1. XML Syntax and DTDs

The goal of this lab is

1. to try out tools for editing XML
2. to develop Document Type Definitions (DTDs) and to validate documents against DTDs.

1.1 XML-Syntax: MathML and XHTML

You are asked to write

- a MathML document with presentation markup for a formula given below,
- create a simple XHTML document, and
- embed the MathML into the XHTML so that the entire document is valid.

1.1.1 MathML

MathML is a markup language for mathematics. It has two sublanguages, one for specifying the presentation of formulas and another one for representing the content or logical structure of formulas. In this lab, we will only care about the presentation language.
Your task is to write markup for presenting the following formula:

$$(x^{a-1} + 3) \times z$$

To do this, you will only need a rudimentary understanding of how MathML works, which you can grasp quickly from these sources:

- Wikipedia entry on MathML:

  `http://en.wikipedia.org/wiki/MathML`

- A Beginner's Guide to MathML:

  `http://danielscully.co.uk/web/articles/mathml/`

- MathML Presentation for the Impatient:

  `http://www.xmlmind.com/tutorials/MathML/index.html`

Your code should finally be validated as XML meeting the MathML standard. To check it, submit it to the W3C validation service at

```
http://validator.w3.org/
```

Start your XML file with the prologue

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE math PUBLIC "-//W3C//DTD MathML 2.0//EN"
     "http://www.w3.org/Math/DTD/mathml2/mathml2.dtd">
```

The prologue tells the validator which is the DTD that specifies MathML and where it can be found on the Web.

Note as well that the root element of each MathML document must be `<math>`.

Try two different approaches to edit the file:

- with an editor of your choice
- with Eclipse IDE with the XML perspective.

To make it simple, first edit a part of the formula in your editor, then finalize it with Eclipse.

1.1.2 XHTML

Create an XHTML page with embedded MathML. It should have the prologue

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1 plus MathML 2.0//EN"
          "http://www.w3.org/Math/DTD/mathml2/xhtml-math11-f.dtd">
```

Your html page should contain a title and a heading. Add the attribute value `display="block"` to the math element.

Edit the file with:

- an editor of your choice or
- Eclipse IDE with the XML perspective or
- XMLQuire, which runs under Windows or
- any other XML editor you may find.

Load your page into a browser. Check whether there are differences between different browsers.

1.2 Document Type Definitions

In these two exercises you are asked to write DTDs. Again, use Eclipse with the XML perspective as the editing tool.

1.2.1 A DTD for Recipes

Retrieve the document `recipes.xml` from the course website. Write a document type description `recipes.dtd` that captures this document as tightly as possible. This means,

- the document should be valid with respect to the DTD;
- if in any place you constrain the DTD further, then the document does not satisfy the DTD any more.

Constraining the DTD further means, for instance, turning an expression `element*` into `element+`, `element?` or `element`. Another example of constraining the DTD further is to drop an alternative in an "OR" expression.
Connect your document to the DTD. Try both approaches of making it an internal and an external DTD.
If you have time later on, modify your DTD so that it captures the document `recipes-eleme.xml`, which is a variant of `recipes.xml` that has all information in its elements.

1.2.2 A DTD for a Student List

Suppose you need to define a structure for XML documents that list students in the Faculty of Computer Science. Consider the following requirements. (To solve this exercise properly, you need to define attributes and their properties. To understand how to use attributes, you may want to have a look at the lecture slides on DTDs.)

- A `studentlist` consists of one or more student entries.
- A student has a matriculation number by which they can be uniquely identified. A student entry consists of the student's name, the study programme in which the student is enrolled, and the courses the student has taken so far. For each student, we need to record as well the year when the student enrolled.
- A student's name consists of the first name, the middle name, which is optional, and the last name.
- Study programs are "BSc in CS and Engineering", "BSc in Applied CS", "Master in CS", and "PhD in CS".
- Each course entry contains the name of the course. Courses are undergraduate (Bachelor) courses, postgraduate (Master) courses, or both. A course can also be mandatory or optional.
- A course entry contains the title of the course. We want to record as well whether the student has already received a course work mark for the course and, if so, the mark. Moreover, we want to record whether the student has already taken an exam for the course and, if so, what was the outcome of the exam.

Tasks:

1. Write a DTD that defines the documents of type `studentlist` according to the requirements above.

2. Give an instance of a `studentlist` document that is *valid* with respect to your DTD.

3. Give an instance of a `studentlist` that is *invalid* with respect to your DTD. Make sure that the instance contains only element and attribute names introduced in the DTD. Explain why it is invalid.