browser unterstützt welche Features (HTML, CSS, JS)?

- http://caniuse.com/
- http://html5please.com/

Wie standardkonform ist mein Browser?

- http://html5test.com/

Feature Detection im Browser

- http://modernizr.com/

Modernizr runs quickly on page load to detect features; it then creates a JavaScript object with the results, and adds classes to the `html` element for you to key your CSS on.

5.4 Cascading Style Sheets (CSS)

CSS Selectors and CSS Examples

- MDN — CSS reference[16]
- http://css-tricks.com/examples/nth-child-tester/
- http://www.csszengarden.com/

CSS Transformations and CSS Transitions

Beispiel 5.1.
```
.box {
    border-style: solid;
    border-width: 1px;
    display: block;
    width: 100px;
    height: 100px;
    background-color: #0000FF;
    transition: width 2s, height 2s, background-color 2s,
                transform 2s;
}
```

[16] https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

```
.box:hover {
    background-color: #FFCCCC;
    width: 200px;
    height: 200px;
    transform: rotate(180deg);
}
```

Link zum Beispiel auf MDN[17]

5.5 Bootstrap

Bootstrap

Reale Webanwendungen — Anforderungen

- Schick anzuschauen, konsistent und *responsive*
- browserunabhängig
- viele Features (in HTML, CSS und JavaScript)
- Standards ändern sich laufend

Das Rad jedes Mal neu erfinden?

Bootstrap

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

- https://getbootstrap.com/
- ursprünglich bei Twitter entwickelt
- einheitliches Design für HTML-Elemente
- Design Patterns, *Grid-Layout* (basiert auf Flexbox)
- Komponenten: Navigation, Menüs, Knöpfe, Tabs, ...
- jQuery plugins: Nicht zwingend erforderlich, aber manche Komponenten funktionieren nur so
- Kann einen CSS-Preprocessor (Sass) verwenden. Variablenwerte können überschrieben werden, um Bootstrap zu personalisieren.

Bootstrap — CSS (Beispiele)

[17] https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions/Using_CSS_transitions

- Mobile first
- Containers
  - Use `.container` for a responsive fixed width container
  - Use `.container-fluid` for a full width container, spanning the entire width of your viewport.
- Grid System: responsive, up to 12 columns, predefined classes, mixins
- Typography, Tables, Forms, Buttons, Images, ...

6 8. Juni 2018

6.1 JavaScript

JavaScript

Themen

- Unobtrusive JavaScript
- Beispiele
- Vue.js
- ESLint
- jQuery (eine JavaScript Library)

JavaScript — Vorsicht!

Schlecht: Verhalten vermischt mit Inhalten

```
<button
  type="button"
  onclick="document.getElementById('xyz').style.color='red';">
    Click Me
</button>
```

Unobtrusive JavaScript

Also Verhalten und Inhalte sauber trennen:

```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    document.getElementById('testButton')
      .addEventListener('click', () => {
        document.getElementById('xyz').style.color = 'red';
      });
  });
</script>
...
<button type="button" id="testButton">Click Me</button>
...
<div id=xyz>I will become red.</div>
```

Noch besser ist, das Script aus einer separaten Datei zu laden:

```
<script src="main.js" defer></script>
```

Unobtrusive JavaScript

JavaScript selbst lernen

- Reguläre Ausdrücke
- Weitere Array- und String-Methoden
- Events (Keyboard, Maus, ...) im DOM

JavaScript üben

Beispiel 6.1 (Closure). `function f1(x) {`

```
  return y => x * y;
};

const f3 = f1(3);
const f5 = f1(5);

console.log(f3(11), f5(7));
```

Beispiel 6.2 (Iterables, Iteratoren). `const myList = [1, 2];`

```
const n1 = myList[Symbol.iterator]();
const n2 = myList[Symbol.iterator]();

n1.next();
n2.next();
n2.next();
n1.next();
n2.next();
```

Beispiel 6.3 (Iterables, Iteratoren). `> n1.next();`

```
{ value: 1, done: false }
> n2.next();
{ value: 1, done: false }
> n2.next();
{ value: 2, done: false }
> n1.next();
{ value: 2, done: false }
> n2.next();
{ value: undefined, done: true }
```

Beispiel 6.4 (Funktionen besitzen Properties). `function f (x, y, z) {`

```
  return x + y * z;
}

const g = (x, y) => 2 * x + y;

f.name;       // 'f'
f.length;     // 3
f.toString(); // 'function f(x, y, z) {\nreturn x + y * z;\n}'

g.name;       // 'g'
g.length;     // 2
g.toString(); // '(x, y) => 2 * x + y'

(x => x + 1).name; // ''
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function

Beispiel 6.5 (Currying — JS Style). `function curry(f, arity = f.length) {`

