XML and Databases
Tutorial session 1: DOM

Kim.Nguyen@nicta.com.au

Week 2

XML

An XML document is basically a tree-like structure:

XML

An XML document is basically a tree-like structure:

```
<adressbook>
    <contact cat="Work">
            <name>
                <first>Sebastian</first>
                <last>Maneth</last>
            </name>
            <email>smaneth@cse.unsw.edu.au</email>
    </contact>
    <contact cat="Family">
            <name>
                <first>Yoh</first>
                <last>Nguyễn</last>
            </name>
            <address>4, foo St, London, UK</address>
    </contact>
</adressbook>
```

XML

An XML document is basically a tree-like structure:

```
<adressbook>
    <contact cat="Work">
            <name>
                <first>Sebastian</first>
                <last>Maneth</last>
            </name>
            <email>smaneth@cse.unsw.edu.au</email>
    </contact>
    <contact cat="Family">
            <name>
                <first>Yoh</first>
                <last>Nguyễn</last>
            </name>
            <address>4, foo St, London, UK</address>
    </contact>
</adressbook>
```

XML

An XML document is basically a tree-like structure:

```
<adressbook>
    <contact cat="Work">
        <name>
            <first>Sebastian</first>
            <last>Maneth</last>
        </name>
        <email>smaneth@cse.unsw.edu.au</email>
    </contact>
    <contact cat="Family">
        <name>
            <first>Yoh</first>
            <last>Nguyễn</last>
        </name>
        <address>4, foo St, London, UK</address>
    </contact>
</adressbook>
```

XML

An XML document is basically a tree-like structure:

```
<adressbook>
    <contact cat="Work">
        <name>
            <first>Sebastian</first>
            <last>Maneth</last>
        </name>
        <email>smaneth@cse.unsw.edu.au</email>
    </contact>
    <contact cat="Family">
        <name>
            <first>Yoh</first>
            <last>Nguyễn</last>
        </name>
        <address>4, foo St, London, UK</address>
    </contact>
</adressbook>
```

XML

An XML document is basically a tree-like structure:

```
<adressbook>
    <contact cat="Work">
            <name>
                <first>Sebastian</first>
                <last>Maneth</last>
            </name>
            <email>smaneth@cse.unsw.edu.au</email>
    </contact>
    <contact cat="Family">
            <name>
                <first>Yoh</first>
                <last>Nguyễn</last>
            </name>
            <address>4, foo St, London, UK</address>
    </contact>
</adressbook>
```

Manipulating XML Documents within a program

- "By hand" (reading the document as a pure text-file)

Manipulating XML Documents within a program

- "By hand" (reading the document as a pure text-file)
  $\Rightarrow$ Error-prone and too much to take into account

Manipulating XML Documents within a program

- "By hand" (reading the document as a pure text-file)
  $\Rightarrow$ Error-prone and too much to take into account
- Using DOM: mapping the tree sturcture on a Object Hierarchy

Manipulating XML Documents within a program

- "By hand" (reading the document as a pure text-file)
  $\Rightarrow$ Error-prone and too much to take into account
- Using DOM: mapping the tree sturcture on a Object Hierarchy

- Using SAX: event-based parsing (next week)

Manipulating XML Documents within a program

- "By hand" (reading the document as a pure text-file)
  $\Rightarrow$ Error-prone and too much to take into account
- Using DOM: mapping the tree sturcture on a Object Hierarchy
  $\Rightarrow$ Today's lecture and 1^{st} assignment
- Using SAX: event-based parsing (next week)

DOM Concepts

`http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/`

Node : A document is seen as a set of **nodes**.

$\Rightarrow$ Abstract concept, mapped on an abstract *class*

DOM Concepts

`http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/`

Node : A document is seen as a set of **nodes**.

$\Rightarrow$ Abstract concept, mapped on an abstract *class*

$\Rightarrow$ Derived to get many kinds of nodes: `ElementNode`, `TextNode`, `AttributeNode`, ...

DOM Concepts

`http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/`

Node : A document is seen as a set of **nodes**.

$\Rightarrow$ Abstract concept, mapped on an abstract *class*

$\Rightarrow$ Derived to get many kinds of nodes: `ElementNode`, `TextNode`, `AttributeNode`, ...

NodeList : to return sets of Nodes

DOM Concepts

`http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/`

Node : A document is seen as a set of **nodes**.

- $\Rightarrow$ Abstract concept, mapped on an abstract *class*
- $\Rightarrow$ Derived to get many kinds of nodes: `ElementNode`, `TextNode`, `AttributeNode`, ...

NodeList : to return sets of Nodes

Parser : to return a `DocumentNode` from a filename or URI

DOM Concepts

`http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/`

Node : A document is seen as a set of **nodes**.

$\Rightarrow$ Abstract concept, mapped on an abstract *class*

$\Rightarrow$ Derived to get many kinds of nodes: `ElementNode`, `TextNode`, `AttributeNode`, ...

