CMPSC380 Laboratory Assignment 8
Using the Document Object Model to Parse XML Files

Andreas Bach Landgrebe[1] and Alex Means[1]

[1]Allegheny College, Department of Computer Science

Wednesday, November 19th, 2014

1. A two paragraph commentary on the work that each team member completed.

   Commentary of Andreas

   I felt the process to complete this laboratory assignment went successful. Alex and I were able to divide up the work effectively to be able to complete this assignment at the correct time. Alex was responsible for providing the required deliverables 4 and 5 which were the output from running the ReadXMLFileWithDOM and DOMEcho program with both of the provided XML files. He was also responsible for providing the deliverable 7 which was snippets of the five XML files and the output from running the enhanced DOMEcho program.

   I was able to complete the agreed deliverables to complete. I was responsible for providing a description of DOM-based XML parsing, examining its features, strengths, and weaknesses. Both Alex and I agreed to work together to provided required deliverable 3 which was the final version of the staff.xml XML file that you customized and extended as appropriate. I was able to complete deliverable 6 to be able to implement an enhanced version of the DOMEcho program that produced less debugging output. Both Both and I discussed with each other from deliverable 8 to explain the performance trends associated with DOM-based XML parsing. I felt that each partner did their fair share of work to complete this assignment at the correct time. The partnership and communication between the partners was outstanding to be able to complete this laboratory assignment effectively.

Commentary of Alex

This lab seemed to progress fairly well overall, and I feel both partners did well in operating as a team and smoothly completing this assignment. We essentially just divided up the jobs as we saw fit and got to work. Andreas wrote out the DOM description. We both gave our own customizations to staff.xml, and I recorded the output from the original XML files.

We initially worked together on the main force of the lab, extending DOMEcho. I saw the basic idea of adjusting the command line argument, but Andreas was ultimately the one to actually implement the solution. Once the program was complete, I took to Google in order to find more example XML files with which to test the program. Using his code updates, I was able to test XML files of varying complexity with no difficulties. I then compiled the report of DOM-based XML performance trade-offs by utilizing our program's results. Being the one to have prior knowledge of Latex, Andreas finished the lab by bringing our deliverables together into a single lab report. Overall, we both accomplished a fair portion of the work and, more to the point, both came out of the lab understanding all concepts fully.

2. A description of DOM-based XML parsing, examining its features, strengths, and weaknesses.
   Features
   (a) It a hierarchy-based parser that creates an object model of the entire XML document.
   (b) XML will be broken down into three main pieces: Elements (sometimes called tags), Attributes, and the data that the elements and attributes describe.

   Strengths
   (a) It is considered to be the most popular way to manipulate XML files.
   (b) DOM is able to support random-access manipulation. One will be able to see the tokens more than once in document order.
   (c) DOM XML parsing is able to retain a complete model which is able to prevent reduced memory overhead.
   (d) The DOM approach discards more debugging output than an approach like SAX (Simple API for XML).
   (e) The DOM parser is not as challenging to implement compared to the SAX parser
   (f) The DOM parser is able to provide a document representation to be able to manipulate, serialize and traverse XML documents.

   Weaknesses
   (a) An approach like SAX presents a document as a serialized "event stream" which DOM does not do.
   (b) The SAX-based XML parsing is able to scan and parse gigabytes worth of XML document without hitting resource limits since it does not try to create a DOM representation in memory.
   (c) Generally, the SAX-based XML parsing is faster and requires fewer resources.

3. The final version of the staff.xml XML file that you customized and extended as appropriate.

```xml
<?xml version="1.0"?>
<company>
    <staff id="1001">
        <firstname>gregory</firstname>
        <lastname>kapfhammer</lastname>
        <nickname>gkapfham</nickname>
        <salary>200000</salary>
        <department>CS
            <location state="PA">Meadville</location>
        </department>
    </staff>
    <staff id="2001">
        <firstname>john</firstname>
        <lastname>wenskovitch</lastname>
        <nickname>jwenskovitch</nickname>
        <salary>100000</salary>
        <department>CS</department>
    </staff>
    <staff id="3001">
      <firstname>janyl</firstname>
      <lastname>jumadinova</lastname>
      <nickname>jjumadinova</nickname>
      <salary>100000</salary>
       <department>CS</department>
    </staff>
    <staff id="4001">
        <firstname>robert</firstname>
        <lastname>roos</lastname>
        <nickname>rroos</nickname>
        <salary>0</salary>
        <department>CS</department>
    </staff>
</company>
```

4. The output from running the ReadXMLFileWithDOM program with both of the XML files.

```
   staff.xml:
meansa@aldenv179:~/cs380f2014-share/cs380-team9-lab8\$ java
    ReadXMLFileWithDOM

Root element :company

Current Element: staff
Staff id : 1001
First Name : gregory
Last Name : kapfhammer
Nick Name : gkapfham
Salary : 200000

Current Element: staff
Staff id : 2001
First Name : john
Last Name : wenskovitch
Nick Name : jwenskovitch
Salary : 100000

Current Element: staff
Staff id : 3001
First Name : janyl
Last Name : jumadinova
Nick Name : jjumadinova
Salary : 100000

nih-mesh.xml:

meansa@aldenv179:~/cs380f2014-share/cs380-team9-lab8\$ java
    ReadXMLFileWithDOM
Root element :DescriptorRecordSet
```

