SAX Parser

SAX Parser

Introduction

SAX (Simple API for XML) is an event based sequential access parser API. SAX provides a mechanism for reading on each piece of XML document sequentially

How the SAX parser works

- It takes the occurrences of the components of a input document as it reads and tells the client as it reads from the document.
- But a SAX parser serves the client application always only with pieces of the document at any given time
- Some certain methods will be invoked automatically (implicitly) as a callback
- When some certain events occur. These methods should not be called explicitly by client

Handlers

The SAX API defines four kinds of handlers: content handlers, DTD handlers, error handlers, and entity resolvers. Applications normally only need to implement those interfaces whose events they are interested in; they can implement the interfaces in a single object or in multiple objects. Handler implementations should inherit from the base classes provided in the module `xml.sax.handler`, so that all methods get default implementations.

ContentHandler

Methods such as startDocument, endDocument, startElement, and endElement are invoked when an XML tag is recognized. This interface also defines the methods characters and processingInstruction, which are invoked when the parser encounters the text in an XML element or an inline processing instruction, respectively.

ErrorHandler

Methods error, fatalError, and warning are invoked in response to various parsing errors. The default error handler throws an exception for fatal errors and ignores other errors (including validation errors). That's one reason you need to know something about the SAX parser, even if you are using the DOM. Sometimes, the application may be able to recover from a validation error. Other times, it may need to generate an exception. To ensure the correct handling, you'll need to supply your own error handler to the parser.

DTDHandler

Defines methods you will generally never be called upon to use. Used when processing a DTD to recognize and act on declarations for an unparsed entity.

EntityResolver

The resolveEntity method is invoked when the parser must identify data identified by a URI. In most cases, a URI is simply a URL, which specifies the location of a document, but in some cases the document may be identified by a URN a public identifier, or name, that is unique in the web space. The public identifier may be specified in addition to the URL. The EntityResolver can then use the public identifier instead of the URL to find the document for example, to access a local copy of the document if one exists.

Disadvantages

The event-view of a document is not intuitive to many of today's server-side, object oriented Java developers. SAX also does not support modifying the document, nor does it allow random access to the document

When to use SAX Parser

- If the input document is too big for available memory
- Process the document in small contiguous chunks of input

Referenced links

http://download.oracle.com/javaee/1.4/tutorial/doc/JAXPIntro4.html
http://totheriver.com/learn/xml/xmltutorial.html
http://www.java-samples.com/showtutorial.php?tutorialid=152
http://www.ibm.com/developerworks/xml/library/x-tiphandl/index.html
http://java.sun.com/developer/Books/xmljava/ch03.pdf

Sample

Name	Size	Creator	Creation Date	Comment
sax.zip	3 kB	MULLAIKANI A.	Jul 20, 2012 06:51	Sax Parser

Exercise

- Create a Handler by implementing ContentHandler to know more about callback.