XML and JAXP

PB138

Intuitive introduction into XML

- XML is text
- Very often XML document is text file or string
- First line is called Prolog

```
<?xml version=”1.0” encoding=”UTF-8” ?>
<text>This is an XML document</text>
```

Element

start tag: <person>

end tag: </person>

- May contain text and/or other elements

Create Simple Well Formed XML

- Create an XML document (in a text file) that contains 3 people. Each person has a name and surname.
- Try xmllint to validate that your document is well formed

XML Terminology

- well formed
  - It contains prolog (heading) and exactly one root element. Before and after the root element, there can be processing instructions, comments (Misc).
  - It meets all the well-formedness constraints given in the specification.
- Parent vs Ancestor, Child vs Descendant

XML 1.0

- www.w3.org/TR/2008/REC-xml-20081126/
- http://www.xml.com/axml/axml.html

This is not well-formed prolog:
<?xml encoding='UTF-8' version='1.0'?>

Xml Attributes

- Attributes add information to elements
- One element cannot have 2 attributes with same name (provided the namespaces of the attributes are the same)

```
...
<person age="27">
...
```

XML Entities

- Variables/shortcuts

  < when you need to write **<**

  > for **>**

  & for **&**

- `<![CDATA[SOME escaped TEXT]]>`

Modify your document

- Add attribute age to your people
- Add “html name” element and using entities put this into its content:
- Filip

Java API for XML parsing/writing

- Acronym JAXP
- http://docs.oracle.com/javase/8/docs/api/
  - org.w3c - important objects such as Document
  - javax.xml.parsers - for parsing Document from file

JAXP Documentation

- Find the methods for disabling validation and namespace awarness in javadoc for DocumentBuilderFactory
- Turn the features off in JaxpParser.java

Print Root Element Name

- Modify XmlIntroduction main class
- Use JaxpParser.getDocument to get Document
- use Document to get the Document Element (root Element) and print name the element (tag name). The output should be “people”

Print names of people in people.xml

- use Document.getElementsByTagName to get *people* elements
- Iterate through NodeList
  - int NodeList.getLength()
  - Node NodeList.item(int i)
- Cast each Node to Element and use getAttribute to print out name of the person!

XML modifications

- in XmlIntroduction uncomment usage of saveToFile method
- try to run the XmlIntroduction and inspect people-new.xml

Method to get Elements

- Create method in XmlIntroduction that will return List<Element> of elements of a specific name
  - List<Element> getElements(Document doc,String name);
  - use getElementsByTagName

Method to rename people

- Create method in XmlIntroduction that will rename a person with some id: rename (Document doc, int id, String newName);
  - Integer.parseInt
  - Element.setAttribute

Remove element

- Create method that will delete a person with some id: delete(Document doc, int id);
  - **x**.getParentNode().removeChild(**x**)

Add element

- Create method that will add a new person with some id: create(Document doc, int id, String name);
  - Make sure the IDs in the document are unique!
  - use Document.createElement
  - use Element.appendChild