Intelligent Internet Technologies

Lecture 7.

XML Document Object Model

Alexandra V. Vitko

KNURE, AI Department, alexandra_vitko@yahoo.com

What is XML Parser?

- A program or module that checks a well-formed syntax and provides a capability to manipulate XML data elements:
  - navigate through the XML document
  - extract or query data elements
  - add/delete/modify data elements
- Use **API** (**A**pplication **P**rogramming **I**nterface)

API for XML Programming

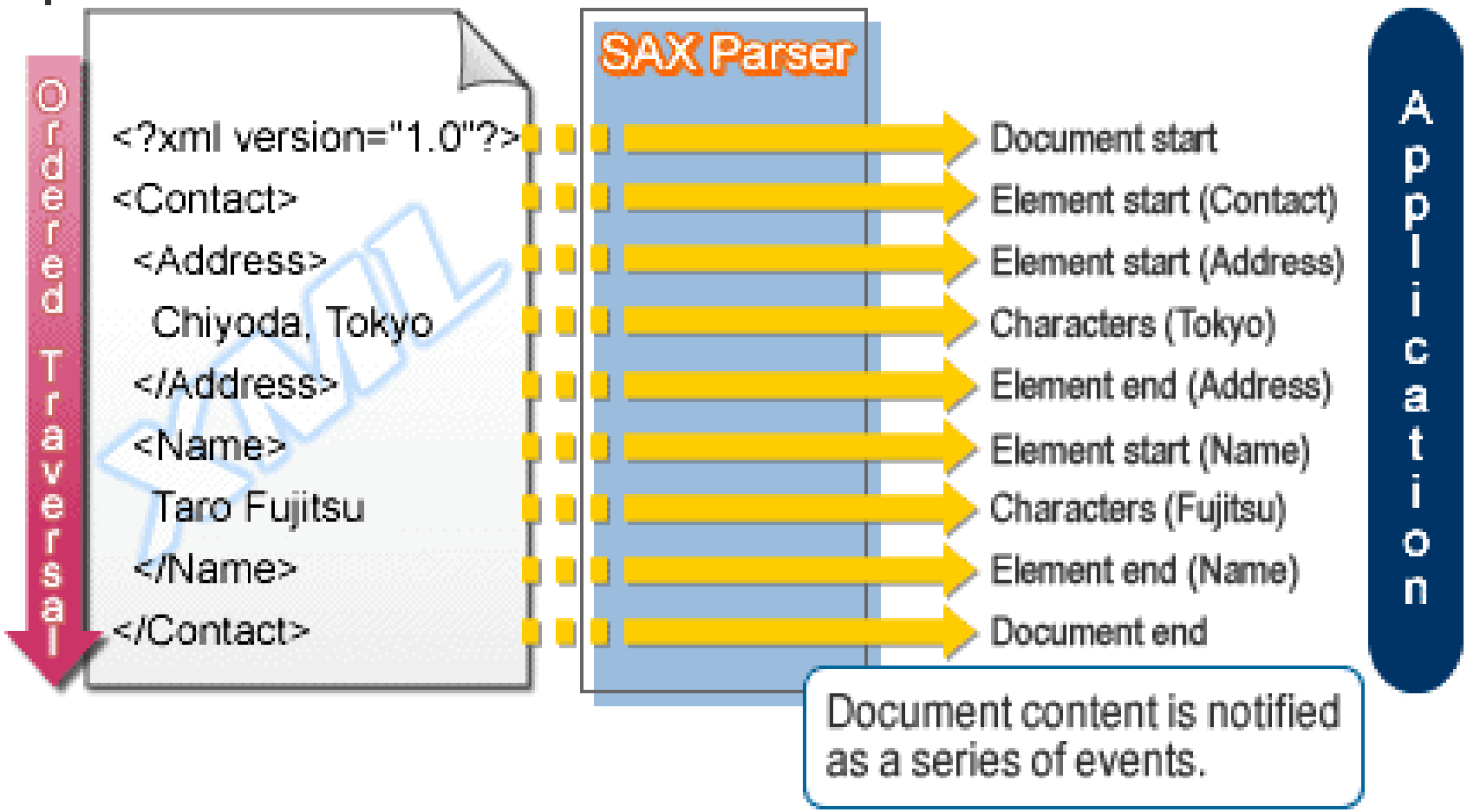
- Standard APIs for XML programming in Java
  - Document Object Model (DOM)
    - Defined by W3C
    - Logical model
    - Generates tree model
  - Simple API for XML (SAX)
    - “Event” model
  - Others
    - DOM4J, JDOM, XNI, etc...

