XML Parser Generator

Group 18

Bryan Redick
Guang Chen

Document Definition Language:

Extention of Group 5's Document Definition Language

Language Elements:

- TYPE
- ATTRIBUTE
- ELEMENT
- UNION
- CONTAINS

Features:

- Type checking of attribute values
  - STRING
  - INTEGER
- Union capable
- Allows comments in document defintion and document
- Default values allow for optional attributes
- Four different multiplicity types for elements:
  - ONE 1 time
  - MANY 1 - N times
  - ANY 0 - N times
  - AT_MOST_ONCE 0 - 1 times

Example:

```
<!-- email.xdtd -->
<!-- simple email document type -->
<TYPE name="email">
  <ATTRIBUTE name="protocol" type=string>
  <ELEMENT name="header" mult=one>
    <ATTRIBUTE name="from" type=string>
    <ATTRIBUTE name="subject" type=string
default="(none)">
    <CONTAINS>to</CONTAINS>
  </ELEMENT>

  <ELEMENT name="to" mult=many>
    <ATTRIBUTE name="receiver" type=string>
  </ELEMENT>

  <ELEMENT name="body" mult=one>
    <CONTAINS>#STRING</CONTAINS>
    <CONTAINS>attachment</CONTAINS>
  </ELEMENT>

  <ELEMENT name="attachment" mult=any>
    <ATTRIBUTE name="fileName" type=string>
    <ATTRIBUTE name="size" type=integer>
  </ELEMENT>

  <UNION name="authentication">
    <CONTAINS>checksum</CONTAINS>
    <CONTAINS>rsa-signature</CONTAINS>
  </UNION>

  <ELEMENT name="checksum" mult="one">
    <ATTRIBUTE name="value" type=integer>
  </ELEMENT>

  <ELEMENT name="rsa-signature" mult="one">
    <ATTRIBUTE name="value" type=integer>
  </ELEMENT>

  <CONTAINS>header</CONTAINS>
  <CONTAINS>body</CONTAINS>
</TYPE>
```

Parser Generator Design:

Four Classes:

- **XMLParserGenerator**
  Parses the document type definition file.
  Contains list of XMLElement objects.
  Performs checks to make sure all contained data has a definition
  Generates the parser code.

- **XMLElement**
  Contains data about an element.
  Contains a list of XMLAttribute objects.
  Generates code for parsing the element.

- **XMLAttribute**
  Contains data about an attribute
  Generates code for parsing the attribute
  Generates type checking code for the attribute value

- **XMLUnion**
  Derives from XMLElement.
  Contains data for the union.
  Generates code for parsing the union.

Error Handling:

- Parse errors handled in generator and generated parser by throwing an XMLParseException.
- Parser Generator
  - Unrecognized token
  - Unexpected token
  - Union without data
  - Unrecognized attribute type
  - Unrecognized element multiplicity
  - No element definition
- Generated Parser
  - Unrecognized token
  - Unexpected token
  - Type checking on attributes
  - Required attribute missing
  - Element not handled by union

Interface:

To generate a parser from a XML definition:
java generate <XML_Definition_File> <Output_Parser_File>

Example:
java generate email.xdtd EmailParser.java

The generated parser source file is an object. It provides you a function that takes a XML document name as input argument. It will parse the input document into a tree and return that tree.

Example:

```
EmailParser parser = new EmailParser();
Tree myTree;

myTree = parser.Parse("email.xml");
```

Once the document is parsed in the tree, the tree class provides you three functions to print out the tree, calculate an expression, and count nodes.

Example:

```
myTree.preorderPrint();
myTree.caculate();
myTree.countNodes(0);
```

We provide three test functions with the program as a demo of the parser generator and the generated parser:
TestEmailParser.java, TestExpressionParser.java, TestTreeParser.java

Test Case Performance :

email.xml:

```
java TestEmailParser /u/cs200/public/test_cases/proj1/email.xml
email
        protocol = "SMPT"

        header
                from = "cs200@cs.rochester.edu"
                subject = "Test!"

                to
                        receiver = "ur.cs200@cs.rochester.edu"

                to
                        receiver = "csc200@mail.rochester.edu"

        body

                Here are the test cases for the parser generator.
                Good luck to everybody!

                attachment
                        fileName = "tese.tar.Z"
                        size = 2000


Running time in milliseconds: 36
```

expr1.xml:

```
[gchen@cycle3 phaseII]$ java TestExpressionParser
/u/cs200/public//test_cases/proj1/expr1.xml

Expression value: 4.0
Running time in milliseconds: 28
```

expr2.xml:

```
[gchen@cycle3 phaseII]$ java TestExpressionParser
/u/cs200/public//test_cases/proj1/expr2.xml

Expression value: 1.3333334
Running time in milliseconds: 28
```

expr3.xml:

```
[gchen@cycle3 phaseII]$ java TestExpressionParser /u/cs200/public//test_cases/pr
oj1/expr3.xml
Invalid XML document:
    Element / not expected in op_or_num.
```

NOTE: This input XML document is illegal because the operator at line 5 has only one input number.

expr4.xml:

```
[gchen@cycle3 phaseII]$ java TestExpressionParser /u/cs200/public//test_cases/pr
oj1/expr4.xml
Invalid XML document:
    Required attribute val is 20></NUM>
            <OP type =. It should be an integer value.
```

NOTE: This input XML document is also illegal because at line 7, the attribute value is missing the right double-quotes.

tree1.xml:

[gchen@cycle3 phaseII]$ java TestTreeParser /u/cs200/public/test_cases/proj1/tree1.xml 3

Number of nodes with value 18 at level 3: 1
Running time in milliseconds: 43

tree2.xml:

[gchen@cycle3 phaseII]$ java TestTreeParser /u/cs200/public/test_cases/proj1/tree2.xml 0

Number of nodes with value 18 at all levels: 118
Running time in milliseconds: 243

NOTE: Please note that "0" is used to indicate at all levels.

tree4.xml:

[gchen@cycle3 phaseII]$ java TestTreeParser /u/cs200/public/test_cases/proj1/tree4.xml 6

Number of nodes with value 18 at level 6: 46
Running time in milliseconds: 12404

tree6.xml:

[gchen@cycle3 phaseII]$ java TestTreeParser /u/cs200/public/test_cases/proj1/tre
e6.xml 7
Exception in thread "main" java.lang.OutOfMemoryError

NOTE: Because our program need to parse the document and store everything in a tree, and this file is almost 100MB. I believe that the CSUG machine is running out of memory before all can be put in the tree.