NodeList : to return sets of Nodes

Parser : to return a `DocumentNode` from a filename or URI

...

Xerces: a DOM implementation

- Open source DOM implementation from the Apache foundation

Xerces: a DOM implementation

- Open source DOM implementation from the Apache foundation
- Two flavors : Xerces-J (Java), Xerces-C (C++)

Xerces: a DOM implementation

- Open source DOM implementation from the Apache foundation
- Two flavors : Xerces-J (Java), Xerces-C (C++)
- The one used for your assignments

Xerces: a DOM implementation

- Open source DOM implementation from the Apache foundation
- Two flavors : Xerces-J (Java), Xerces-C (C++)
- The one used for your assignments

Today: Focus on simple document loading with Xerces and document traversal

Xerces-J: Java packages

Program compilation:

```
javac -cp /usr/share/java/xercesImpl.jar  MyClass.java
```

Xerces-J: Java packages

Program compilation:

```
javac -cp /usr/share/java/xercesImpl.jar  MyClass.java
```

Java package inclusions:

```
// In xercesImpl.jar, this is the actual Xerces implementation
import org.apache.xerces.parsers.DOMParser;
// Abstract DOM classes, implemented by Xerces
import org.w3c.dom.*;

// Various java utilities
import java.util.*;
import java.io.*;
```

Xerces-J: Getting a Parser

```
public class DOMExample {
  DOMParser myparser;

  DOMExample() {
    try {
        myparser = new DOMParser();

    } catch(Exception e) {
      System.out. println ("Problem during   initialization  " + e);
    }
  };

  ⋮
```

Xerces-J: Loading a document

```
   ⋮
Document loadDocument(string filename){



   // first   parse   the   document
   myparser.parse(filename);
   //then   retrieve   it
   return myparser.getDocument();









}
```

Xerces-J: Loading a document

```
   ⋮
Document loadDocument(string filename){

  try {
  // first  parse  the  document
  myparser.parse(filename);
  //then  retrieve  it
  return myparser.getDocument();

  catch (Exception e) {
    System.out.println("could not parse document")
  [do something else here to finish properly]
  }
}
```

Xerces-J: Iterating the XML document (1/2)

```
void traverse(Node n){

  switch (n.getNodeType()){
```

Xerces-J: Iterating the XML document (1/2)

```
void traverse(Node n){

  switch (n.getNodeType()){
    case Node.DOCUMENT_NODE:
          ...
          break;
```

Xerces-J: Iterating the XML document (1/2)

```
void traverse(Node n){

  switch (n.getNodeType()){
    case Node.DOCUMENT_NODE:
          ...
          break;
    case Node.ELEMENT_NODE:
           ...
          break;
    case Node.TEXT_NODE:

         break;
    :
```

Xerces-J: Iterating the XML document (2/2)

```
    Node child;
     child  = n.getFirstChild();

    while ( child  != null){
       // recursive    call
        traverse ( child );

        child  = child.getNextSibling();
    };

}
```

Xerces-J: Invocation

```
public static void main(string [] args){
   ⋮

   DOMExample e = new DOMExample();
   Document d = e.loadDocument("filename.xml");
   e. traverse (d);
   ⋮

}
```

Xerces-J: Summary

- call the **parse()** method of a DOMParser to load the document

Xerces-J: Summary

- call the **parse()** method of a DOMParser to load the document
- a Document is *a Node*. In particular, its a *Node* for which: **getNodeType()** returns **Node.DOCUMENT_ELEMENT**

Xerces-J: Summary

- call the **parse()** method of a DOMParser to load the document
- a Document is *a Node*. In particular, its a *Node* for which: **getNodeType()** returns **Node.DOCUMENT_ELEMENT**
- tons of method in the Node class: **getNodeType()**, **getFirstChild()**, **getNextSibling()** and many others like **getNodeValue()**, **getAttributes()**,...

Xerces-J: Summary

- call the **parse()** method of a DOMParser to load the document
- a Document is *a Node*. In particular, its a *Node* for which: **getNodeType()** returns **Node.DOCUMENT_ELEMENT**
- tons of method in the Node class: **getNodeType()**, **getFirstChild()**, **getNextSibling()** and many others like **getNodeValue()**, **getAttributes()**,...
- consult the javadoc here for the full description: `http://java.sun.com/j2se/1.5.0/docs/api/index.html` browse the **org.w3c.dom.*** package.