5. The output from running the DOMEcho program with both of the provided XML files.

```
staff.xml:

meansa@aldenv105:~/cs380f2014-share/labs/lab8$ java DOMEcho
    staff.xml
DOC: nodeName="#document"
  ELEM: nodeName="company" local="company"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="staff" local="staff"
        ATTR: nodeName="id" local="id" nodeValue="1001"
          TEXT: nodeName="#text" nodeValue="1001"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="firstname" local="firstname"
        TEXT: nodeName="#text" nodeValue="gregory"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="lastname" local="lastname"
        TEXT: nodeName="#text" nodeValue="kapfhammer"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="nickname" local="nickname"
        TEXT: nodeName="#text" nodeValue="gkapfham"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="salary" local="salary"
        TEXT: nodeName="#text" nodeValue="200000"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="staff" local="staff"
        ATTR: nodeName="id" local="id" nodeValue="2001"
          TEXT: nodeName="#text" nodeValue="2001"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="firstname" local="firstname"
        TEXT: nodeName="#text" nodeValue="john"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="lastname" local="lastname"
        TEXT: nodeName="#text" nodeValue="wenskovitch"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="nickname" local="nickname"
        TEXT: nodeName="#text" nodeValue="jwenskovitch"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="salary" local="salary"
        TEXT: nodeName="#text" nodeValue="100000"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="staff" local="staff"
        ATTR: nodeName="id" local="id" nodeValue="3001"
         TEXT: nodeName="#text" nodeValue="3001"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="firstname" local="firstname"
        TEXT: nodeName="#text" nodeValue="janyl"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="lastname" local="lastname"
        TEXT: nodeName="#text" nodeValue="jumadinova"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="nickname" local="nickname"
        TEXT: nodeName="#text" nodeValue="jjumadinova"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="salary" local="salary"
        TEXT: nodeName="#text" nodeValue="100000"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
```