```
  return (...args) => {
    if (args.length >= arity) {
      return f(...args);
    } else {
      return curry((...moreargs) => f(...args, ...moreargs),
                   arity - args.length);
    }
  };
}

const add = curry((a, b) => a + b);

console.log(add(2)(3)); // 5
console.log(add(2, 3)); // 5

const add2 = add(2);
console.log(add2(3));   // 5;
```

6.2 Vue.js

Vue.js

What is Vue.js?

- A framework for building user interfaces (single-page applications)
- Incrementally adoptable
- The core library is focused on the view layer only.
- Models are plain JavaScript objects.

Features

- Templates
- Reactivity
- Components / custom elements
- Routing (vue-router); based on the current URL path
- Conditionals and Loops
- Handling User Input
- State Management (vuex)
- ...

Declarative Rendering

Beispiel 6.6 (HTML).
```
<div id="app">
  {{ message }}
</div>
```

Beispiel 6.7 (JavaScript).
```
const app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

Vue.js and Web Components

- Vue's component syntax is loosely modeled after the Web Components Spec.
- Vue components provide important features that are not available in plain custom elements, most notably
  - cross-component data flow,
  - custom event communication and
  - build tool integrations.

6.3 ESLint

ESLint

https://eslint.org/

ESLint is an open source JavaScript *linting* utility originally created by Nicholas C. Zakas in June 2013. Code linting is a type of static analysis that is frequently used to find *problematic patterns* or code that doesn't adhere to certain *style guidelines*.

6.4 jQuery

jQuery

jQuery

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.

Eigenschaften

- CSS3 compliant
- Cross-browser

Status

- jQuery hat hauptsächlich historische Relevanz.
- Heute: Vanilla JavaScript[18] oder Frameworks/Libraries wie Vue, React, Angular, ...
- CSS-Animationen effizienter (GPU) als jQuery-Animationen

Quellen

- http://jquery.com/
- Bear Bibeault and Yehuda Katz; *jQuery in Action* (Second Edition); Manning, 2010
- jQuery Event Basics[19]
- jQuery's Ajax-Related Methods[20]

[18] http://vanilla-js.com/
[19] http://learn.jquery.com/events/event-basics/
[20] http://learn.jquery.com/ajax/jquery-ajax-methods/

jQuery

Die `$()` Funktion

`$(selector)`

`jQuery(selector)`

- *The `$()` function returns a special JavaScript object containing an array of the DOM elements, in the order in which they are defined within the document, that match the selector.*
- *This object possesses a large number of useful predefined methods that can act on the collected group of elements.*
- *... this type of construct is termed a wrapper because it wraps the collected elements with extended functionality.*

Vgl. Chrome Console

- `$` Kurzform für `document.querySelector`
- `$$` Kurzform für `document.querySelectorAll`
- `$x` wertet XPath-Ausdrücke aus.

jQuery — Selectors

CSS-Selektoren oder *XPath*-Pfadausdrücke

`$("p a")`	Selects all `a` elements that are nested inside `p` elements.
`$("p:even")`	Selects all even `p` elements
`$("tr:nth-child(1)")`	Selects the first row of each table
`$("body > div")`	Selects direct `div` children of `body`
`$("a[href $= 'pdf']")`	Selects links to PDF files
`$("body > div:has(a)")`	Selects direct `div` children of `body`-containing links

Beispiel 6.8 (jQuery Chain).

```
$("div.notLongForThisWorld").hide().addClass("removed");
```

jQuery

Beispiel 6.9 (Button mit jQuery).

```
<html>
 <head>
```

```
    <title>jQuery Button Beispiel</title>
    <script src="jquery.min.js"></script>
    <script type="text/javascript" src="button-jquery.js"></script>
  </head>
  <body>
    <h1 id="xyz">Ein Text</h1>
    <button type="button" id="testButton">Click Me</button>
  </body>
</html>
```

Beispiel 6.10 (`button-jquery.js`).

```
$(document).ready(function(){
   $("#testButton").click(function(event){
        $("#xyz").css("color", "red");
   });
 });
```

jQuery

Beispiel 6.11 (umständlich).

```
let elements = $("div.fillMeIn");
for(let i = 0; i < elements.length; i++)
  elements[i].innerHTML =
    "I have added some text to a group of nodes";
```