Simple API for XML (SAX)

- Java Event-based parser for XML
- Reports parsing events through call-backs.

- http://www.saxproject.org/

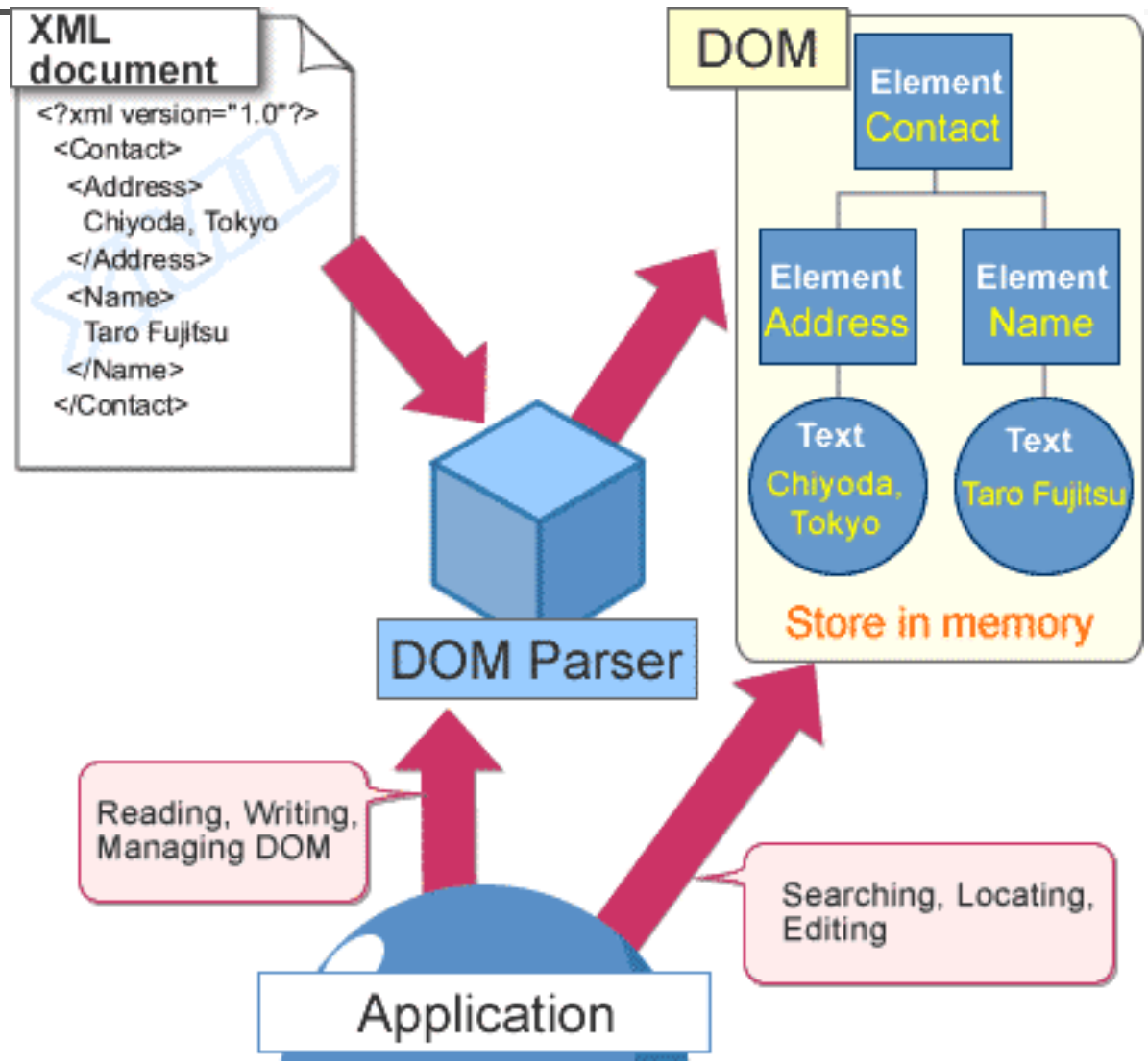
Processing with SAX



Document Object Model (DOM)