nih-mesh.xml:

```
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="QualifierUI" local="QualifierUI"
       TEXT: nodeName="#text" nodeValue="Q000633"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="QualifierName" local="QualifierName"
       TEXT: nodeName="#text" nodeValue=[WS]
       ELEM: nodeName="String" local="String"
        TEXT: nodeName="#text" nodeValue="toxicity"
       TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="Abbreviation" local="Abbreviation"
      TEXT: nodeName="#text" nodeValue="TO"
     TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="AllowableQualifier"
      local="AllowableQualifier"
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="QualifierReferredTo"
       local="QualifierReferredTo"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="QualifierUI" local="QualifierUI"
       TEXT: nodeName="#text" nodeValue="Q000652"
      TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="QualifierName" local="QualifierName"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="String" local="String"
        TEXT: nodeName="#text" nodeValue="urine"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Abbreviation" local="Abbreviation"
    TEXT: nodeName="#text" nodeValue="UR"
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="AllowableQualifier"
    local="AllowableQualifier"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="QualifierReferredTo"
      local="QualifierReferredTo"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="QualifierUI" local="QualifierUI"
      TEXT: nodeName="#text" nodeValue="Q000737"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="QualifierName" local="QualifierName"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="String" local="String"
        TEXT: nodeName="#text" nodeValue="chemistry"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Abbreviation" local="Abbreviation"
    TEXT: nodeName="#text" nodeValue="CH"
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="AllowableQualifier"
    local="AllowableQualifier"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="QualifierReferredTo"
      local="QualifierReferredTo"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="QualifierUI" local="QualifierUI"
      TEXT: nodeName="#text" nodeValue="Q000744"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="QualifierName" local="QualifierName"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="String" local="String"
        TEXT: nodeName="#text"
            nodeValue="contraindications"
```

```
          TEXT: nodeName="#text" nodeValue=[WS]
        TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Abbreviation" local="Abbreviation"
        TEXT: nodeName="#text" nodeValue="CT"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="AllowableQualifier"
        local="AllowableQualifier"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="QualifierReferredTo"
          local="QualifierReferredTo"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="QualifierUI" local="QualifierUI"
          TEXT: nodeName="#text" nodeValue="Q000819"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="QualifierName" local="QualifierName"
          TEXT: nodeName="#text" nodeValue=[WS]
          ELEM: nodeName="String" local="String"
            TEXT: nodeName="#text" nodeValue="agonists"
          TEXT: nodeName="#text" nodeValue=[WS]
        TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Abbreviation" local="Abbreviation"
        TEXT: nodeName="#text" nodeValue="AG"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="HistoryNote" local="HistoryNote"
    TEXT: nodeName="#text" nodeValue="91(75); was A 23187
        1975-90 (see under ANTIBIOTICS 1975-83)
"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="OnlineNote" local="OnlineNote"
    TEXT: nodeName="#text" nodeValue="use CALCIMYCIN to
        search A 23187 1975-90
"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="PublicMeSHNote" local="PublicMeSHNote"
    TEXT: nodeName="#text" nodeValue="91; was A 23187
        1975-90 (see under ANTIBIOTICS 1975-83)
"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="PreviousIndexingList"
      local="PreviousIndexingList"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="PreviousIndexing"
        local="PreviousIndexing"
      TEXT: nodeName="#text" nodeValue="Antibiotics
          (1973-1974)"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="PreviousIndexing"
        local="PreviousIndexing"
      TEXT: nodeName="#text" nodeValue="Carboxylic Acids
          (1973-1974)"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="PharmacologicalActionList"
      local="PharmacologicalActionList"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="PharmacologicalAction"
        local="PharmacologicalAction"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="DescriptorReferredTo"
          local="DescriptorReferredTo"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="DescriptorUI" local="DescriptorUI"
          TEXT: nodeName="#text" nodeValue="D061207"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="DescriptorName"
            local="DescriptorName"
          TEXT: nodeName="#text" nodeValue=[WS]
          ELEM: nodeName="String" local="String"
            TEXT: nodeName="#text" nodeValue="Calcium
                Ionophores"
          TEXT: nodeName="#text" nodeValue=[WS]
        TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="PharmacologicalAction"
        local="PharmacologicalAction"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="DescriptorReferredTo"
          local="DescriptorReferredTo"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="DescriptorUI" local="DescriptorUI"
          TEXT: nodeName="#text" nodeValue="D000900"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="DescriptorName"
            local="DescriptorName"
          TEXT: nodeName="#text" nodeValue=[WS]
          ELEM: nodeName="String" local="String"
            TEXT: nodeName="#text"
                nodeValue="Anti-Bacterial Agents"
          TEXT: nodeName="#text" nodeValue=[WS]
        TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="TreeNumberList" local="TreeNumberList"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="TreeNumber" local="TreeNumber"
      TEXT: nodeName="#text" nodeValue="D03.438.221.173"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="RecordOriginatorsList"
      local="RecordOriginatorsList"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="RecordOriginator"
        local="RecordOriginator"
      TEXT: nodeName="#text" nodeValue="NLM"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="RecordMaintainer"
        local="RecordMaintainer"
      TEXT: nodeName="#text" nodeValue="SYSTEM"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="RecordAuthorizer"
        local="RecordAuthorizer"
      TEXT: nodeName="#text" nodeValue="jtk"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="ConceptList" local="ConceptList"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Concept" local="Concept"
        ATTR: nodeName="PreferredConceptYN"
            local="PreferredConceptYN" nodeValue="Y"
          TEXT: nodeName="#text" nodeValue="Y"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="ConceptUI" local="ConceptUI"