Beispiel 6.12 (elegant ohne Schleife).

```
$("div.fillMeIn")
  .html("I have added some text to a group of nodes");
```

jQuery — weitere Features

- Event Handling
- Ajax (fetch ist besser!)
- Forms

7 15. Juni 2018

7.1 Programmierprojekt

Programmierprojekt

Zwischenabgabe

7.2 Promises

Asynchrones Programmieren

Vorteile

- blockiert nicht
- „natürliches Umfeld“ für eventbasiertes Programmieren
- in JavaScript keine Race-Conditions (single-threaded)

Probleme

- übliche Kontrollstrukturen funktionieren nicht

Patterns für Asynchrones Programmieren

- Callbacks, Events (Observer Pattern)
- Promises
- (Higher-Order) Funktionsbibliotheken

Versuch: sequenzielle Ausführung

```
function f1(x) {
  console.log('f1 starts', x)
  setTimeout(function () { console.log('f1 ends') }, 2000)
}

function f2(x) {
  console.log('f2 starts', x)
  setTimeout(function () { console.log('f2 ends') }, 1000)
}

f1(5)
f2(13)

// Output:
// f1 starts 5
// f2 starts 13
// f2 ends
// f1 ends
```

Beispiel 7.1 (Callbacks). `function f1(x, callback) {`

```
  console.log('f1 starts', x)
  setTimeout(function () {
    console.log('f1 ends')
    callback()
  }, 2000)
}

function f2(x, callback) {
  console.log('f2 starts', x)
  setTimeout(function () {
    console.log('f2 ends')
    callback()
  }, 1000)
}

f1(5, () => f2(13, () => null))

// f1 starts 5, f1 ends, f2 starts 13, f2 ends
```

Beispiel 7.2 (`async` / `await`). `function ratz(t) {`

```
  return new Promise(rslv => setTimeout(_ => rslv(t), t))
}

async function run(...args) {
  let t = 100
  for (const a of args) {
    await ratz(t)
    t += 100
    console.log(a)
  }
}

run(1, 2, 3, 4, 5).then(x => console.log('Test 1 done.'))
                  .catch(console.log)
run(6, 7, 8, 9, 10, 11).then(x => console.log('Test 2 done.'))
                       .catch(console.log)

// 1 6 2 7 3 8 4 9 5 10 Test 1 done. 11 Test 2 done.
```

Module und Promises in Node

Beispiel 7.3 (`fs.readFile` in Node.js). `import fs from 'fs'`

```
fs.readFile('testdata/file1.txt', (err, data) => {
  if (err) throw err
  console.log(data)
  console.log(String(data))
})
```

```
// <Buffer 48 65 6c 6c 6f 20 57 6f 72 6c 64 21 0a>
// Hello World!
```

Beispiel 7.4 (Promises: `fs.readFile` — mit `fs-extra`). `import fs from 'fs-extra'`

```
fs.readFile('testdata/file1.txt').then(buf => String(buf)).then(console.log)
// Hello world!

async function run () {
  try {
    const buf = await fs.readFile('testdata/fiel1.txt')
    console.log(String(buf))
  } catch (e) {
    console.log('Error:', e)
  }
}

run();  // Error
```

`fetch()` und Promises im Browser

Beispiel 7.5 (Promises). `const prefix = 'http://localhost:8087/testdata/'`

```
console.log(fetch(prefix + 'file1.txt'))
// Promise {<pending>}

fetch(prefix + 'file1.txt').then(console.log)
// Response { ... }

fetch(prefix + 'file1.txt').then(r => console.log(r.text()))
// Promise {<pending>}

fetch(prefix + 'file1.txt').then(r => r.text()).then(console.log)
// 'Hello World!'
```

Beispiel 7.6 (`async`/`await`). `const prefix = 'http://localhost:8087/testdata/'`

```
async function serial () {
  const r1 = await fetch(prefix + 'file1.txt')
  const s1 = await r1.text()
  const r2 = await fetch(prefix + 'file2.txt')
  const s2 = await r2.text()
  const r3 = await fetch(prefix + 'file3.txt')
  const s3 = await r3.text()
  return [s1, s2, s3]
}

serial().then(console.log)
```

```
// ['Hello World!', 'Content 2', 'Content 3']
```

Beispiel 7.7 (`Promise.all`). `const prefix = 'http://localhost:8087/testdata/'`

```
async function parallel () {
  const uris = ['file1.txt', 'file2.txt', 'file3.txt'].map(f => prefix + f)

  return Promise.all(uris.map(u => fetch(u)).map(p => p.then(r => r.text())))
}

parallel().then(console.log)

// ['Hello World!', 'Content 2', 'Content 3']
```

7.3 Fragen

Fragen?

