JAXP

Java XML Parsing

Summary

- DOM: Tree-based API
- SAX: Event-based API
- XPath: Query-based API

XML example

```xml
<?xml version="1.0" encoding="UTF-8"?>

<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>
</catalog>
```

Exercise 1

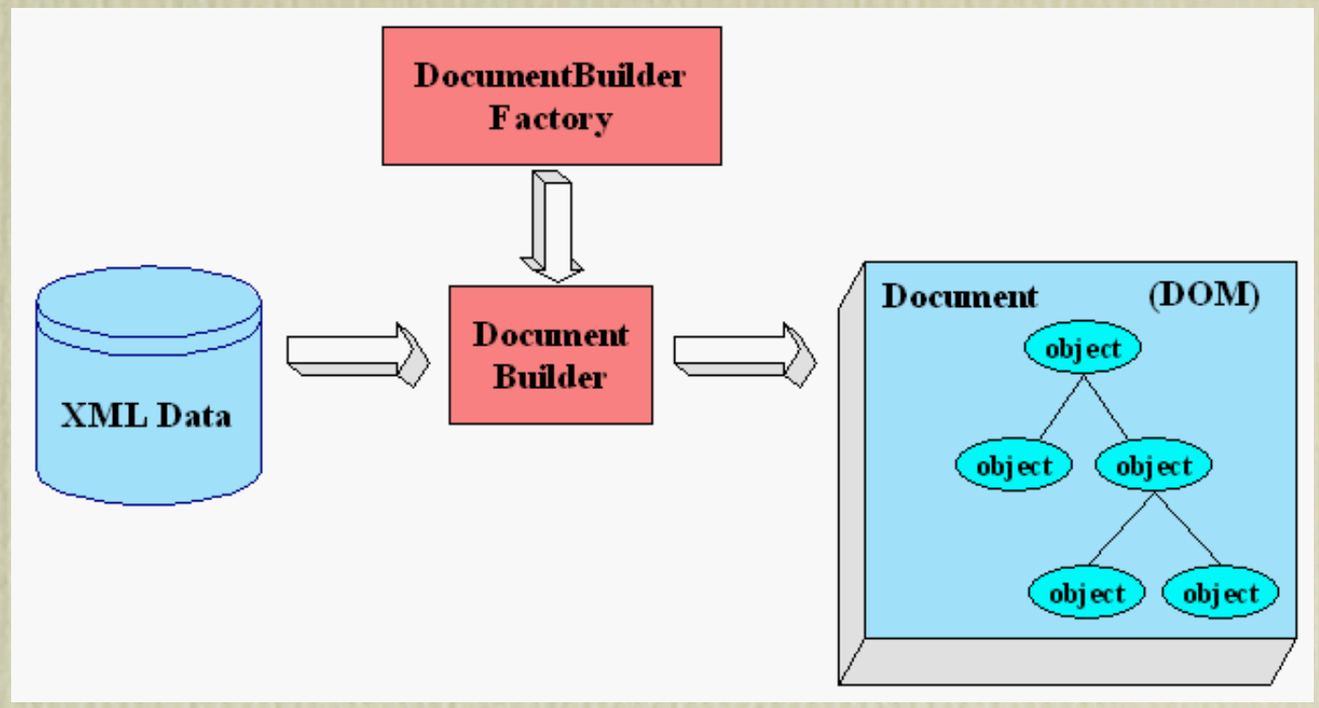
DOM

DOM

A tree-based API compiles an XML document into an internal tree structure.

This makes it possible for an application program to navigate the tree to achieve its objective.

DOM



- Node getParentNode(); //The parent of this node.
- NodeList getChildNodes(); //A NodeList that contains all children of this node.
- Node getFirstChild(); //The first child of this node.
- Node getLastChild(); //The last child of this node.
- Node getNextSibling(); //The node immediately following this node.
- Node getPreviousSibling(); //The node immediately preceding this node.

DOM example 1/3

```
public class DomDemo {

public static void main(String[] args) throws Exception {


Node node = readFile(new File("Books.xml"));
System.out.println("elementsCount: " + getElementsCount(node) + "\n");
//visitNodeTree(node);
}
```

DOM example 2/3

```
public static Document readFile(File file) throws Exception {
Document doc;
try {
    DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
    dbf.setValidating(false);
        DocumentBuilder db = dbf.newDocumentBuilder();
        doc = db.parse(file);
        return doc;
    } catch (SAXParseException ex) {
        throw (ex);
    } catch (SAXException ex) {
        Exception x = ex.getException(); // get underlying Exception
        throw ((x == null) ? ex : x);
    }
}
```

DOM example 3/3

```
public static int getElementsCount(Node node) {
if (null == node) {
    return 0;
}
int sum = 0;
boolean isElement = (node.getNodeType() == Node.ELEMENT_NODE);
if (isElement) {
    sum = 1;
}
NodeList children = node.getChildNodes();
if (null == children) {
    return sum;
}

for (int i = 0; i < children.getLength(); i++) {
    sum += getElementsCount(children.item(i)); // recursive call
}
return sum;
}
```

