Web Engineering

Assignment 1

Daniel Kostić, Guilherme Fial, Igor Fedotov, Stefan Vujović

Exercise 1: To which category or categories of the web applications belongs “Klips” https://klips.uni-koblenz-landau.de/ . (1 points)

If we follow the classification described here, which specifies the following categories:

1. Document Centric (Static homepage,web radio,company web site)
2. Interactive (Virtual exhibition, news site, travel planning)
3. Transactional (online banking, shopping, booking system)
4. Workflow based (E government, B2B solution)
5. Collaborative (chat room, E learning platform, P2P-services)
6. Portal oriented (community portal, online shopping mall, business portal)
7. Ubiquitous (customized services, location aware services, Multi platform delivery)
8. Semantic (Knowledge management,syndication, recommender system)
9. Social web (logs, collaborative filtering, Virtual shared workplace)

We can easily skip the last three categories, as “Klips” does not have any advanced functionalities like recommendation systems, multi platform delivery, location awareness or support for interaction between users. The same goes for the first, fourth and fifth category, because it has more functionalities than Document Centric applications do. It is not workflow based, because it does not have functionalities that track progress, help with automation of processes etc. It is not collaborative since it does not support any kind of collaboration or communication between students.

We would assign “Klips” to the category of **Portal Oriented** web applications, since it allows users to find information about subjects of their interest and register for them. It could be considered as a community because it enables finding information about other members. It also supports planning as it displays a schedule of taken classes which also makes it **Interactive**. Since it allows the user to update underlying content (register for a course) it also fits into the **Translactional** category.

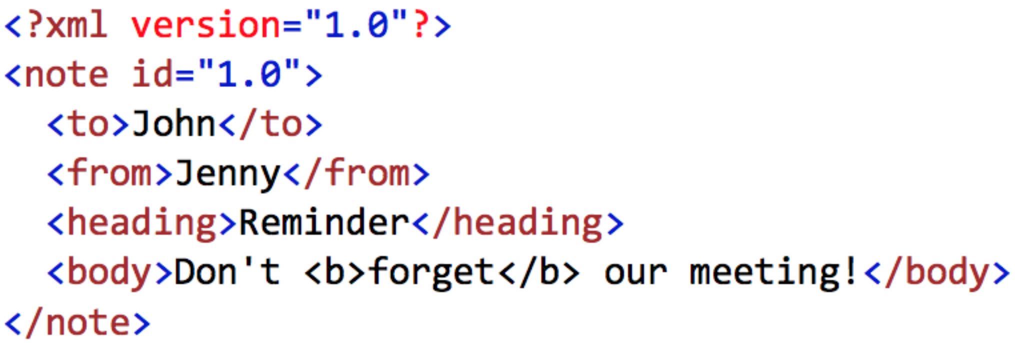
Exercise 2: What are the differences between Dom and SAX parsers? Please explain (2 differences). (2 points)

Difference 1: Where the DOM operates on the document as a whole, SAX parsers operate on each piece of the XML document sequentially. With SAX you naturally cannot modify the structure of the XML tree, because you never have it in hand as a whole. SAX reads the XML file from top to bottom and backward navigation is not possible. With DOM backward and forward search is possible for searching the tags and evaluation of the information inside the tags. So this gives the ease of navigation.

Difference 2: **Memory usage.** A **SAX parser** only needs to report each parsing event as it happens, and normally discards almost all of that information once reported (it does, however, keep some things, for example a list of all elements that have not been closed yet, in order to catch later errors such as end-tags in the wrong order). Thus, the minimum memory required for a SAX parser is proportional to the maximum depth of the XML file (i.e., of the XML tree) and the maximum data involved in a single XML event (such as the name and attributes of a single start-tag, or the content of a processing instruction, etc.).

This much memory is usually considered negligible. A **DOM parser**, in contrast, has to build a tree representation of the entire document in memory to begin with, thus using memory that increases with the entire document length. This takes considerable time and space for large documents (memory allocation and data-structure construction take time). The compensating advantage, of course, is that once loaded *any* part of the document can be accessed in any order.

Exercise 3: Provide the sequence of events that would be handled when parsing the XML document below using a SAX parser. (3 points)



```
<?xml version="1.0"?>
<note id="1.0">
  <to>John</to>
  <from>Jenny</from>
  <heading>Reminder</heading>
  <body>Don't <b>forget</b> our meeting!</body>
</note>
```

startDocument
startElement note, attribute (id, 1.0)
startElement to
characters John
endElement to
startElement from
characters Jenny
endElement from
startElement heading
characters Reminder
endElement heading
startElement body
characters Don’t
startElement b
characters forget
endElement b
characters our meeting!
endElement body
endElement note
endDocument

Exercise 4: Provide a DTD for a XML document, which describes a student semester plan. Express the following rules in DTD.
A student semester plan contains:

- The **personal information** of the student, namely **name, family, birthday (provided as day, month, year)**, and **address (provided as street, number, code, city)**.
- The **educational information, namely semester (mandatory), and field of study (optional)**.
- The registered course(s) and their relevant information, namely **name of the course, time, and room**.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE semester_plan [
<!ELEMENT semester_plan (personal,educational,registered_courses)>
<!ELEMENT personal (name,family,birthday,address)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT family (#PCDATA)>
<!ELEMENT birthday (day, month, year)>
<!ELEMENT day (#PCDATA)>
<!ELEMENT month (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT address (street, number, code, city)>
<!ELEMENT street (#PCDATA)>
<!ELEMENT number (#PCDATA)>
<!ELEMENT code (#PCDATA)>
<!ELEMENT city (#PCDATA)>
<!ELEMENT educational (semester,field_of_study?)>
<!ELEMENT semester (#PCDATA)>
<!ELEMENT field_of_study (#PCDATA)>
<!ELEMENT registered_courses (course*)>
<!ELEMENT course (course_name,course_time,course_room)>
<!ELEMENT course_name (#PCDATA)>
<!ELEMENT course_time (#PCDATA)>
<!ELEMENT course_room (#PCDATA)>
]>
<semester_plan>
        <personal>
                <name>Daniel</name>
                <family>Kostic</family>
                <birthday>
                        <day>16</day>
                        <month>02</month>
                        <year>1994</year>
                </birthday>
                <address>
                        <street>Muhlental</street>
                        <number>28</number>
                        <code>56077</code>
                        <city>Koblenz</city>
                </address>
        </personal>
        <educational>
                <semester>First</semester>
                <field_of_study>Web science</field_of_study>
```

```
    </educational>
    <registered_courses>
        <course>
            <course_name>Intro to web science</course_name>
            <course_time>Mondays</course_time>
            <course_room>K101</course_room>
        </course>
        <course>
            <course_name>Web engineering</course_name>
            <course_time>Thursday</course_time>
            <course_room>K101</course_room>
        </course>
    </registered_courses>
</semester_plan>
```