8 22. Juni 2018

8.1 Aufgaben

Aufgaben

Aufgabe 11 g)

```
/descendant-or-self::*[ancestor-or-self::*[@xml:lang][1][@xml:lang='de']]
```

Aufgaben ...

Beispiel 8.1 (Aufgabe 10: zu erzeugendes HTML5). `<!DOCTYPE HTML>`
`<html>...</html>`

XSLT: DOCTYPE für HTML5 erzeugen

```
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="html" version="5.0" encoding="UTF-8" indent="yes" />

  <xsl:template match="/">
    <html>...</html>
  </xsl:template>

</xsl:stylesheet>
```

Funktioniert nur mit Saxon (oder XSLT 3.0)!

8.2 Sequence Types

Sequence Types

Beispiele 8.2.

- `xs:date`
- `attribute()?`
- `element()`
- `element(po:shipto, po:address)`
- `element(*, po:address)`
- `element(customer)`
- `node()*`
- `item()+`

9 29. Juni 2018

9.1 Programmierprojekt

Programmierprojekt

Endabgabe

...

Präsentation

- Alle Abgabepartner (1–2) müssen anwesend sein.
- Kurze Präsentation zu vorgegebenem Thema
- Ca. 3 Folien; maximal 10 Minuten Zeit
- Wer vortragen muss und zu welchem Thema vorgetragen wird, ist im Feedback auf eCampus angegeben.
- Termine werden über Foodle[21] vergeben. Jede Gruppe wählt bitte nur einen freien Termin!
  - Di, 10.7. 13-14 Uhr
  - Mi, 11.7. 11-12 Uhr
- Wer zu keinem der Termine kann, trägt in der Übung vor.
  - Fr, 6.7.
  - Fr, 13.7.

[21]

9.2 Aufgaben

Aufgaben

HTML5 erzeugen

```
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" version="5.0" encoding="UTF-8" indent="yes" />
  ...
</xsl:stylesheet>
```

9.3 Projektgruppe

Projektgruppe *Web-Technologien*, WS 2018/18

Details: nächste Woche

Interesse?

- Bitte Nachricht per E-Mail: stefan@iai.uni-bonn.de
- Gerne auch: Erwartungen, Themenvorschläge
- Keine Angst, es ist noch ganz unverbindlich

9.4 Programmieren mit XPath

Quicksort, siehe https://idea-instructions.com/quick-sort/

„Programmieren“ mit XPath (1)

Beispiel 9.1 (Quicksort: sortiere die Liste `$l`).

```
declare namespace my = "http://example.org";

declare function my:qsort($l as xs:anyAtomicType*)
                              as xs:anyAtomicType* {
   if (fn:count($l) gt 1) then
      ( my:qsort(fn:subsequence($l, 2)
           [. lt fn:subsequence($l, 1, 1)]),
        fn:subsequence($l, 1, 1) (: Pivotelement :),
        my:qsort(fn:subsequence($l, 2)
           [. ge fn:subsequence($l, 1, 1)])
      )
   else
      $l
};

my:qsort((3,5,2,8,9,1,4,7,6))
```

Beispiel 9.2 (Quicksort mit XQuery).

```
declare namespace my = "http://example.org";

declare function my:qsort($l as xs:anyAtomicType*)
                                as xs:anyAtomicType* {
   let $pivot := $l[1],
       $rest  := $l[position() gt 1]
   return
      if (fn:count($l) gt 1) then
         ( my:qsort($rest[. lt $pivot]),
           $pivot,
           my:qsort($rest[. ge $pivot])
         )
      else
         $l
};

my:qsort((3,5,2,8,9,1,4,7,6))
```

Programmieren mit XPath (2)

XPath in JavaScript

Introduction to using XPath in JavaScript[22]

DOM 3 XPath[23]

libxml2 — The XML C parser and toolkit of Gnome

http://www.xmlsoft.org/

- XML-Parser
- HTML4-Parser
- XInclude
- XPointer
- Teile von XML Schema
- XML Path Language (XPath) 1.0
- ...

Viele Sprachbindungen, z. B. `lxml` für Python
Beispiel 9.3 (ElementTree: XPath 1.0 in Python).