Exercise 11

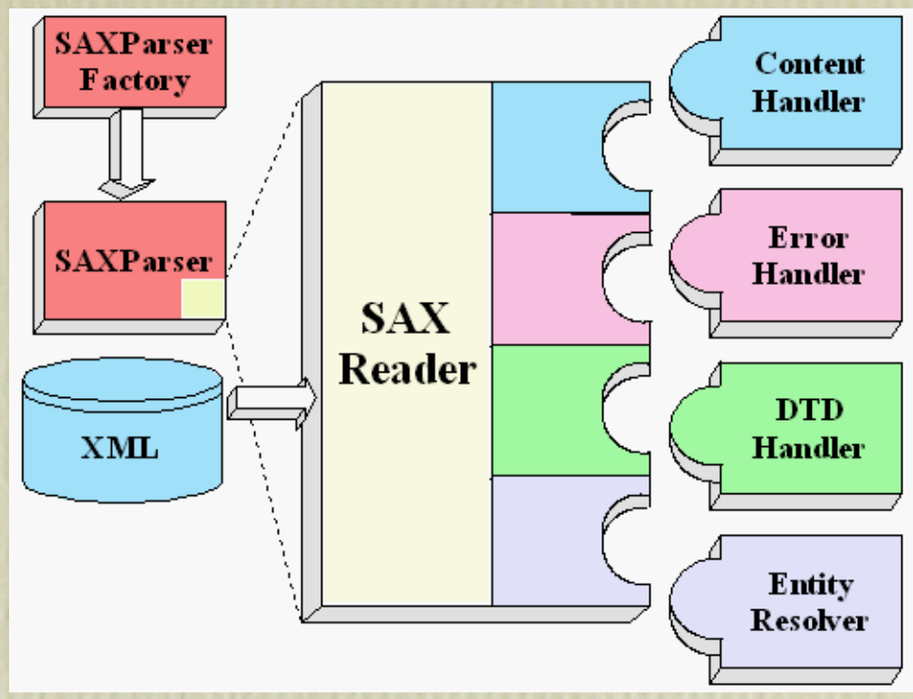
SAX

SAX

An event-based API reports parsing events (such as the start and end of elements) to the application using callbacks.

The application implements and registers event handlers for the different events.

SAX



- void characters(char[] ch, int start, int length);
- void startDocument();
- void startElement(String name, AttributeList attrs);
- void endElement(String name);
- void endDocument();
- void processingInstruction(String target, String data);

SAX example 1/2

```
public class SaxDemo {

    public static void main(String[] args) {
        if (args.length != 1) {
            new SaxDemo("Books.xml");
        } else {
            new SaxDemo(args[0]);
        }
    }
```

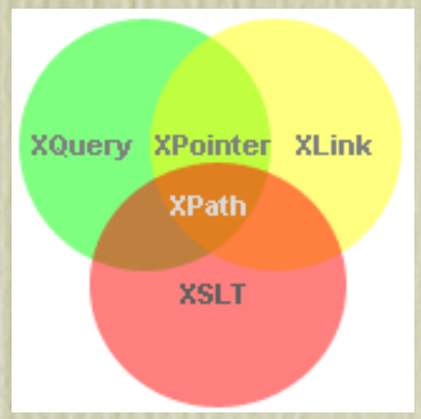
SAX example 2/2

```
public SaxDemo(String filename) {

    DefaultHandler handler = new MySaxHandler();
    // The Handler will be explained in Eclipse

    SAXParserFactory factory = SAXParserFactory.newInstance();

    try {
        SAXParser saxParser = factory.newSAXParser();
        saxParser.parse(new File(filename), handler);
    } catch (Throwable t) {
        t.printStackTrace();
    }

    System.exit(0);
}
```

Exercise III

XPath

XPath



- XPath is a syntax for defining parts of an XML document
- XPath uses path expressions to navigate in XML documents
- XPath contains a library of standard function
- XPath is a major element in XSLT
- XPath is a W3C recommendation
- http://www.w3schools.com/xpath/xpath_syntax.asp

XPath Syntax

Expression	Description
nodename	Selects all child nodes of the named node
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes

XPath example

```
public class XPathDemo {

  public static void main(String[] args)
   throws ParserConfigurationException, SAXException,
          IOException, XPathExpressionException {

    DocumentBuilderFactory domFactory = DocumentBuilderFactory.newInstance();
    domFactory.setNamespaceAware(true);

    DocumentBuilder builder = domFactory.newDocumentBuilder();
    Document doc = builder.parse("Books.xml");

    XPath xpath = XPathFactory.newInstance().newXPath();
    XPathExpression expr = xpath.compile("/catalog/book/title/text()");
    Object result = expr.evaluate(doc, XPathConstants.NODESET);

    NodeList nodes = (NodeList) result;
    for (int i = 0; i < nodes.getLength(); i++) {
      System.out.println(nodes.item(i).getNodeValue());
    }
  }
}
```

Summary

- DOM, Tree-based API
- SAX, Event-based API
- XPath, Query language