        TEXT: nodeName="#text" nodeValue="M0000001"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="ConceptName" local="ConceptName"
        TEXT: nodeName="#text" nodeValue=[WS]
        ELEM: nodeName="String" local="String"
          TEXT: nodeName="#text" nodeValue="Calcimycin"
      TEXT: nodeName="#text" nodeValue=[WS]
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="CASN1Name" local="CASN1Name"
      TEXT: nodeName="#text"
        nodeValue="4-Benzoxazolecarboxylic acid,
        5-(methylamino)-2-((3,9,11-trimethyl-8-(1-methyl-2-oxo-2-(1H-pyrrol-2-yl)
        (6S-(6alpha(2S*,3S*),8beta(R*),9beta,11alpha))-"
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="RegistryNumber" local="RegistryNumber"
      TEXT: nodeName="#text" nodeValue="37H9VM9WZL"
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="ScopeNote" local="ScopeNote"
      TEXT: nodeName="#text" nodeValue="An ionophorous,
        polyether antibiotic from Streptomyces
        chartreusensis. It binds and transports CALCIUM
        and other divalent cations across membranes and
        uncouples oxidative phosphorylation while
        inhibiting ATPase of rat liver mitochondria.
        The substance is used mostly as a biochemical
        tool to study the role of divalent cations in
        various biological systems.
  "
     TEXT: nodeName="#text" nodeValue=[WS]
     ELEM: nodeName="SemanticTypeList"
       local="SemanticTypeList"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="SemanticType" local="SemanticType"
       TEXT: nodeName="#text" nodeValue=[WS]
       ELEM: nodeName="SemanticTypeUI"
         local="SemanticTypeUI"
        TEXT: nodeName="#text" nodeValue="T109"
       TEXT: nodeName="#text" nodeValue=[WS]
       ELEM: nodeName="SemanticTypeName"
         local="SemanticTypeName"
        TEXT: nodeName="#text" nodeValue="Organic
          Chemical"
       TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="SemanticType" local="SemanticType"
       TEXT: nodeName="#text" nodeValue=[WS]
       ELEM: nodeName="SemanticTypeUI"
         local="SemanticTypeUI"
        TEXT: nodeName="#text" nodeValue="T195"
       TEXT: nodeName="#text" nodeValue=[WS]
       ELEM: nodeName="SemanticTypeName"
```

```
        local="SemanticTypeName"
      TEXT: nodeName="#text" nodeValue="Antibiotic"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="RelatedRegistryNumberList"
    local="RelatedRegistryNumberList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="RelatedRegistryNumber"
      local="RelatedRegistryNumber"
    TEXT: nodeName="#text" nodeValue="52665-69-7
        (Calcimycin)"
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="ConceptRelationList"
    local="ConceptRelationList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="ConceptRelation"
      local="ConceptRelation"
      ATTR: nodeName="RelationName"
          local="RelationName" nodeValue="NRW"
        TEXT: nodeName="#text" nodeValue="NRW"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Concept1UI" local="Concept1UI"
      TEXT: nodeName="#text" nodeValue="M0000001"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Concept2UI" local="Concept2UI"
      TEXT: nodeName="#text" nodeValue="M0353609"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="TermList" local="TermList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Term" local="Term"
      ATTR: nodeName="ConceptPreferredTermYN"
          local="ConceptPreferredTermYN" nodeValue="Y"
        TEXT: nodeName="#text" nodeValue="Y"
      ATTR: nodeName="IsPermutedTermYN"
          local="IsPermutedTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="LexicalTag" local="LexicalTag"
          nodeValue="NON"
        TEXT: nodeName="#text" nodeValue="NON"
      ATTR: nodeName="PrintFlagYN"
          local="PrintFlagYN" nodeValue="Y"
```

```
     TEXT: nodeName="#text" nodeValue="Y"
    ATTR: nodeName="RecordPreferredTermYN"
      local="RecordPreferredTermYN" nodeValue="Y"
     TEXT: nodeName="#text" nodeValue="Y"
   TEXT: nodeName="#text" nodeValue=[WS]
   ELEM: nodeName="TermUI" local="TermUI"
    TEXT: nodeName="#text" nodeValue="T000002"
   TEXT: nodeName="#text" nodeValue=[WS]
   ELEM: nodeName="String" local="String"
    TEXT: nodeName="#text" nodeValue="Calcimycin"
   TEXT: nodeName="#text" nodeValue=[WS]
   ELEM: nodeName="DateCreated" local="DateCreated"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Year" local="Year"
     TEXT: nodeName="#text" nodeValue="1999"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Month" local="Month"
     TEXT: nodeName="#text" nodeValue="01"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Day" local="Day"
     TEXT: nodeName="#text" nodeValue="01"
    TEXT: nodeName="#text" nodeValue=[WS]
   TEXT: nodeName="#text" nodeValue=[WS]
   ELEM: nodeName="ThesaurusIDlist"
     local="ThesaurusIDlist"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="ThesaurusID" local="ThesaurusID"
     TEXT: nodeName="#text" nodeValue="FDA SRS
       (2014)"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="ThesaurusID" local="ThesaurusID"
     TEXT: nodeName="#text" nodeValue="NLM (1975)"
    TEXT: nodeName="#text" nodeValue=[WS]
   TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
 TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="Concept" local="Concept"
  ATTR: nodeName="PreferredConceptYN"
    local="PreferredConceptYN" nodeValue="N"
   TEXT: nodeName="#text" nodeValue="N"
 TEXT: nodeName="#text" nodeValue=[WS]
 ELEM: nodeName="ConceptUI" local="ConceptUI"
  TEXT: nodeName="#text" nodeValue="M0353609"
 TEXT: nodeName="#text" nodeValue=[WS]
```

```
ELEM: nodeName="ConceptName" local="ConceptName"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="String" local="String"
    TEXT: nodeName="#text" nodeValue="A-23187"
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="RegistryNumber" local="RegistryNumber"
  TEXT: nodeName="#text" nodeValue="0"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="SemanticTypeList"
    local="SemanticTypeList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="SemanticType" local="SemanticType"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="SemanticTypeUI"
        local="SemanticTypeUI"
      TEXT: nodeName="#text" nodeValue="T109"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="SemanticTypeName"
        local="SemanticTypeName"
      TEXT: nodeName="#text" nodeValue="Organic
          Chemical"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="SemanticType" local="SemanticType"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="SemanticTypeUI"
        local="SemanticTypeUI"
      TEXT: nodeName="#text" nodeValue="T195"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="SemanticTypeName"
        local="SemanticTypeName"
      TEXT: nodeName="#text" nodeValue="Antibiotic"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="ConceptRelationList"
    local="ConceptRelationList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="ConceptRelation"
      local="ConceptRelation"
      ATTR: nodeName="RelationName"
          local="RelationName" nodeValue="NRW"
        TEXT: nodeName="#text" nodeValue="NRW"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Concept1UI" local="Concept1UI"