Xerces-C: C++ headers inclusion and linking

Program compilation:

```
g++ -o myprog -lxerces-c myClass.cpp
```

Xerces-C: C++ headers inclusion and linking

Program compilation:

```
g++ -o myprog -lxerces-c myClass.cpp
```

C++ headers inclusion:

```
//DOM parser and xerces utilities
#include <xercesc/parsers/XercesDOMParser.hpp>
#include <xercesc/util/PlatformUtils.hpp>

//C++ utilities
#include <string>
#include <stdexcept>

// namespaces opening
using namespace xercesc;
using namespace std;
```

Xerces-C: Getting a Parser

```
class DOMExample {
  private:
  DOMParser * myparser;

  public:
  DOMExample() {
    try {
        XMLPlatformUtils::Initialize();
        myparser = new XercesDOMParser();

    } catch(Exception &e) {
      cout << "Problem during  initialization  " << e.what()
           << endl;
    }
  };
```

Xerces-C: Cleaning

```
~DOMExample() {
        myparser->release();
        XMLPlatformUtils::Terminate();
   };

   ⋮
```

Xerces-C: Loading a document

```
   ⋮
DOMDocument loadDocument(char * filename){


  XMLCh* str = XMLString::translate(filename);

  myparser->parse(filename);
  DOMDocument d = myparser->getDocument();
  XMLString::release(&str);
  return d;




}
```

Xerces-C: Loading a document

```
   ⋮
DOMDocument loadDocument(char * filename){

   try {
   XMLCh* str = XMLString::translate(filename);

   myparser->parse(filename);
   DOMDocument d = myparser->getDocument();
   XMLString::release(&str);
   return d;

   }catch (Exception &e) {
     cout << "could not parse document";
   [do something else here to finish properly]
   }
}
```

Xerces-C: Iterating the XML document (1/2)

```
void traverse(DOMNode n){

  switch (n->getNodeType()){
```

Xerces-C: Iterating the XML document (1/2)

```
void traverse(DOMNode n){

  switch (n->getNodeType()){
    case DOMNode::DOCUMENT_NODE:
          ...
          break;
```

Xerces-C: Iterating the XML document (1/2)

```
void traverse(DOMNode n){

  switch (n->getNodeType()){
    case DOMNode::DOCUMENT_NODE:
          ...
          break;
    case DOMNode::ELEMENT_NODE:
           ...
          break;
    case DOMNode::TEXT_NODE:

         break;
    :
```

Xerces-C: Iterating the XML document (2/2)

```
    DOMNode child;
    child  = n->getFirstChild();

    while ( child  != null ){
       // recursive    call
       traverse ( child );

       child  = child->getNextSibling();
    };

}
```

Xerces-C: Invocation

```
}; // closing the DOMExample class

int main(int argc, char ** argv){
  ⋮

  // this calls the constructor
  DOMExample e;
  DOMDocument d = e.loadDocument(argv[1]);
  e.traverse(d);
  ⋮

}
```

Xerces-C: Summary

Xerces-C is similar to Xerces-J with the following differences:

- Need to manually call **XMLPlatformUtils::Initialize()** and **XMLPlatformUtils::Terminate()**

Xerces-C: Summary

Xerces-C is similar to Xerces-J with the following differences:

- Need to manually call **XMLPlatformUtils::Initialize()** and **XMLPlatformUtils::Terminate()**
- Xerces-C uses exclusively **XMLCh *** as string. Use **XMLString::translate()** to convert back and forth to **char *** and **XMLString::release()** to free, don't use **delete** directly!

Xerces-C: Summary

Xerces-C is similar to Xerces-J with the following differences:

- Need to manually call **XMLPlatformUtils::Initialize()** and **XMLPlatformUtils::Terminate()**
- Xerces-C uses exclusively **XMLCh *** as string. Use **XMLString::translate()** to convert back and forth to **char *** and **XMLString::release()** to free, don't use **delete** directly!
- everything is passed by reference (e.g. **n->getNodeType()**) but never free a structure directly. Use **release()** methods when available, otherwise the pointer doesn't need to be freed.

Xerces-C: Summary

Xerces-C is similar to Xerces-J with the following differences:

- Need to manually call **XMLPlatformUtils::Initialize()** and **XMLPlatformUtils::Terminate()**
- Xerces-C uses exclusively **XMLCh *** as string. Use **XMLString::translate()** to convert back and forth to **char *** and **XMLString::release()** to free, don't use **delete** directly!
- everything is passed by reference (e.g. **n->getNodeType()**) but never free a structure directly. Use **release()** methods when available, otherwise the pointer doesn't need to be freed.
- The API documentation is here: `http://xerces.apache.org/xerces-c/apiDocs-2/`

Xerces-C: Summary

Xerces-C is similar to Xerces-J with the following differences:

- Need to manually call **XMLPlatformUtils::Initialize()** and **XMLPlatformUtils::Terminate()**
- Xerces-C uses exclusively **XMLCh *** as string. Use **XMLString::translate()** to convert back and forth to **char *** and **XMLString::release()** to free, don't use **delete** directly!
- everything is passed by reference (e.g. **n->getNodeType()**) but never free a structure directly. Use **release()** methods when available, otherwise the pointer doesn't need to be freed.
- The API documentation is here: `http://xerces.apache.org/xerces-c/apiDocs-2/`