- W3C recommendation
- Logical model of XML document
- DOM tree structure
  - Document, element, text... – parts of model
  - Nodes, NodeList, ... – parts of tree

Processing with DOM



DOM Parsers

- Parses the entire document into a DOM tree.
- Provide functions to examine pieces of the tree
- Provides a createDocument interface which generates a XML document from the DOM tree

SAX vs. DOM

- Unlike DOM (Document Object Model), SAX does not store information in an internal tree structure
- Because of this, SAX is able to parse huge documents (think gigabytes) without having to allocate large amounts of system resources

DOM vs. SAX

- SAX does not allow random access to the file; it proceeds in a single pass, firing events as it goes
- SAX makes it hard to implement cross-referencing in XML (ID and IDREF) as well as complex searching routines

When to use DOM

- If changing XML file - inserting or deleting elements or changing structure
- Navigating to parts of XML file
- Complex hierarchies

- SAX only suitable for sequential processing
  - Can’t look ahead, can not look back
  - Suitable for transformations, validation
  - Very suitable for large documents

Introduction to DOM

- What is the DOM?
  - Set of standards agreed upon by the World Wide Web Consortium (W3C)
  - From http://www.w3.org/DOM/

Logical object model

The Document Object Model is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of XML documents.

Why is the DOM important to web developers?

- W3C sets standards based upon recommendations provided by its members. Some of the current members who developing top Web Browsers are Microsoft, Mozilla Foundation and Opera Software.

- By understanding the DOM, you will be **able to write scripts once that function in multiple browsers.**

Browser Versions Supporting DOM

- Mozilla's DOM Support
- Internet Explorer's DOM Support
- Opera 7

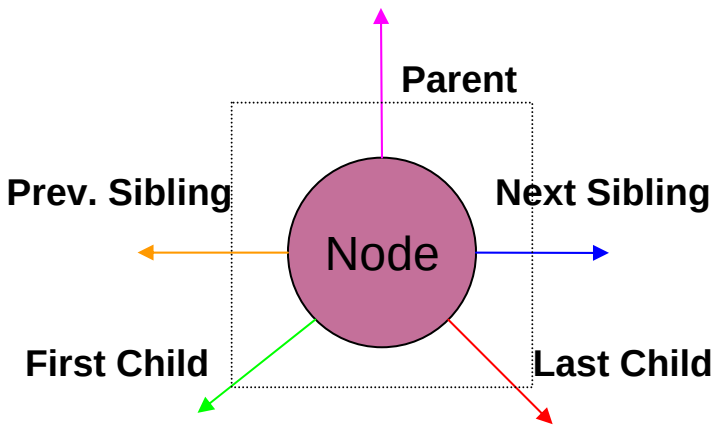
! Note - Loading XML is different in different browsers

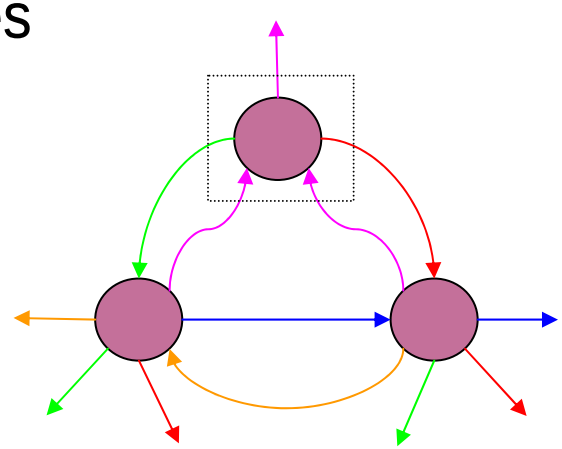
DOM Interfaces

- Document
- Node
- Nodelist
- Element
- Attr
- CharacterData
- Text
- Comment
- ProcessingInstruction
- CDATASection

Document Object Model

- Tree model
  - Node
    - Type, name, value
    - Attributes
    - Parent node
    - Previous, next sibling nodes
    - First, last child nodes
  - Collections
    - Lists
    - Maps

