

SAX Parsing

Prof. Jim Whitehead
CMPS 183, Spring 2006
April 19, 2006

Styles of XML Parsers

- Tree-based
 - Typically make use of DOM
 - Document Object Model, a standard way of viewing the elements of an XML document
 - Read an XML document into a tree structure in memory
- Event-based
 - SAX – Simple API for XML
 - Read the XML document, and generate events as the scanner reaches interesting parts of the document
 - Begin element, end element, etc.

XML Parser Tradeoffs

- Tree based parsers
 - Support complex interactions with the data, since it is all in memory
 - Can be used for transforming the XML data
 - Can keep reusing the data, only need to read once
 - Large XML document means large memory footprint
- Event based parsers
 - Support more simple forms of interaction with the data
 - Lower memory footprint, do not need to store the entire document in memory
 - Permits handling of larger documents
 - Somewhat easier to learn, and work with

SAX Parsers

- A standard interface for event-based XML parsing
- <http://www.saxproject.org/>
 - An open source project
 - Originally created by David Megginson

SAX Parser Model

- As the parser scans through an XML document, it will encounter various structural components of the document
 - Start & end of a document
 - Start & end of an element
 - A processing instruction
 - Start & end of a namespace URI
 - Character data
- You can create your own code to perform processing when these events occur

ContentHandler - SAX

- The primary interface within SAX is the ContentHandler interface
 - Defines methods associated with important structural conditions in the document
 - Start element → startElement()
 - End element → endElement()
 - Need to create implementations of these methods
- DefaultHandler is an implementation of these methods
 - People using SAX typically create a subclass of DefaultHandler
 - Implement only those handlers that are of interest to their application, using implementation in DefaultHandler for the others

ContentHandler Interface

```
package org.xml.sax;
public interface ContentHandler {
    public void setDocumentLocator(Locator locator);
    public void startDocument( ) throws SAXException;
    public void endDocument( ) throws SAXException;
    public void startPrefixMapping(String prefix, String uri)
        throws SAXException;
    public void endPrefixMapping(String prefix) throws SAXException;
    public void startElement(String namespaceURI, String localName,
        String qualifiedName, Attributes atts) throws SAXException;
    public void endElement(String namespaceURI, String localName,
        String qualifiedName) throws SAXException;
    public void characters(char[] text, int start, int length)
        throws SAXException;
    public void ignorableWhitespace(char[] text, int start, int length)
        throws SAXException;
    public void processingInstruction(String target, String data)
        throws SAXException;
    public void skippedEntity(String name) throws SAXException;
}
```

Examples

- Show MySAXApp example
- Run on example XML file
- Character data:
 - Character data inside an element can be returned in *multiple* chunks
 - Character data is returned between elements as well
 - To gather element data, need to create a simple state machine