4. Unit: XML Processing with Java (II)

Exercise 4.1 (XML Digester) Use the XML Digester in a Java class that creates an internal data structure about seas, rivers and their tributaries (including lakes) and generates an output similar to that in Exercise 2.1.
Additionally, this output should show for every river the sum of the length of all rivers flowing into it (directly or indirectly).

Exercise 4.2 (XML/JAXB/DOM/XSLT) Application: Personal schedule

This exercise will be reused for a Web Service.

Consider an XML structure for representing a personal schedule ("Terminkalender") with entries (using xsd:dateTime):

```
<?xml version="1.0" encoding="UTF-8"?>
<schedule name="John Doe">
  <year n="2016">
    <month n="12">
      <day n="7">
        <entry starttime="16:00:00" duration="PT02H">
          <name>XML Lab</name>
          <description>weekly course meeting</description>
          <location>IFI 1.101</location>
        </entry>
      </day>
    </month>
    <month n="12">
      <day n="8">
        <entry starttime="11:00:00" duration="PT50M">
          <name>Discussion MSc Thesis X.Y.</name>
        </entry>
      </day>
    </month>
  </year>
</schedule>
```

- Each schedule contains zero or more year elements.
- Each year element contains zero or more month elements.
- Each month element contains zero or more day elements.
- Each day element contains zero or more entry elements.
- Each entry element, describes a certain date, containing information about starting time, duration, location and a textual description of the date.
- Appointments do not span over more than one day.
- The month and day elements appear in their correct temporal order inside the document.

The Exercise consists of the following steps:

a) Design an XML Schema (use the dateTime datatype from XML Schema), and validate an example instance.

b) Generate basic classes and interfaces for the schedule application, using the JAXB Schema binding compiler on your XML schema.

c) Unmarshal your example schedule.xml file into memory using JAXB.

d) Write a method insertEntry(date, time, duration, title) that inserts a new appointment into the internal schedule. If the appointment is collision-free, it should be inserted, otherwise the method should return "false" zurückgegeben.

e) Marshal the in-memory schedule into an in-memory JDOM Tree, again using JAXB.

f) Write an XSLT styhesheet that transforms an XML schedule instance (which can be given as a file or an JDOM instance) into an HTML output presentation of the appointments if the given month (HTML table or list).
Write a method that applies the stylesheet directly to the JDOM tree and outputs the result into a file (use e.g. the javax.xml.transform classes).

Notes:

- The command: "xjc -help" issued on the command line displays usage and possible options.
- At `/afs/informatik.uni-goettingen.de/course/xml-prakt/xml/JAXB-README` you find an explanation for installing, testing and using JAXB.
- The Java WebService Tutorial at http://java.sun.com/webservices/tutorial.html provides some help for working with JAXB.

Exercise 4.3 (Calexit in JAXB)

- Do the Calexit exercise using JAXB.
(`mondial.xsd` is available on the Web site)
- Why is it not necessarily a good idea to do it with the Digester?