      TEXT: nodeName="#text" nodeValue="M0000001"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Concept2UI" local="Concept2UI"
      TEXT: nodeName="#text" nodeValue="M0353609"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="TermList" local="TermList"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Term" local="Term"
      ATTR: nodeName="ConceptPreferredTermYN"
          local="ConceptPreferredTermYN" nodeValue="Y"
        TEXT: nodeName="#text" nodeValue="Y"
      ATTR: nodeName="IsPermutedTermYN"
          local="IsPermutedTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="LexicalTag" local="LexicalTag"
          nodeValue="LAB"
        TEXT: nodeName="#text" nodeValue="LAB"
      ATTR: nodeName="PrintFlagYN"
          local="PrintFlagYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="RecordPreferredTermYN"
          local="RecordPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="TermUI" local="TermUI"
      TEXT: nodeName="#text" nodeValue="T000001"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="String" local="String"
      TEXT: nodeName="#text" nodeValue="A-23187"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="DateCreated" local="DateCreated"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Year" local="Year"
        TEXT: nodeName="#text" nodeValue="1990"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Month" local="Month"
        TEXT: nodeName="#text" nodeValue="03"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Day" local="Day"
        TEXT: nodeName="#text" nodeValue="08"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="ThesaurusIDlist"
        local="ThesaurusIDlist"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="ThesaurusID" local="ThesaurusID"
        TEXT: nodeName="#text" nodeValue="NLM (1991)"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Term" local="Term"
      ATTR: nodeName="ConceptPreferredTermYN"
          local="ConceptPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="IsPermutedTermYN"
          local="IsPermutedTermYN" nodeValue="Y"
        TEXT: nodeName="#text" nodeValue="Y"
      ATTR: nodeName="LexicalTag" local="LexicalTag"
          nodeValue="LAB"
        TEXT: nodeName="#text" nodeValue="LAB"
      ATTR: nodeName="PrintFlagYN"
          local="PrintFlagYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="RecordPreferredTermYN"
          local="RecordPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="TermUI" local="TermUI"
      TEXT: nodeName="#text" nodeValue="T000001"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="String" local="String"
      TEXT: nodeName="#text" nodeValue="A 23187"
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Term" local="Term"
      ATTR: nodeName="ConceptPreferredTermYN"
          local="ConceptPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="IsPermutedTermYN"
          local="IsPermutedTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="LexicalTag" local="LexicalTag"
          nodeValue="NON"
        TEXT: nodeName="#text" nodeValue="NON"
      ATTR: nodeName="PrintFlagYN"
          local="PrintFlagYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="RecordPreferredTermYN"
          local="RecordPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="TermUI" local="TermUI"
      TEXT: nodeName="#text" nodeValue="T000003"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="String" local="String"
      TEXT: nodeName="#text" nodeValue="Antibiotic
          A23187"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="DateCreated" local="DateCreated"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Year" local="Year"
        TEXT: nodeName="#text" nodeValue="1990"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Month" local="Month"
        TEXT: nodeName="#text" nodeValue="03"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="Day" local="Day"
        TEXT: nodeName="#text" nodeValue="08"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="ThesaurusIDlist"
        local="ThesaurusIDlist"
      TEXT: nodeName="#text" nodeValue=[WS]
      ELEM: nodeName="ThesaurusID" local="ThesaurusID"
        TEXT: nodeName="#text" nodeValue="NLM (1991)"
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="Term" local="Term"
      ATTR: nodeName="ConceptPreferredTermYN"
          local="ConceptPreferredTermYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="IsPermutedTermYN"
          local="IsPermutedTermYN" nodeValue="Y"
        TEXT: nodeName="#text" nodeValue="Y"
      ATTR: nodeName="LexicalTag" local="LexicalTag"
          nodeValue="NON"
        TEXT: nodeName="#text" nodeValue="NON"
      ATTR: nodeName="PrintFlagYN"
          local="PrintFlagYN" nodeValue="N"
        TEXT: nodeName="#text" nodeValue="N"
      ATTR: nodeName="RecordPreferredTermYN"
```

```
        local="RecordPreferredTermYN" nodeValue="N"
      TEXT: nodeName="#text" nodeValue="N"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="TermUI" local="TermUI"
    TEXT: nodeName="#text" nodeValue="T000003"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="String" local="String"
    TEXT: nodeName="#text" nodeValue="A23187,
        Antibiotic"
  TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="Term" local="Term"
    ATTR: nodeName="ConceptPreferredTermYN"
        local="ConceptPreferredTermYN" nodeValue="N"
      TEXT: nodeName="#text" nodeValue="N"
    ATTR: nodeName="IsPermutedTermYN"
        local="IsPermutedTermYN" nodeValue="N"
      TEXT: nodeName="#text" nodeValue="N"
    ATTR: nodeName="LexicalTag" local="LexicalTag"
        nodeValue="LAB"
      TEXT: nodeName="#text" nodeValue="LAB"
    ATTR: nodeName="PrintFlagYN"
        local="PrintFlagYN" nodeValue="N"
      TEXT: nodeName="#text" nodeValue="N"
    ATTR: nodeName="RecordPreferredTermYN"
        local="RecordPreferredTermYN" nodeValue="N"
      TEXT: nodeName="#text" nodeValue="N"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="TermUI" local="TermUI"
    TEXT: nodeName="#text" nodeValue="T000004"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="String" local="String"
    TEXT: nodeName="#text" nodeValue="A23187"
  TEXT: nodeName="#text" nodeValue=[WS]
  ELEM: nodeName="DateCreated" local="DateCreated"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Year" local="Year"
      TEXT: nodeName="#text" nodeValue="1974"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Month" local="Month"
      TEXT: nodeName="#text" nodeValue="11"
    TEXT: nodeName="#text" nodeValue=[WS]
    ELEM: nodeName="Day" local="Day"
      TEXT: nodeName="#text" nodeValue="11"
    TEXT: nodeName="#text" nodeValue=[WS]
```

```
            TEXT: nodeName="#text" nodeValue=[WS]
            ELEM: nodeName="ThesaurusIDlist"
                local="ThesaurusIDlist"
              TEXT: nodeName="#text" nodeValue=[WS]
              ELEM: nodeName="ThesaurusID" local="ThesaurusID"
                TEXT: nodeName="#text" nodeValue="UNK (19XX)"
              TEXT: nodeName="#text" nodeValue=[WS]
            TEXT: nodeName="#text" nodeValue=[WS]
          TEXT: nodeName="#text" nodeValue=[WS]
        TEXT: nodeName="#text" nodeValue=[WS]
      TEXT: nodeName="#text" nodeValue=[WS]
    TEXT: nodeName="#text" nodeValue=[WS]
  TEXT: nodeName="#text" nodeValue=[WS]
```

