The logical parser

1. The logical parser

As we have already said, the JaxMe parser for XML Schema, is an application of multiple layers. There is a generic parser, which is mostly independent of a certain XML language. The second layer is the syntax parser, which you typically do not want to use. Most probably you are mainly interested in the topmost layer: The logical parser.

The logical parser presents the XML Schema in a way, which you will definitely like, because it is easy to use and frees you from the burden to understand XML Schema. It handles groups, restrictions, extensions, redefinitions and all that kind of stuff for you. Ideally you do not even notice, that they are in use.

- Using the logical parser
- Accessing locical context information

2. Using the logical parser

The logical parser is used as follows:

```
import java.io.File;
import java.io.FileInputStream;
import org.apache.ws.jaxme.xs.XSElement;
import org.apache.ws.jaxme.xs.XSParser;
import org.apache.ws.jaxme.xs.XSSchema;
import org.xml.sax.InputSource;

XSParser xsp = new XSParser();
// xsp.setValidating(false);  // In case your schema does not contain a schema decl
    File f = new File("myschema.xsd");
InputSource isource = new InputSource(new FileInputStream(f));
isource.setSystemId(f.toURL().toString());
XSSchema schema = xsp.parse(isource);

// Print the names of all global elements:
XSElement[] elements = schema.getElements();
for (int i = 0;  i < elements.length;  i++) {
  System.out.println(elements[i].getName());
}
```

3. Accessing logical context information

Within your own beans or bean methods, it might be interesting from time to time, whether you are currently within an imported or included schema. If so, you might also want to know about the outer schemas. Access to these items is provided through

```
getXSSchema().getContext().getXSLogicalParser().getSyntaxSchemas()
```