SSD UE

Übung zu SSD4
Semistrukturierte Datenmodelle und XML

SS 2022, Übung 5

Abgabetermin: 21.04.2022
Abgabeform elektronisch

☐ **SSD41UE Traxler** **Name** Angelos Angelis **Aufwand in h** 9
☒ **SSD42UE Niklas**
☐ **SSD43UE Niklas** **Punkte** _____ **Kurzzeichen Tutor** _____

1. SAX: Entlassungsbrief (10 Punkte)

Gegeben ist das XML-Dokument Entlassungsbrief_SAX.xml. Erstellen Sie eine SAX-Implementierung, die aus dem Dokument eine kurze Zusammenfassung mit Patientennamen und die aktuellen Diagnosen (inkl. bis-Datum) ausgibt (Konsole). Die Anrede soll geschlechtsspezifisch gewählt und die Titel in der richtigen Reihenfolge (XML-Dokument) angegeben werden. Geben Sie nur jene Diagnosen aus, die relevant sind: Schließen Sie Diagnosen aus, die vor dem beschriebenen Aufenthalt abgeschlossen wurden, d.h. deren bis-Datum länger als das von-Datum des Aufenthalts zurückliegt, falls dieses vorliegt. Formatieren Sie die Ausgabe schlüssig (dh. nur wenn die angegebenen Elemente/Attribute vorhanden sind), sh. Beispiel-Ausgabe in Abbildung 1. Der SAX-Parser muss das Dokument vor der Verarbeitung nicht validieren.

```
Sehr geehrter Herr Dipl.Ing. Hofrat Konrad Anton Tneitap BA MA!
Es liegen folgende Diagnosen vor:
- Meniskus: Empyema gen. sin. post corpus alienum ligneum operat. (bis 2022-02-24)
- Zähneknirschen
```

Abbildung 1 - LÖSUNG

2. DOM: Entlassungsbrief (10 Punkte)

Erstellen Sie eine DOM-Implementierung, die aus dem Dokument Entlassungsbrief_DOM.xml die wichtigsten Informationen extrahiert und in einer Summary zusammenfasst und stellen sie diese in Form einer XML-Datei zur Verfügung. Die Struktur ist in Summary.dtd beschrieben. Enthalten sein sollten der Name (inkl. Titel) des Patienten, sowie relevante Diagnosen und deren Abschlussdatum. Schließen Sie jene Diagnosen aus, die vor dem beschriebenen Aufenthalt abgeschlossen wurden, d.h. deren bis-Datum länger als das von-Datum des Aufenthalts zurückliegt, falls dieses vorliegt.

- Kopieren Sie den gesamten Diagnose-Knoten in das neue Dokument und benennen Sie den Knoten entsprechend um.
- Um XPath mit Namespaces zu verwenden, benutzen Sie die Klasse NamespaceResolver.java und geben Sie mit .setNamespaceContext dem XPath-Objekt den ausgelesenen Namespace bekannt; verwenden Sie anschließend „entl“ als Präfix.
- Speichern Sie das neu erstellte Dokument als summary.xml ab und vergewissern Sie sich, dass es der vorgegebenen Struktur entspricht (siehe summary.dtd).

```
factory.setNamespaceAware(true);
factory.setValidating(true);
factory.setAttribute(
        "http://java.sun.com/xml/jaxp/properties/schemaLanguage",
        "http://www.w3.org/2001/XMLSchema");
factory.setAttribute(
        "http://java.sun.com/xml/jaxp/properties/schemaSource",
        "Entlassungsbrief.xsd");
```

Abbildung 1: Code-Ausschnitt zur Validierung des XML-Dokuments mit einer XSD-Datei

3. Transformieren eines DOM-Baumes mit XSLT (4 Punkte)

Gegeben ist ein umfangreicher Entlassungsbrief aus dem ELGA-Beispieldatensatz (`ELGA-023-Entlassungsbrief_aerztlich_EIS-FullSupport.xml`) mit einem zugehörigen XSL-Dokument zur visuellen Darstellung.

Erstellen Sie eine Java-Implementierung und führen Sie eine Transformation des Entlassungsbriefes durch. Setzen Sie dazu die Klasse TransformerFactory ein, um die Transformation mit Hilfe des Stylesheets durchzuführen. Die Klasse TransformerFactory dient zur Erzeugung von Transformer-Exemplaren, mit denen Dokumente transformiert werden können. Beim Starten ihrer Applikation soll auch die Stylesheet-Datei übergeben werden. Speichern Sie das transformierte XML-Dokument in der Datei `ELGA_Entlassungsbrief_aerztlich.html` ab.

