6. Unit: Web Services

Exercise 6.1 (Web Services: tomcat)

Install tomcat. You can do this on your own computer, and/or in the CIP Pool in your own account. Starting it makes it available under the URL localhost:8080 from the browser.

Copy the .war file for the XQuery Demo Web Service into tomcat/webapps and adapt the configuration (and restart tomcat again) to check whether it basically works.

Exercise 6.2 (Web Services: Stream Processing)

Write a Web Service that reads an XML document from the HTTP input (request) and returns (into the response stream) an XML document that has the same structure, but all element and attribute names are backwards. E.g., for

```
<country car_code='D'>
  <name>Germany</name><population year="2011">80219695</population>
</country>
```

the response is

```
<yrtnuoc edoc_rac='D'>
  <eman>Germany</eman><noitalupop raey="2011">80219695</noitalupop>
</yrtnuoc>
```

It should do this by streaming – i.e. whenever something is read, the result is immediately added to the answer. Log every action to the System.out to illustrate the proceeding.

Write a Java program that calls this Web Service, e.g., using mondial.xml (or any other XML data). It should also log its activity (sending and also receiving the answer) to System.out.
For the writing of the xml to the request, consider the output method of JDOM (which will not log every step, but should be good for seeing that already during writing, the response is received), and a SAX/StAX writer.

Are there documents where the response is the same as the document?

Exercise 6.3 (Application: An E-Exams System)

The topic of this exercise is to experiment with some features that are/would be useful for an E-Exams system like Ilias. Not to build a better Ilias, but to focus on some features (that partially, like printing and uploading exercises) are even not present in Ilias.

- Basic Functionality
  - generate a simple students' interface: students fill in simple text form fields, upload files or check multiple-choice questions. "Submit" button sends to WebService.
    (as there are multiple students: have a simple login form and a form field with the student's ID which is pre-filled when the page is then created, or use cookies if you like)
  - generate a simple correctors' interface: corrector fills in a simple text form field and assigns points. "Submit" button sends to WebService.
  - ... build on the source code of the XQuery servlet which already contains forms (and which integrates saxon, see below)
- Printing functionality:
  - generate LaTeX directly
  - generate DocBook

  - generate XSL-FO

  A process describing how DocBook can be converted to XSL-FO and then to pdf is described here:
  `https://stackoverflow.com/questions/2615002/how-to-generate-pdf-from-docbook-5-0/4728751`

- Exam design/input functionality
  - Source code in LaTeX (using the LaTeX DBIS exam package)
  - use tex4ht package to map to XHTML (mzlatex generates XHTML with MathML)
  - XSL stylesheet that quickly (and easy to modify) turns XHTML into an XML format for exercises (number/index, title, points, body, solution)
  - Java program: "upload" exercises via HTTP to the exam system Web Service.
    Java program must open HTTP connection to server; response only "ok".
    QTI XML Schema: `https://www.imsglobal.org/sites/default/files/spec/qti/v3/bind/index.html`
- Specific functionality for exams on certain topics, e.g. an XML exam.
  (the source code of the XQuery Web Service, that integrates saxon can be used)
  - Exercise with an XML fragment (with code input or file upload) stores the XML,
  - Exercise with a DTD stores the DTD, runs validation and outputs messages,
  - Exercises with queries or XSL run the code immediately
- Mockup service simulating FlexNow:
  - create exam, students register for exam,
  - Ilias-Service asks for participants (request: XML, response: XML), generates exams,
  - Ilias-Service sends grades to FlexNow.
- Discuss the pros and cons of implementing something like Ilias or FlexNow via a Web Service based on JAXB.