Exercise #1

Group GOLF:

Anish Girija Sivarajan, Md. Shohel Ahamad,

Slobodan Kocevski, Veselin Stefanov

22.11.2016

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

Answer:

Interactive – the app provides documents related to users' data. Weekly timetables are constructed by using students' course selection.

Portal-oriented – users have access to various types of information like course descriptions, dates, useful links

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

DOM produces a whole model of a XML document and SAX (simple API for XML) is an event-based API. A SAX parser emits a sequence of events while reading the document.

DOM parser loads whole XML document in memory because of this reason DOM parser is faster than SAX as it access whole XML document in memory. DOM represents a XML document as a tree and each element is represented as a branch. representr creates an In Memory tree representation of XML file and then parses it.For this purpose it requires more memory. It is recommended to have increased heap size to avoid out of memory error

SAX (Simple API for XML Parsing) is an event based xml parsing. It is suitable for large XML file as it parse xml file step by step. It triggers an event when it finds opening tag, attribute or element and the parsing Woks accordingly. SAX is good for parsing large XML files as it doesn't require the whole file. It is capable of reading big XML file in small parts. SAX parser requires more code than DOM parser. SAX reads the xml file from top to down and if needed, from down to up. It is used for reading only not for inserting or deleting the node. SAX is faster at run time.

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

Answer:

SAX parsers are event-based - they emit a sequence of events while reading a XML document. SAX parsers move forward through the XML content, and emit an event for each significant node. In the XML file some of the most commonly used events to get the data from XML by SAX are start document and end document events. Start element, end element and the characters event which reports when usable text is available. There are also error-handling events including warning error and fatal error. These events are handled by the super class. A fatal error will stop the processing in its tracks. If we want some custom handling of the errors, we have to override the methods for these events. Other events that are available include notations processing instructions, ignore about white space and entities. For the given XML file the SAX will parse it like flowing:

startElement ?xml version 1.0

startElement Note id=1.0

startElement to

cherecter John

endElement to

startElement from

cherecter Jenny

endElement from

startElement heading

endElement heading

startElement body

cherecter Don't

startElement b

cherecter forget

endElement b

cherecter our meeting!

endElement body

endElement note

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.
We recommend to use Notepad++ as the editor. However, you can use any other editor to write the XML description. As mentioned above, only the DTD is required. Please provide your solution as text contained in the PDF of your solutions, no .dtd file is required. (6 points)

Answer:

<!ELEMENT STUDENT (PERSONAL_INFORMATION,EDUCATIONAL_INFORMATION,REGISTERED_INFORMATION)+>

<!ELEMENT PERSONAL_INFORMATION (name,dateOfBirth+,PhoneNumber+,PostCode,City)>

<!ELEMENT name (firstname,familyName)>

<!ELEMENT firstname(#PCDATA)>

<!ELEMENT familyName(#PCDATA)>

<!ELEMENT dateOfBirth(day,month,year)>

<!ELEMENT day(#PCDATA)>

<!ELEMENT month(#PCDATA)>

<!ELEMENT year(#PCDATA)>

<!ELEMENT PhoneNumber(#PCDATA)>

<!ELEMENT postcode(#PCDATA)>

<!ELEMENT city(#PCDATA)>

<!ELEMENT EDUCATIONAL_INFORMATION(Semester+,fieldOfStudy?)>

<!ELEMENT Semester(#PCDATA)>

<!ELEMENT fieldOfStudy(#PCDATA)>

<!ELEMENT REGISTERED_INFORMATION(courseName,time,room)>

<!ELEMENT courseName(#PCDATA)>

<!ELEMENT time(#PCDATA)>

<!ELEMENT room(#PCDATA)>