1)

Entlassungsbrief

```
package tests.sax;

import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;
import java.io.File;

public class Entlassungsbrief {
    public static void main(String... args) {
        SAXParserFactory factory = SAXParserFactory.newInstance();
        factory.setValidating(false);
        try {
            SAXParser parser = factory.newSAXParser();

            File file = new File("xmlfiles/Entlassungsbrief_SAX.xml");

            parser.parse(file, new EntlassungsbriefHandler());
        } catch (Throwable e) {
            System.out.println("Exception Type: " + e.getClass().toString());
            System.out.println("Exception Message: " + e.getMessage());
            e.printStackTrace();
        }

    }
}
```

EntlassungsbriefHandler

```
package tests.sax;

import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import org.xml.sax.helpers.DefaultHandler;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;


public class EntlassungsbriefHandler extends DefaultHandler {

    String currentTag;
    Person person;
    boolean titlePos;
    Diagnosis diagnosis;

    @Override
    public void startElement(String uri, String localName, String qName,
Attributes attributes) throws SAXException {
        currentTag = qName;
        if (qName.equals("Patient")) {
            person = new Person();
```

```
            return;
        }
        if (qName.equals("Titel")) {
            titlePos = attributes.getValue("position").equals("vor");
            return;
        }
        if (qName.equals("Diagnose")) {
            diagnosis = new Diagnosis();
            final String to = attributes.getValue("bis");
            if (to != null) {
                final LocalDate parse = LocalDate.parse(to);
                LocalDate nowHalfYear = LocalDate.now().minusDays(180);
                if (nowHalfYear.isAfter(parse)) {
                    diagnosis = null;
                    return;
                }
                diagnosis.setTo(parse);
            }
            final String from = attributes.getValue("von");
            if (from != null) {
                diagnosis.setFrom(LocalDate.parse(from));
            }

        }
    }

    @Override
    public void endElement(String uri, String localName, String qName) throws
SAXException {
        currentTag = null;
    }

    @Override
    public void characters(char[] ch, int start, int length) throws
SAXException {
        if (currentTag == null) {
            return;
        }
        String s = new String(ch, start, length);
        if (currentTag.equals("Vorname")) {
            person.addFirstname(s);
            return;
        }
        if (currentTag.equals("Nachname")) {
            person.setLastname(s);
            return;
        }
        if (currentTag.equals("Titel")) {
            if (titlePos) {
                person.addTitleFront(s);
            } else {
                person.addTitleBack(s);
            }
            return;
        }
        if (currentTag.equals("Geschlecht")) {
            person.setGender(s.equals("weiblich"));
```

```
            return;
        }
        if (currentTag.equals("Diagnose") && diagnosis != null) {
            diagnosis.setText(s);
            person.addDiagnosis(diagnosis);
            diagnosis = null;
        }
    }

    @Override
    public void endDocument() throws SAXException {
        System.out.println(person);
    }

    private static class Person {
        boolean gender;
        List<String> firstnames = new ArrayList<>();
        String lastname;
        List<String> titlesFront = new ArrayList<>();
        List<String> titlesBack = new ArrayList<>();
        List<Diagnosis> diagnoses = new ArrayList<>();


        public void setGender(boolean gender) {
            this.gender = gender;
        }

        public void setLastname(String lastname) {
            this.lastname = lastname;
        }

        public void addTitleFront(String title) {
            titlesFront.add(title);
        }

        public void addTitleBack(String title) {
            titlesBack.add(title);
        }

        public void addFirstname(String firstname) {
            firstnames.add(firstname);
        }

        public void addDiagnosis(Diagnosis d) {
            diagnoses.add(d);
        }

        @Override
        public String toString() {
            return "Sehr geehrte" +
                    (gender ? " Frau " : "r Herr ") +
                    String.join(" ", titlesFront) +
                    (titlesFront.isEmpty() ? "" : " ") +
                    String.join(" ", firstnames) +
                    " " +
                    lastname +
                    " " +
```

```
                    String.join(" ", titlesBack) +
                    "\n" +
                    "Es liegen folgende Diagnosen vor:\n" +
                    diagnoses
                            .stream()
                            .map(Object::toString)
                            .collect(Collectors.joining("\n")
                            );
        }
    }

    private static class Diagnosis {
        LocalDate from;
        LocalDate to;
        String text;

        public void setFrom(LocalDate from) {
            this.from = from;
        }

        public void setTo(LocalDate to) {
            this.to = to;
        }

        public void setText(String text) {
            this.text = text.trim();
        }

        @Override
        public String toString() {
            return text + " " + (to != null ? ("(bis " + to + ")") : "");
        }
    }
}
```

```
Sehr geehrter Herr Dipl.Ing. Hofrat Konrad Anton Tneitap BA MA
Es liegen folgende Diagnosen vor:
Meniskus: Empyema gen. sin. post corpus alienum ligneum operat. (bis 2022-02-24)
Zähneknirschen
bekannt rezidivierende Rückenschmerzen (bis 2022-01-31)
```