6. The source code of the DOMEcho program that you enhanced to produce less debugging output.

```
// JAXP packages
import javax.xml.parsers.*;
import org.xml.sax.*;
import org.xml.sax.helpers.*;
import org.w3c.dom.*;

import java.io.*;



/**
 * This is a program to echo a DOM tree using DOM Level 2
   interfaces. Use
 * JAXP to load an XML file and create a DOM tree. DOM
   currently does not
 * provide a method to do this. (This is planned for Level 3.)
   See the
 * method "main" for the three basic steps. Once the
   application obtains a
 * DOM Document tree, it dumps out the nodes in the tree and
   associated
 * node attributes for each node.
 *
 * This program also shows how to validate a document along
   with using an
 * ErrorHandler to capture validation errors.
 *
 * Note: Program flags may be used to create non-conformant
    but possibly
 * useful DOM trees. In some cases, particularly with element
    content
 * whitespace, applications may not want to rely on JAXP to
    filter out
 * these nodes but may want to skip the nodes themselves so
    the application
 * will be more robust.
 *
 * Update 2002-04-18: Added code that shows how to use JAXP
    1.2 features to
 * support W3C XML Schema validation. See the JAXP 1.2
    maintenance review
 * specification for more information on these features.
 *
 * @author Edwin Goei
 */
public class DOMEcho {
    /** All output will use this encoding */
    static final String outputEncoding = "UTF-8";

    /** Output goes here */
    private PrintWriter out;

    /** Indent level */
    private int indent = 0;

    /*as a default, if the parameter -no is not called, then
        the debugging output
    will be displayed
    */
    static boolean noDebuggingOutput = false;

    /** Indentation will be in multiples of basicIndent */
    private final String basicIndent = " ";

    /** Constants used for JAXP 1.2 */
    static final String JAXP_SCHEMA_LANGUAGE =
        "http://java.sun.com/xml/jaxp/properties/schemaLanguage";
    static final String W3C_XML_SCHEMA =
        "http://www.w3.org/2001/XMLSchema";
    static final String JAXP_SCHEMA_SOURCE =
        "http://java.sun.com/xml/jaxp/properties/schemaSource";
```

```
DOMEcho(PrintWriter out) {
    this.out = out;
}

/**
 * Echo common attributes of a DOM2 Node and terminate
     output with an
 * EOL character.
 */
private void printlnCommon(Node n) {
    out.print(" nodeName=\"" + n.getNodeName() + "\"");

    String val = n.getNamespaceURI();
    if (val != null) {
        out.print(" uri=\"" + val + "\"");
    }

    val = n.getPrefix();
    if (val != null) {
        out.print(" pre=\"" + val + "\"");
    }

    val = n.getLocalName();
    if (val != null) {
        out.print(" local=\"" + val + "\"");
    }

    val = n.getNodeValue();
    if (val != null) {
        out.print(" nodeValue=");
        if (val.trim().equals("")) {
            // Whitespace
            out.print("[WS]");
        } else {
            out.print("\"" + n.getNodeValue() + "\"");
        }
    }
    out.println();
}

/**
 * Indent to the current level in multiples of basicIndent
 */
private void outputIndentation() {
    for (int i = 0; i < indent; i++) {
        out.print(basicIndent);
    }
}

/**
 * Recursive routine to print out DOM tree nodes
 */
private void echo(Node n) {
    // Indent to the current level before printing anything
    outputIndentation();

    int type = n.getNodeType();
    if(!noDebuggingOutput){
    switch (type) {
        case Node.ATTRIBUTE_NODE:
            out.print("ATTR:");
            printlnCommon(n);
            break;
        case Node.CDATA_SECTION_NODE:
            out.print("CDATA:");
            printlnCommon(n);
            break;
        case Node.COMMENT_NODE:
            out.print("COMM:");
            printlnCommon(n);
            break;
        case Node.DOCUMENT_FRAGMENT_NODE:
            out.print("DOC_FRAG:");
            printlnCommon(n);
            break;
        case Node.DOCUMENT_NODE:
            out.print("DOC:");
            printlnCommon(n);
            break;
        case Node.DOCUMENT_TYPE_NODE:
            out.print("DOC_TYPE:");
            printlnCommon(n);

            // Print entities if any
            NamedNodeMap nodeMap =
                ((DocumentType)n).getEntities();
            indent += 2;
            for (int i = 0; i < nodeMap.getLength(); i++) {
                Entity entity = (Entity)nodeMap.item(i);
                echo(entity);
            }
            indent -= 2;
            break;
        case Node.ELEMENT_NODE:
            out.print("ELEM:");
            printlnCommon(n);

            // Print attributes if any. Note: element
                attributes are not
            // children of ELEMENT_NODEs but are properties
                of their
            // associated ELEMENT_NODE. For this reason,
                they are printed
            // with 2x the indent level to indicate this.
            NamedNodeMap atts = n.getAttributes();
            indent += 2;
            for (int i = 0; i < atts.getLength(); i++) {
                Node att = atts.item(i);
                echo(att);
            }
            indent -= 2;
            break;
        case Node.ENTITY_NODE:
            out.print("ENT:");
            printlnCommon(n);
            break;
        case Node.ENTITY_REFERENCE_NODE:
            out.print("ENT_REF:");
            printlnCommon(n);
            break;
        case Node.NOTATION_NODE:
            out.print("NOTATION:");
            printlnCommon(n);
            break;
        case Node.PROCESSING_INSTRUCTION_NODE:
            out.print("PROC_INST:");
            printlnCommon(n);
            break;
        case Node.TEXT_NODE:
            out.print("TEXT:");
            printlnCommon(n);
            break;
        default:
            out.print("UNSUPPORTED NODE: " + type);
            printlnCommon(n);
```

```
                break;
        }
    }

        // Print children if any
        indent++;
        for (Node child = n.getFirstChild(); child != null;
                child = child.getNextSibling()) {
            echo(child);
                }
        indent--;
    }

    private static void usage() {
        System.err.println("Usage: DOMEcho [-options]
            <file.xml>");
        System.err.println("    -dtd = DTD validation");
        System.err.println("    -xsd | -xsdss <file.xsd> = W3C
            XML Schema validation using xsi: hints");
        System.err.println("        in instance document or
            schema source <file.xsd>");
        System.err.println("    -ws = do not create element
            content whitespace nodes");
        System.err.println("    -co[mments] = do not create
            comment nodes");
        System.err.println("    -cd[ata] = put CDATA into Text
            nodes");
        System.err.println("    -e[ntity-ref] = create
            EntityReference nodes");
        System.err.println("    -usage or -help = this
            message");
        System.exit(1);
    }

    public static void main(String[] args) throws Exception {
        String filename = null;
        boolean dtdValidate = false;
        boolean xsdValidate = false;
        String schemaSource = null;

        boolean ignoreWhitespace = false;
        boolean ignoreComments = false;
        boolean putCDATAIntoText = false;
        boolean createEntityRefs = false;
```

```
long startTime = System.nanoTime();
/*
Will be calculating how long it takes to
parse the XMl document. Start the time here.

*/

for (int i = 0; i < args.length; i++) {
    if (args[i].equals("-dtd")) {
        dtdValidate = true;
    } else if (args[i].equals("-xsd")) {
        xsdValidate = true;
    } else if (args[i].equals("-xsdss")) {
        if (i == args.length - 1) {
            usage();
        }
        xsdValidate = true;
        schemaSource = args[++i];
    } else if (args[i].equals("-ws")) {
        ignoreWhitespace = true;
    } else if (args[i].startsWith("-co")) {
        ignoreComments = true;
    } else if (args[i].startsWith("-cd")) {
        putCDATAIntoText = true;
    } else if (args[i].startsWith("-e")) {
        createEntityRefs = true;
    } else if (args[i].equals("-usage")) {
        usage();
    } else if (args[i].equals("-help")) {
        usage();
        /*
        if the command line argument -no is passed,
        then no debugging output will be displayed
        */
    } else if (args[i].startsWith("-no")) {
        noDebuggingOutput = true;
    } else {
        filename = args[i];

        // Must be last arg
        if (i != args.length - 1) {
            usage();
        }
    }
}
```

```
    long estimatedTime = System.nanoTime() - startTime;
    //stop timing how long it takes after the parsing
        is over

    double estimatedTime2 = ((System.nanoTime() -
        startTime) / 1000000000.0);
    //convert the estimatedTime to a double to
        displayed in seconds
    System.out.println("Time in nano seconds : " +
        estimatedTime);
    //display how long it took to parse in nano seconds
    System.out.printf("Time in seconds is %.9f",
        estimatedTime2);
    //display how long it took to parse in seconds
if (filename == null) {
    usage();
}

// Step 1: create a DocumentBuilderFactory and
    configure it
DocumentBuilderFactory dbf =
    DocumentBuilderFactory.newInstance();

// Set namespaceAware to true to get a DOM Level 2 tree
    with nodes
// containing namesapce information. This is necessary
    because the
// default value from JAXP 1.0 was defined to be false.
dbf.setNamespaceAware(true);

// Set the validation mode to either: no validation, DTD
// validation, or XSD validation
dbf.setValidating(dtdValidate || xsdValidate);
if (xsdValidate) {
    try {
        dbf.setAttribute(JAXP_SCHEMA_LANGUAGE,
            W3C_XML_SCHEMA);
    } catch (IllegalArgumentException x) {
        // This can happen if the parser does not
            support JAXP 1.2
        System.err.println(
                "Error: JAXP DocumentBuilderFactory
                    attribute not recognized: "
                + JAXP_SCHEMA_LANGUAGE);
            System.err.println(
                    "Check to see if parser conforms to JAXP
                        1.2 spec.");
            System.exit(1);
        }
    }

    // Set the schema source, if any. See the JAXP 1.2
        maintenance
    // update specification for more complex usages of this
        feature.
    if (schemaSource != null) {
        dbf.setAttribute(JAXP_SCHEMA_SOURCE, new
            File(schemaSource));
    }

    // Optional: set various configuration options
    dbf.setIgnoringComments(ignoreComments);
    dbf.setIgnoringElementContentWhitespace(ignoreWhitespace);
    dbf.setCoalescing(putCDATAIntoText);
    // The opposite of creating entity ref nodes is
        expanding them inline
    dbf.setExpandEntityReferences(!createEntityRefs);

    // Step 2: create a DocumentBuilder that satisfies the
        constraints
    // specified by the DocumentBuilderFactory
    DocumentBuilder db = dbf.newDocumentBuilder();

    // Set an ErrorHandler before parsing
    OutputStreamWriter errorWriter =
        new OutputStreamWriter(System.err, outputEncoding);
    db.setErrorHandler(
            new MyErrorHandler(new PrintWriter(errorWriter,
                true)));

    // Step 3: parse the input file
    Document doc = db.parse(new File(filename));

    // Print out the DOM tree
    OutputStreamWriter outWriter =
        new OutputStreamWriter(System.out, outputEncoding);
    new DOMEcho(new PrintWriter(outWriter, true)).echo(doc);
}
```

```
// Error handler to report errors and warnings
private static class MyErrorHandler implements
    ErrorHandler {
    /** Error handler output goes here */
    private PrintWriter out;

    MyErrorHandler(PrintWriter out) {
        this.out = out;
    }

    /**
     * Returns a string describing parse exception details
     */
    private String getParseExceptionInfo(SAXParseException
        spe) {
        String systemId = spe.getSystemId();
        if (systemId == null) {
            systemId = "null";
        }
        String info = "URI=" + systemId +
            " Line=" + spe.getLineNumber() +
            ": " + spe.getMessage();
        return info;
    }

    // The following methods are standard SAX ErrorHandler
        methods.
    // See SAX documentation for more info.

    public void warning(SAXParseException spe) throws
        SAXException {
        out.println("Warning: " +
            getParseExceptionInfo(spe));
    }

    public void error(SAXParseException spe) throws
        SAXException {
        String message = "Error: " +
            getParseExceptionInfo(spe);
        throw new SAXException(message);
    }

    public void fatalError(SAXParseException spe) throws
        SAXException {
        String message = "Fatal Error: " +
                getParseExceptionInfo(spe);
            throw new SAXException(message);
        }
    }
}
```