[22] https://developer.mozilla.org/en-US/docs/Introduction_to_using_XPath_in_JavaScript
[23] http://www.w3.org/TR/DOM-Level-3-XPath/xpath.html

```
Python 2.7.3 (default, Apr 14 2012, 08:58:41) [GCC] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from lxml import etree
>>> t = etree.parse("adressen.xml")
>>> t
<lxml.etree._ElementTree object at 0x7f8db880a3f8>
>>> t.xpath("descendant::adresse")
[<Element adresse at 0x7f8db87fd910>, <Element adresse at 0x7f8db87fd960>, <Element adresse at 0x7
>>> n = t.xpath("descendant::adresse[2]//nachname")
>>> n
[<Element nachname at 0x7f8db87fdc30>]
>>> n[0].text
'Mustermann'
>>>
```

9.6 Üben XSLT

Eine XSLT-Aufgabe

Stylesheet

```
<xsl:stylesheet xmlns:xsl="..." version="2.0">

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

  <xsl:template match="d|g"/>

  <xsl:template match="b">
    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="h"/>

  <xsl:template match="text()">
    <xsl:text>T</xsl:text>
  </xsl:template>

</xsl:stylesheet>
```

Eingabe

```
<a>
  <b>
    <c>
      <d>Hello</d>
      <e>world</e>
    </c>
    <f>Hi</f>
```

```
    <g>XML</g>
  </b>
  <h>
    <i>
      <j/>
      <k/>
    </i>
  </h>
</a>
```

Ausgabe?

Testlauf

```
$ alias saxon='java -cp /.../saxon9he.jar net.sf.saxon.Transform'
$ saxon -s:data.xml bsp1.xsl
<?xml version="1.0" encoding="UTF-8"?>
<x>TT<x>TT<x>T</x>T</x>T<x>T</x>TTTT</x>
```

`xsl:copy` — „flache Kopie“

Stylesheet

```
<xsl:stylesheet xmlns:xsl="..." version="2.0">
  <xsl:strip-space elements="*"/>

  <xsl:template match="c">
    <xsl:copy>
      <p>Bye</p>
      <xsl:apply-templates select="e"/>
    </xsl:copy>
  </xsl:template>

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

</xsl:stylesheet>
```

Eingabe

```
<c>
  <d>Hello</d>
  <e>world</e>
</c>
```

Ausgabe?

```
saxon -s:copy-example.xml copy-example.xsl
<c><p>Bye</p><e>world</e></c>
```

10 6. Juli 2018

10.1 Aufgaben

10.2 Vertiefung XSLT

Gruppierung mit `group-by`

Beispiel 10.1 (Eingabe). `<data>`

```
  <A color="red"/>
  <B color="green"/>
  <C color="red"/>
  <D color="blue"/>
  <E color="green"/>
  <F color="green"/>
  <G color="blue"/>
  <H color="red"/>
</data>
```

```
<xsl:for-each-group select="data/*" group-by="@color">
...
</xsl:for-each-group>
```

⇒ *Population*: (A, B, C, D, E, F, G, H)

Beispiel 10.2 (Stylesheet). `<xsl:stylesheet version="2.0"`

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

  <xsl:output method="xml" encoding="UTF-8" indent="yes"/>

  <xsl:template match="/">
    <result>
      <xsl:for-each-group select="data/*" group-by="@color">
        <group key="{current-grouping-key()}"
               color="{@color}"
               initial-item-name="{name(.)}">
          <xsl:for-each select="current-group()">
            <xsl:copy-of select="."/>
          </xsl:for-each>
        </group>
      </xsl:for-each-group>
    </result>
  </xsl:template>