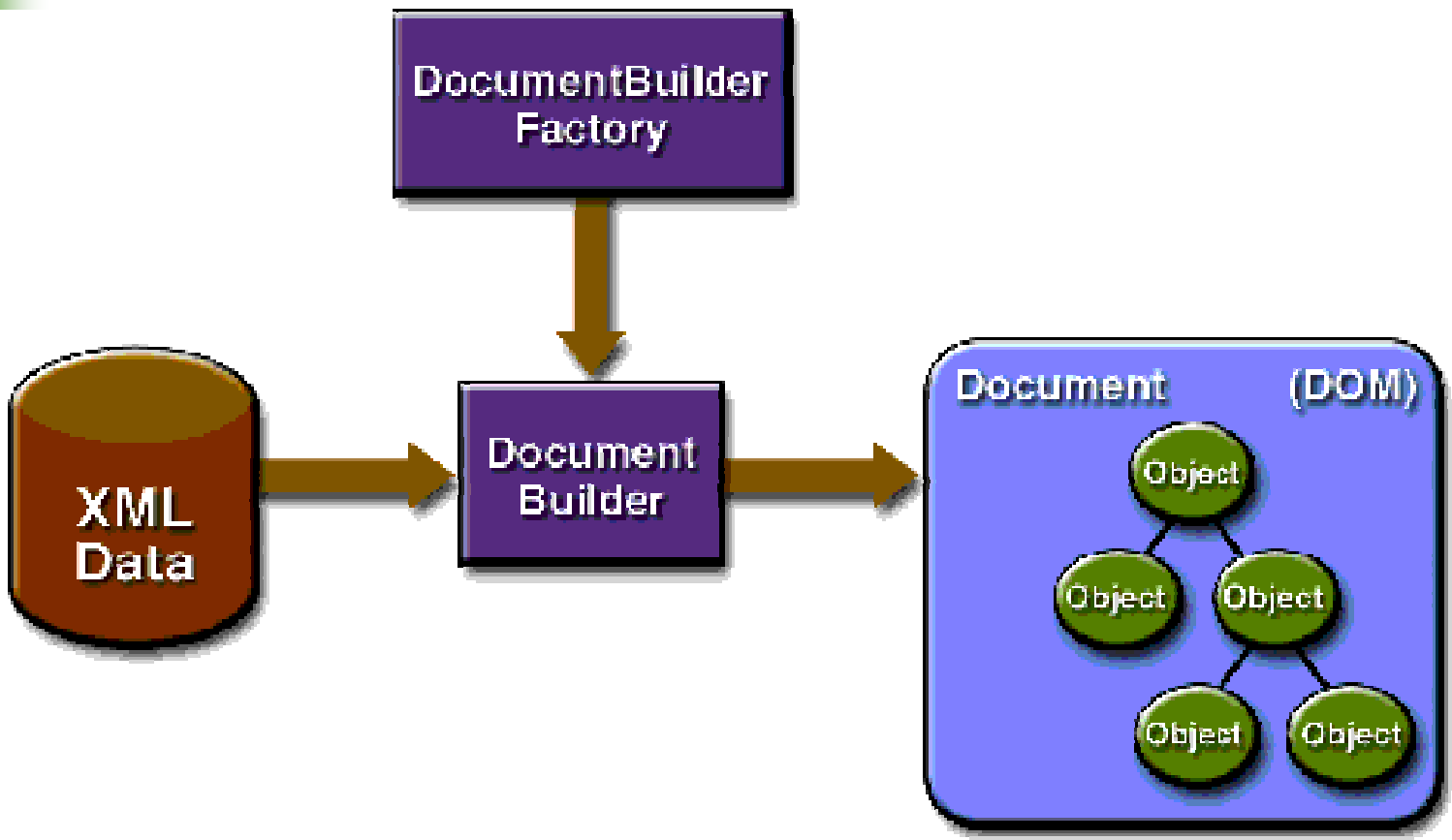




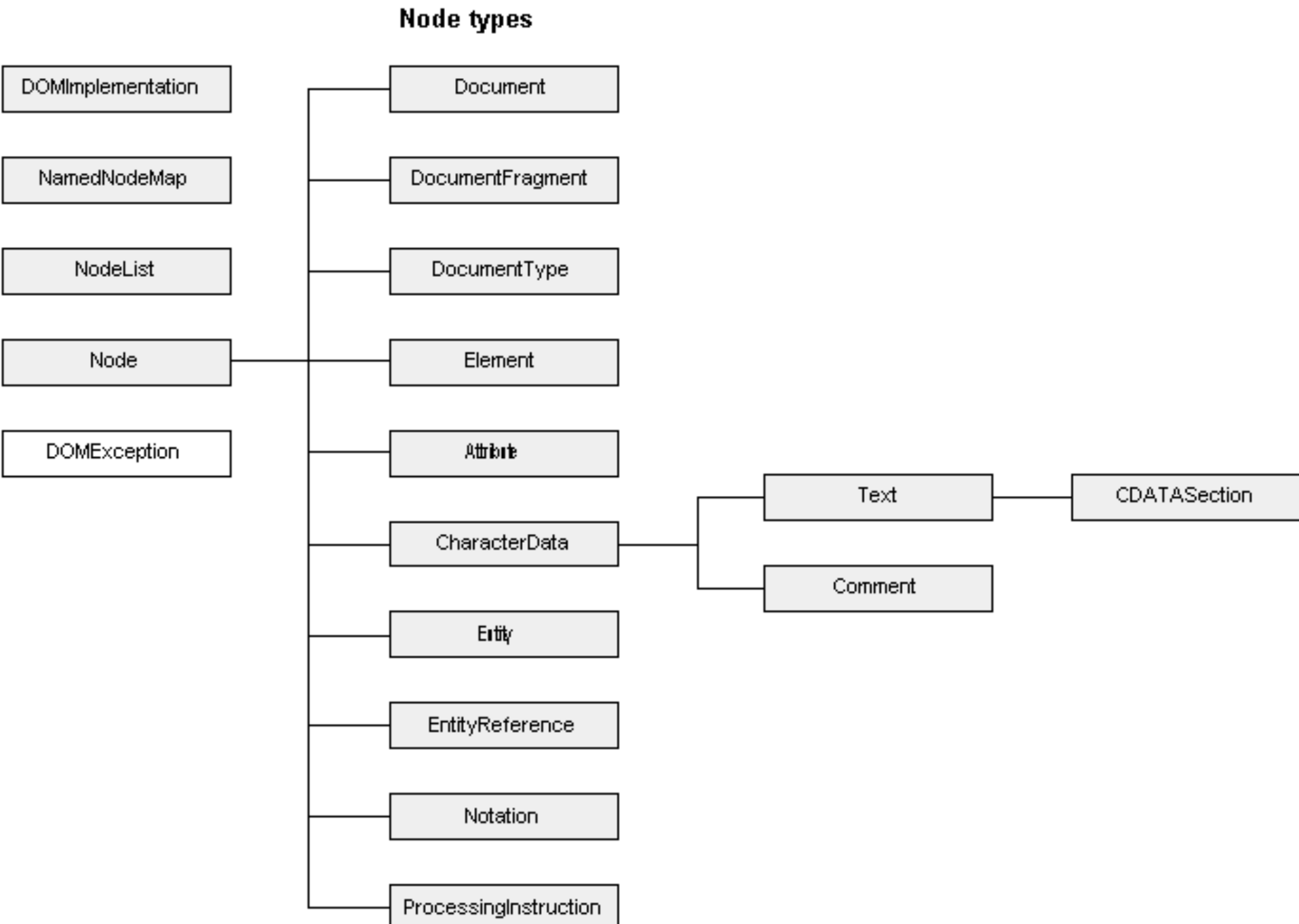
DOM API Framework

- DocumentBuilderFactory - creates instance of factory
- DocumentBuilder – ensures the XML loading and processing
- parse() - runs the DocumentBuilder
- Document - the top level document
- Node - any sort of node in tree
- NodeList - a list of nodes (children)
- NamedNodeMap - use to get attributes
- Exceptions

Document Builder Factory



DOM API


Node types
DOMImplementation
NamedNodeMap
NodeList
Node
DOMException
Document
DocumentFragment
DocumentType
Element
Attribute
CharacterData
Entity
EntityReference
Notation
ProcessingInstruction
Text
Comment
CDATASection

Node Methods (Basic)

Short getNodeType ()

Node.DOCUMENT_NODE, Node.ELEMENT_NODE, Node.TEXT_NODE, Node.COMMENT_NODE, etc.