7. Snippets of the five XML files and the output from running the enhanced DOMEcho program
Source Code

```
staff.xml sample:
<?xml version="1.0"?>
<company>
    <staff id="1001">
        <firstname>gregory</firstname>
        <lastname>kapfhammer</lastname>
        <nickname>gkapfham</nickname>
        <salary>200000</salary>
        <department>CS
            <location state="PA">Meadville</location>
        </department>
    </staff>


nih-mesh.xml sample:
<?xml version="1.0"?>
<DescriptorRecordSet LanguageCode = "eng">
<DescriptorRecord DescriptorClass = "1">
  <DescriptorUI>D000001</DescriptorUI>
  <DescriptorName>
   <String>Calcimycin</String>
  </DescriptorName>
  <DateCreated>
   <Year>1974</Year>
   <Month>11</Month>
   <Day>19</Day>
  </DateCreated>
  <DateRevised>
   <Year>2013</Year>
   <Month>07</Month>
   <Day>08</Day>
  </DateRevised>
```

```
note.xml sample:
<!-- Edited by XMLSpy -->
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>


books.xml sample:
<?xml version="1.0"?>
<catalog>
   <book id="bk101">
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
      <price>44.95</price>
      <publish_date>2000-10-01</publish_date>
      <description>An in-depth look at creating applications
      with XML.</description>
   </book>
   <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <price>5.95</price>
      <publish_date>2000-12-16</publish_date>
      <description>A former architect battles corporate
          zombies,
      an evil sorceress, and her own childhood to become queen
      of the world.</description>
   </book>


Am.-Footaball-NFL.xml sample:
<?xml version="1.0" encoding="UTF-8"?><spocosy version="1.0"
    responsetime="2013-05-17 04:15:12"
    exec="0.481"><query-response requestid=""
    service="objectquery"><sport name="Am. Football"
    enetSportCode="fo" del="no" n="0" ut="2012-12-11 08:31:48"
    id="24"><tournament_template name="USA 1" sportFK="24"
    gender="male" enetID="0" del="no" n="0" ut="2012-07-04
    13:46:50" id="415"><tournament name="2012/2013"
    tournament_templateFK="415" enetSeasonID="0" del="no"
    locked="none" n="0" ut="2012-05-10 08:48:13"
    id="6804"><tournament_stage name="NFL" tournamentFK="6804"
    countryFK="16" gender="male" enetID="0"
    startdate="2012-09-06" enddate="2012-12-31" del="no"
    locked="none" n="22" ut="2012-12-24 06:45:34"
    id="827295"><event name="Minnesota Vikings-Green Bay
    Packers" tournament_stageFK="827295" startdate="2012-12-30
    22:25:00" eventstatusFK="0" status_type="finished"
    status_descFK="6" enetID="0" enetSportID="" del="no"
    locked="none" n="8" ut="2012-12-31 02:08:03"
    id="1202478"><properties><property object="event"
    objectFK="1202478" type="metadata" name="Round" value="1"
    del="no" n="0" ut="2012-05-10 09:30:11"
    id="14609443"></property>
```