</xsl:stylesheet>
```

Beispiel 10.3 (Ausgabe).
```
<result>
   <group key="red" color="red" initial-item-name="A">
      <A color="red"/>
      <C color="red"/>
      <H color="red"/>
   </group>
   <group key="green" color="green" initial-item-name="B">
      <B color="green"/>
      <E color="green"/>
      <F color="green"/>
   </group>
   <group key="blue" color="blue" initial-item-name="D">
      <D color="blue"/>
      <G color="blue"/>
   </group>
</result>
```

Groups: (A, C, H); (B, E, F); (D, G)

Initial Items: A, B, D ⇒ Reihenfolge der *Groups*, Kontext bei `xsl:sort`, ...

11 13. Juli 2018

11.1 Aufgaben

11.2 Termine

Termine

- Übung am 20. Juli:
  - *Aufgaben* 13–15 (XQuery, SAX und DOM)
  - *Fragestunde* zum gesamten Stoff
  - Anschließend: Erste Vorbesprechung zur ***Projektgruppe***

11.3 Projektgruppe

Projektgruppe *Web-Technologien*, WS 2018/19

Typisches Szenario

- Ein gemeinsames Projekt, Teilaufgaben ⇒ Untergruppen
- Single-Page HTML5-Anwendung, auch für mobile Geräte
- Server (Web-Service) mit Node.js, Express, usw.
- Grafische Ausgabe mit Canvas 2D oder WebGL

Thema

- Gemeinsam festlegen! Vorschläge?

Weitere Technologien (optional)

- XML-, oder NoSQL-Datenbank (*MongoDB*)
- XQuery, XSLT, ...

Interesse?

- Bitte Nachricht per E-Mail: stefan@iai.uni-bonn.de
- *Vorbesprechung: Freitag, 20.7.2018, ca. 15:15 Uhr im Hörsaal 7, HSZ*

11.4 Vertiefung XSLT

`xsl:function` — Rekursion mit Akkumulator

Beispiel 11.1 (Produkt ohne Akkumulator).
```
<xsl:function name="my:product" as="xs:double">
  <xsl:param name="l" as="xs:double*"/>
  <xsl:choose>
    <xsl:when test="empty($l)">
      <xsl:value-of select="1"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="$l[1] * my:product($l[position() gt 1])"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:function>
```

Aufruf: `my:product((4,5,3,2))`

Beispiel 11.2 (Produkt mit Akkumulator).
```
<xsl:function name="my:producta" as="xs:double">
  <xsl:param name="l" as="xs:double*"/>
  <xsl:param name="a" as="xs:double"/>
  <xsl:choose>
    <xsl:when test="empty($l)">
      <xsl:value-of select="$a"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="my:producta($l[position() gt 1], $a * $l[1])"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:function>
```

Aufruf: `my:producta((4,5,3,2), 1)`

`xsl:function` — Rekursion mit Akkumulator

Beispiel 11.3 (Eingabe).

```
  1 XSL Transformations (XSLT) Version 2.0
 27 XML Path Language (XPath) 2.0 (Second Edition)
 42 XQuery 1.0 and XPath 2.0 Functions and Operato
rs (Second Edition)
191 Cascading Style Sheets Level 2 Revision 1 (CSS
2.1) Specification
193 Media Queries
901 Of all possible committee reactions to any giv
en agenda item, the reaction that will occur is th
e one which will liberate the greatest amount of h
ot air.  -- Thomas L. Martin
```

Beispiel 11.4 (Ausgabe).

```
  1 XSL Transformations (XSLT) Version 2.0
 27 XML Path Language (XPath) 2.0 (Second Edition)
 42 XQuery 1.0 and XPath 2.0 Functions and Operators (Second Edition)
191 Cascading Style Sheets Level 2 Revision 1 (CSS2.1) Specification
193 Media Queries
901 Of all possible committee reactions to any given agenda item, the reaction that will occur is the one whi
```

Beispiel 11.5.

```
<xsl:transform version="2.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:my="http://iai.uni-bonn.de/web+xml">

  <xsl:output method="text"/>
  <xsl:param name="infile"/>

  <xsl:template name="start">
    <xsl:variable name="rawLines" as="xs:string*"
        select="tokenize(unparsed-text($infile), '\r?\n')"/>
    <xsl:for-each select="my:lines($rawLines, ())">
      <xsl:value-of select="."/>
      <xsl:text>&#10;</xsl:text>
    </xsl:for-each>
  </xsl:template>

  ...
```

Beispiel 11.6. ...

```
  <xsl:function name="my:lines" as="xs:string*">
    <xsl:param name="l" as="xs:string*"/>
    <xsl:param name="partial" as="xs:string?"/>
    <xsl:choose>
      <xsl:when test="empty($l)">
        <xsl:sequence select="$partial"/>
      </xsl:when>
      <xsl:when test="matches($l[1],'^(  \d| \d\d|\d\d\d) ')">
        <xsl:sequence select="$partial, my:lines($l[position() > 1], $l[1])"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:sequence
          select="my:lines($l[position() > 1], concat($partial, $l[1]))"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:function>
</xsl:transform>
```

Gruppierung mit group-starting-with

Beispiel 11.7 (Stylesheet). `<xsl:transform version="2.0"`

```
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="text"/>
  <xsl:param name="infile"/>

