Lab 3 XML Processing

- Task 1: XML data processing with SAX
- Task 2: XML data processing with DOM

Some of the slides are adapted from:

- https://www.tutorialspoint.com/java_xml/java_sax_parser.htm
- https://www.tutorialspoint.com/java_xml/java_dom_parser.htm

Application Program Interface

- There are two standard application programming interfaces to XML data:
  - SAX (Simple API for XML)
    - Based on a parser model, user provides event handlers for parsing events
      - e.g. start of element, end of element
  - DOM (Document Object Model)
    - XML data is parsed into a tree representation
    - Variety of functions provided for traversing the DOM tree
    - e.g. Java DOM API provides Node class with methods
      getParentNode( ), getFirstChild( ), getNextSibling( )
      getAttribute( ), getData( ) (for text node)
      getElementsByTagName( ), …
    - Also provides functions for updating DOM tree

SAX

- SAX is an event-based parser for XML document processing.
- Applications using SAX receive event notifications about the XML document being processed (e.g. an element or attribute), at a time, in sequential order, starting at the top of the document, and ending with the closing of the ROOT element.

XML Processing with SAX

- Reads an XML document from top to bottom.
- Tokens are processed in the same order that they appear in the document.
- Reports to the application program the nature of tokens that the parser has encountered as they occur.
- The application program provides an "event" handler that must be registered with the parser.
- As the tokens are identified, callback methods in the handler are invoked.

When to use SAX?

- Process the XML document in a linear fashion top-down.
- The document is not deeply nested.
- Process a very large XML document whose DOM tree would consume too much memory.
- The problem to be solved involves only a part of the XML document.
- Data is available as soon as it is seen by the parser, so SAX works well for an XML document that arrives over a stream.

Limitations of SAX

- No random access to an XML document since it is processed in a forward-only manner.
- If one needs to keep track of data that the parser has seen or to change the order of items, s/he must write the code and store the data on her/his own.

Important Classes and Interfaces in SAX

- *javax.xml.parsers.SAXParser;*
- *javax.xml.parsers.SAXParserFactory;*
- *org.xml.sax.Attributes;*
- *org.xml.sax.SAXException;*
- *org.xml.sax.helpers.DefaultHandler;*
- *...*

Task 1: XML processing with SAX

- Download the following two files
  - SAXParserDemo.java
  - XMLDoc.xml
  - *Remove the package declaration if package is not used in your code*
  - Understand the code in SAXParserDemo.java, especially the handler implementation.
  - Be careful of the package and the relative path of the XML file in the Java file.
- Run SAXParserDemo and record the output.

DOM

- Document Object Model (DOM) is an official recommendation of the World Wide Web Consortium (W3C).
- It defines an interface that enables programs to access and update the style, structure, and contents of XML documents.
- When one parses an XML document with a DOM parser, s/he gets a tree structure containing all elements of the document. DOM provides a variety of functions to examine the contents and structure of a document.

When to use DOM?

- Need to know a lot about the structure of a document.
- Need to move parts of an XML document around (e.g. might want to sort certain elements).
- Need to use the information in an XML document more than once.

Important Classes and Interfaces in DOM

- *javax.xml.parsers.DocumentBuilderFactory;*
- *javax.xml.parsers.DocumentBuilder;*
- *org.w3c.dom.Document;*
- *org.w3c.dom.NodeList;*
- *org.w3c.dom.Node;*
- *org.w3c.dom.Element;*
- ...

Common Methods in DOM

- Document.getDocumentElement() – Returns the root element of the document.
- Node.getFirstChild() – Returns the first child of a given Node.
- Node.getLastChild() – Returns the last child of a given Node.
- Node.getNextSibling() – Returns the next sibling of a given Node.
- Node.getPreviousSibling() – Returns the previous sibling of a given Node.
- Node.getAttribute(attrName) – For a given Node, it returns the attribute with the requested name.
- *For more refer to DOM Java API documentation.*

Task 2: XML processing with DOM

- Download the following two files
  - DOMParserDemo.java
  - XMLDoc.xml
  - *Remove the package declaration if package is not used in your code*
  - Understand the code in DOMParserDemo.java.
  - Be careful of the package and the relative path of the XML file in the Java file.
- Run DOMParserDemo and record the output.