2)

```
package tests.dom;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
```

```
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;
import java.io.File;
import java.io.IOException;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

public class Entlassungsbrief {
    private static final String XMLFILES = "xmlfiles/";

    public static void main(String... args) {
        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        factory.setNamespaceAware(true);
        factory.setValidating(true);

factory.setAttribute("http://java.sun.com/xml/jaxp/properties/schemaLanguage"
, "http://www.w3.org/2001/XMLSchema");

factory.setAttribute("http://java.sun.com/xml/jaxp/properties/schemaSource",
"Entlassungsbrief.xsd");
        try {
            DocumentBuilder builder = factory.newDocumentBuilder();
            File inFile = new File(XMLFILES + "Entlassungsbrief_DOM.xml");
            Document entlassungsbriefDoc = builder.parse(inFile);
            Document summaryDoc = builder.newDocument();
            Element summary = summaryDoc.createElement("summary");
            summaryDoc.appendChild(summary);

            Element fullname = summaryDoc.createElement("fullname");
            summary.appendChild(fullname);

            XPathFactory xPathFactory = XPathFactory.newInstance();
            XPath xPath = xPathFactory.newXPath();
            xPath.setNamespaceContext(new
NamespaceResolver(entlassungsbriefDoc));
            final String query = "//entl:Titel[@position='%s']";


            NodeList titlePre = (NodeList) xPath
                    .compile(String.format(query, "vor"))
                    .evaluate(entlassungsbriefDoc, XPathConstants.NODESET);
            List<String> titlesPreList = new ArrayList<>();
            for (int i = 0; i < titlePre.getLength(); i++) {
                titlesPreList.add(titlePre.item(i).getTextContent());
            }
```

```
            NodeList titlePost = (NodeList) xPath
                    .compile(String.format(query, "nach"))
                    .evaluate(entlassungsbriefDoc, XPathConstants.NODESET);
            List<String> titlesPostList = new ArrayList<>();
            for (int i = 0; i < titlePost.getLength(); i++) {
                titlesPostList.add(titlePost.item(i).getTextContent());
            }

            NodeList firstnames =
entlassungsbriefDoc.getElementsByTagName("Vorname");
            List<String> firstnamesList = new ArrayList<>();
            for (int i = 0; i < firstnames.getLength(); i++) {
                firstnamesList.add(firstnames.item(i).getTextContent());
            }

            fullname.appendChild(
                    summaryDoc.createTextNode(
                            String.join(" ", titlesPreList)
                                    + (titlesPreList.isEmpty() ? "" : " ") +
String.join(" ", firstnamesList)
                                    + " " +
entlassungsbriefDoc.getElementsByTagName("Nachname").item(0).getTextContent()
                                    + (titlesPostList.isEmpty() ? "" : " ") +
String.join(", ", titlesPostList)
                    )
            );

            // Copy Diagnoses Node
            Node diagnoses =
entlassungsbriefDoc.getElementsByTagName("Diagnosen").item(0);
            Node diagnosesClone = summaryDoc.importNode(diagnoses, true);
            summary.appendChild(diagnosesClone);

            // Rename diagnoses
            summaryDoc.renameNode(diagnosesClone, null, "diagnoses");
            LocalDate nowHalfYear = LocalDate.now().minusDays(180);

            // Rename Children of diagnoses
            NodeList diags = summaryDoc.getElementsByTagName("Diagnose");
            int i = 0;
            while (i < diags.getLength()) {
                Node diag = diags.item(i);
                Node to = diag.getAttributes().getNamedItem("bis");

                if (to != null
                        &&
nowHalfYear.isAfter(LocalDate.parse(to.getTextContent()))) {
                    diagnosesClone.removeChild(diag);
                } else {
                    summaryDoc.renameNode(diag, null, "diag");
                    i++;
                }
            }

            String fileName = "summary.xml";
            File file = new File(XMLFILES + fileName);
            TransformerFactory transformerFactory =
```

```
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            transformer.setOutputProperty("doctype-system", "summary.dtd");
            transformer.setOutputProperty("indent", "yes");
            transformer.transform(new DOMSource(summaryDoc), new
StreamResult(file));
        } catch (ParserConfigurationException | SAXException | IOException |
TransformerException | XPathExpressionException e) {
            e.printStackTrace();
        }
    }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE summary
  SYSTEM "summary.dtd">
<summary>
   <fullname>Dipl.Ing. Hofrat Konrad Anton Tneitap BA, MA</fullname>
   <diagnoses>
      <diag bis="2022-02-24"
            code="M25.46"
            status="abgeschlossen"
            von="2021-01-11">Meniskus: Empyema gen. sin. post corpus alienum ligneum operat.</diag>
      <diag code="G47.63" status="offen" von="2019-09-19">Zähneknirschen</diag>
      <diag bis="2022-01-31"
            code="M54.9"
            status="abgeschlossen"
            von="2018-01-31">bekannt rezidivierende Rückenschmerzen</diag>
   </diagnoses>
</summary>
```

```
File C:\Users\Angelos Angelis\Desktop\CodeSpace\Semester_4\XML\UE5\SSD4-Uebung05-06-AuszuarbeiteteBeispiele\xmlfiles\summary.xml is valid.
```