  <xsl:template name="start">
    <xsl:variable name="rawLines" as="element()*">
      <xsl:for-each select="tokenize(unparsed-text($infile), '\r?\n')">
        <l><xsl:value-of select="."/></l>
      </xsl:for-each>
    </xsl:variable>
    <xsl:for-each-group select="$rawLines"
         group-starting-with="l[matches(., '^(  \d| \d\d|\d\d\d) ')]">
      <xsl:value-of select="current-group()" separator=""/>
      <xsl:text>&#10;</xsl:text>
    </xsl:for-each-group>
  </xsl:template>
</xsl:transform>
```

XSLT-Beispiel

Beispiel 11.8 (Konvertierung XML nach Spreadsheet). `<xsl:stylesheet version="2.0"`

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

  <xsl:output method="text"/>

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

  <xsl:template match="*">
    <xsl:for-each select="*">
      <xsl:if test="position() gt 1">
        <xsl:text>&#9;</xsl:text>
      </xsl:if>
      <xsl:value-of select="."/>
    </xsl:for-each>
    <xsl:text>&#10;</xsl:text>
  </xsl:template>

</xsl:stylesheet>
```

11.5 Vertiefung XQuery

Beispiel: XQuery

Beispiel zu XQuery — Szenario

Durchschnittsgehälter verschiedener Mitarbeitergruppen

- Ausgeschiedene
- Neu eingestellte
- Länger beschäftigte

Beispiel 11.9 (Rohdaten: Gehälter — `emp-db.xml`). `<data>`

```
  <sal m="1306"><emp>A</emp><amnt>2200</amnt></sal>
  <sal m="1306"><emp>B</emp><amnt>2100</amnt></sal>
  <sal m="1306"><emp>C</emp><amnt>1800</amnt></sal>
  <sal m="1306"><emp>D</emp><amnt>1600</amnt></sal>
  <sal m="1307"><emp>A</emp><amnt>2240</amnt></sal>
  <sal m="1307"><emp>B</emp><amnt>2100</amnt></sal>
  <sal m="1307"><emp>F</emp><amnt>2000</amnt></sal>
  <sal m="1307"><emp>G</emp><amnt>1900</amnt></sal>
</data>
```

Beispiel 11.10 (Vorbereitende Deklarationen). `declare namespace saxon="http://saxon.sf.net/";`

```
declare option saxon:output "indent=yes";
declare option saxon:output "omit-xml-declaration=yes";

declare variable $month external;
declare variable $nextmonth external;

declare variable $sal as element()* :=
  doc("emp-db.xml")/data/sal;

...
```

Beispiel 11.11 (Erster Test: Gruppierung). `let $sal1 := (`

```
  for $e in $sal[@m eq $month]
  where not($sal[emp eq $e/emp][@m eq $nextmonth])
  return $e
)
let $sal2 := (
  for $e in $sal[@m eq $nextmonth]
  where not($sal[emp eq $e/emp][@m eq $month])
  return $e
)
let $sal12 := (
  for $e in $sal[@m eq $nextmonth]
  where $sal[emp eq $e/emp][@m eq $month]
  return $e
)
return
<result month="{$month}" nextmonth="{$nextmonth}">
  <title>Gehaltsgefälle?</title>
  <debug n="1">{$sal1/emp}</debug>
  <debug n="2">{$sal2/emp}</debug>
  <debug n="12">{$sal12/emp}</debug>
</result>
```

Beispiel 11.12 (Erster Test: Ausgabe). `$ xquery emp-query1.xql month=1306 nextmonth=1307`

```
<result month="1306" nextmonth="1307">
   <title>Gehaltsgefälle?</title>
```

```
   <debug n="1">
      <emp>C</emp>
      <emp>D</emp>
   </debug>
   <debug n="2">
      <emp>F</emp>
      <emp>G</emp>
   </debug>
   <debug n="12">
      <emp>A</emp>
      <emp>B</emp>
   </debug>
```

Beispiel 11.13 (Fertige Auswertung).
```
let $sal1 := (
  for $e in $sal[@m eq $month]
  where not($sal[emp eq $e/emp][@m eq $nextmonth])
  return $e
)
let $sal2 := (
  for $e in $sal[@m eq $nextmonth]
  where not($sal[emp eq $e/emp][@m eq $month])
  return $e
)
let $sal12 := (
  for $e in $sal[@m eq $nextmonth]
  where $sal[emp eq $e/emp][@m eq $month]
  return $e
)
return
<result month="{$month}" nextmonth="{$nextmonth}">
  <title>Gehaltsgefälle?</title>
  <avg n="ausgeschieden">{avg($sal1/amnt)}</avg>
  <avg n="neu">{avg($sal2/amnt)}</avg>
  <avg n="bleibend">{avg($sal12/amnt)}</avg>
</result>
```