Output

```
meansa@aldenv105:~/cs380-team9-lab8/Src$ java DOMEjava DOMEcho
    -no staff.xml
Time in nano seconds : 23330
Time in seconds is 0.000023518

meansa@aldenv105:~/cs380-team9-lab8/Src$ java DOMEcho -no
    nih-mesh.xml
Time in nano seconds : 32188
Time in seconds is 0.000032444

meansa@aldenv105:~/cs380-team9-lab8/Src$ java DOMEcho -no
    books.xml
Time in nano seconds : 23335
Time in seconds is 0.000023515

meansa@aldenv105:~/cs380-team9-lab8/Src$ java DOMEcho -no
    note.xml
Time in nano seconds : 24849
Time in seconds is 0.000036792

meansa@aldenv105:~/cs380-team9-lab8/Src$ java DOMEcho -no
    Am.-Football-NFL.xml
Time in nano seconds : 23992
Time in seconds is 0.000024210
```

8. A report explaining the performance trends associated with DOM-based XML parsing.

   Overall, DOM-based XML parsing seems to have the potential to be a very fast and efficient method of parsing XML. Our performance tests of five XML files (two provided by the lab, the other three sample programs found on Google) using DOMEcho.java revealed that each parsing takes an exceptionally short amount of time, with the max clocking in at just 32,188 nanoseconds. It is worth noting, however, that we worked with fairly small XML files throughout the lab. The amount of time it takes to parse files seems to increase linearly with the complexity of the file, as could be reasonably expected.nih-mesh was our largest xml file, so naturally it took the longest. Likewise, staff and books were fairly short and straightforward, so they took less time to parse.

   Naturally, a program is not always parsed in the same amount of time twice. Running DOMEcho several times for the same file showcases a gap as large as 800 nanoseconds- which, while not exactly large by normal standards, is nonetheless an interesting detail. It shows that while DOMEcho can reasonably be expected to perform well consistently, the time overhead of utilizing DOMEcho is somewhat variable.

   Given the trend of larger files taking longer, it can be assumed that larger files than the ones tested would cause DOMEcho to take longer to parse. With exceptionally large files, it could even begin to take long enough that we could reasonably start measuring in seconds rather than nanoseconds. However, even if we reach that point, DOMEcho remains quite the efficient method of parsing. The fact that we even have to speculate about the possibility of a program taking over a second to run speaks volumes towards its efficiency.

   In contrast to DOMEcho's natural efficiency is ReadXMLFileWith-DOM, which seems to perform much worse than DOMEcho across the board. Just look at the time it took to parse staff.xml: 39427991 nanoseconds for ReadXML versus just 23330 for DOMEcho. Of course, ReadXML has the extra work of actually printing out the various elements of the XML program, but even factoring this in, it seems to take much longer. In terms of time overhead, DOMEcho certainly wins the race without much difficulty. There's also the fact that ReadXML isn't as sophisticated as DOMEcho, as it cannot seem to handle files as complex as nih-mesh at all. Overall, DOMEcho is clearly the superior option, thanks to its speed and usability.

DOMEcho provides a strong case for the usability of DOM-based XML parsing. It appears to be comprehensive, able to parse whatever XML files it's given. In addition, it parses with consistently low time-overheard. Basically, DOM-based XML parsing, when implemented well, seems to be an extremely strong approach to parsing an XML document.