

Reading XML

Purpose:

This exercise completes the conversion of your Register to read and write XML documents.

What to hand in:

Hand in any file that has changed from the sample solution to last week's exercise. If you have not used the sample solution as your starting point, please hand in a complete solution.

Also, include the output from a sample run.

Please capture your homework in a single file for submission, in an easily reviewable order.

Problems:

Register DOM reading

Using the DOM (www.w3.org/DOM) model, we read in a register, using the same Xerces code we have used in the past (`doc = parser->parseURI(xmlFile);`).

We walk through this code, using DOM functions and perform exactly the same job we had done in IORegister in the previous exercise.

As one example, the Bank Id can be retrieved with a piece of code like:

```
DOMNodeList *children =
  doc->getElementsByTagName(XMLString::transcode("BankID"));
```

and since there is only one, the Node result is in *children if the result is not null:

```
const char *bankId; // We will "transcode" it to a local code page
DOMNode *child = *children;
XMLSize_t ccLen = child->getLength();
for ( XMLSize_t k = 0; k < ccLen; k++ ) {
  if ( child->item(k)->getNodeTypeId() == DOMNode::TEXT_NODE )
    bankId = XMLString::transcode(child->item(k)->getNodeValue());
}
```

For the iteration of the entries, the code may, similarly, look like this:

```
DOMNodeList *transactions = 0;
DOMNodeList *transactionHead =
  doc->getElementsByTagName(XMLString::transcode("Transactions"));
if ( transactionHead != 0 )
  transactions =
    doc->getElementsByTagName(XMLString::transcode("Transaction"));
if ( transactions != 0 ) {
  XMLSize_t cLen = children->getLength();
  for (XMLSize_t m=0; m<cLen; m++) {
    DOMNode *node = children->item(m);
    if ( node->getNodeTypeId() == DOMNode::ELEMENT_NODE )
      processTransaction(node); // This is the "virtual constructor"
  }
}
```

Notes:

This exercise should start with the XML schema (xsd file) from the Sample Solution.

You will want to reference W3C DOM information from the Resources tab (or www.w3.org/DOM) and the Xerces information from the Resources tab (or <http://xml.apache.org/xerces-c/apiDocs/>).

If the validation check is on (`parser->setFeature(XMLUni::fgDOMValidateIfSchema, true);`), and the validation reference is placed in the top-level register element (`xsi:schemaLocation="http://www.du.edu/~mschwartz/xml/namespace/register.xsd"`), then the file will be validated; otherwise, it will be checked to be well-formed.

Evaluation criteria:

Providing XML reading capabilities in the Register class	80%
Test run(s) to verify proper operation	20%