Beispiel 11.14 (Auswertung: Ausgabe).
```
$ xquery emp-query2.xql month=1306 nextmonth=1307
<result month="1306" nextmonth="1307">
   <title>Gehaltsgefälle?</title>
   <avg n="ausgeschieden">1700</avg>
   <avg n="neu">1950</avg>
   <avg n="bleibend">2170</avg>
</result>
```

11.6 Üben XSLT

Noch eine XSLT-Aufgabe

Stylesheet

```
<xsl:stylesheet xmlns:xsl="..." version="2.0">
  <xsl:strip-space elements="*"/>
```

```
  <xsl:template
      match="*[count(ancestor::*) mod 3 eq 0]">
    <xsl:copy>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>

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

</xsl:stylesheet>
```

Eingabe

```
<a>
  <b>
    <c>
      <d>Hello</d>
      <e>world</e>
    </c>
    <f>Hi</f>
    <g>XML</g>
  </b>
  <h>
    <i>
      <j/>
      <k/>
    </i>
  </h>
</a>
```

Ausgabe?

```
$ saxon -s:data.xml bsp2.xsl
<a><d>Hello</d><e>world</e>HiXML<j/><k/></a>
```

Noch eine XSLT-Aufgabe

Stylesheet

```
<xsl:stylesheet xmlns:xsl="..." version="2.0">
  <xsl:strip-space elements="*"/>

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

  <xsl:template match="c/*">
```

```
    <xsl:apply-templates
      select="parent::*/following-sibling::*"/>
  </xsl:template>

  <xsl:template match="b">
    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="h"/>

  <xsl:template match="text()">T</xsl:template>
</xsl:stylesheet>
```

Eingabe

```
<a>
  <b>
    <c>
      <d>Hello</d>
      <e>world</e>
    </c>
    <f>Hi</f>
    <g>XML</g>
  </b>
  <h>
    <i>
      <j/>
      <k/>
    </i>
  </h>
</a>
```

Ausgabe?

```
$ saxon -s:data.xml bsp3.xsl
<a><c><f>T</f><g>T</g><f>T</f><g>T</g></c><f>T</f><g>T</g></a>
```

Eine XSLT-Aufgabe

Stylesheet

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="2.0">
  <xsl:strip-space elements="*"/>

  <xsl:template match="/">
    <x>
      <xsl:for-each-group select="descendant::*"
                          group-by="count(descendant::text())">
```

```
        <xsl:sort select="current-grouping-key()"/>
        <xsl:apply-templates
          select="current-group()[current-grouping-key() > 0]"/>
      </xsl:for-each-group>
    </x>
  </xsl:template>

  <xsl:template match="*[text()]">
    <xsl:copy/>
  </xsl:template>

</xsl:stylesheet>
```

Eingabe

```
<a>
  <b>
    <c>
      <d>Hello</d>
      <e>world</e>
    </c>
    <f>Hi</f>
    <g>XML</g>
  </b>
  <h>
    <i>
      <j/>
      <k/>
    </i>
  </h>
</a>
```

Ausgabe?

```
$ saxon -s:data.xml bsp4.xsl
<x><d/><e/><f/><g/><d/><e/><d/><e/><f/><g/><d/><e/><f/><g/></x>
```

Fragen

- Welche Knoten des Eingabedokuments besucht XSLT?
- Was ist ein Sequenzkonstruktor?
- Wie kann XSLT Elemente in das Ausgabedokument einfügen?
- Erläutere den Unterschied `xsl:copy` vs. `xsl:copy-of`
- Anwendungen von `xsl:copy` und `xsl:copy-of`?

11.7 XML-Datenbanken

XML-Datenbanksysteme

Einige XML-Datenbanksysteme

- eXist[24] (s. Vorlesung)
- Zorba[25]
- BaseX[26]

Ausprobieren!

- `http://www.exist-db.org/exist/apps/demo/index.html` (Demo)
- `http://try.zorba.io/` (Live Demo)

XQuery and XPath Full Text 1.0[27] W3C Recommendation 17 March 2011

11.8 Fragen

Fragen

- Welche Knoten des Eingabedokuments besucht XQuery?
- Welche Datentypen gibt es in XQuery?
- Auf welche Dokumente kann XQuery zugreifen?
- Wie kann XQuery Elemente in das Ausgabedokument einfügen?
- Ist XQuery berechnungsuniversell (Turing-vollständig)?

- Semantik von *XQuery Update*?
- Architektur eines XML-Datenbanksystems?

[24] `http://www.exist-db.org`
[25] `http://www.zorba.io/`
[26] `http://basex.org/`
[27] `http://www.w3.org/TR/xpath-full-text-10/`