String getNodeName ()

String getNodeValue()

NodeList getChildNodes ()

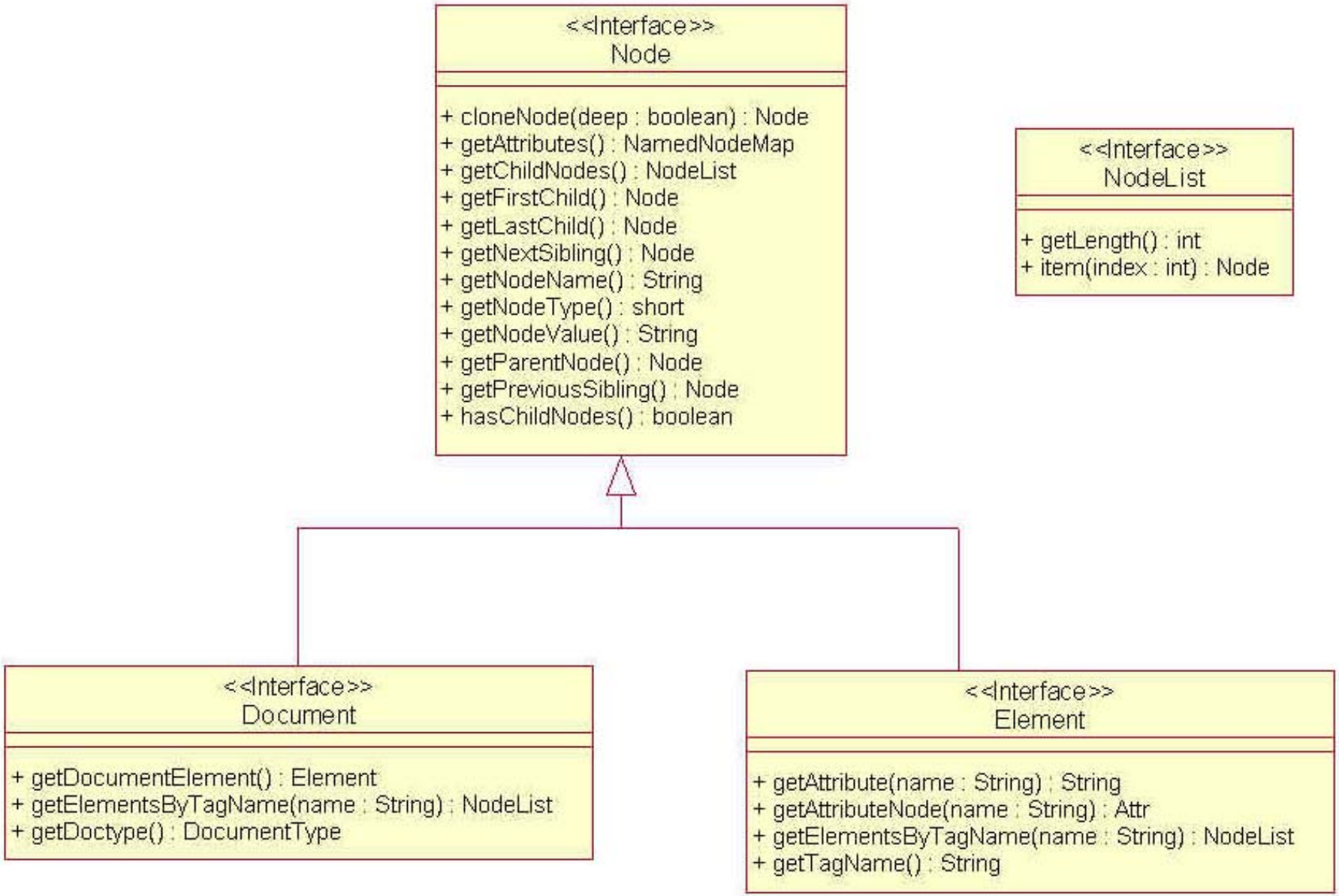
Boolean hasChildNodes ()

NamedNodeMap getAttributes ()

nodeName and nodeValue

Node	nodeName	nodeValue
Attr	attrib name	attrib value
CDataSection	#cdata-section	content
Comment	#comment	comment
Document	#document	null
DocumentType	document type name	null
Element	tag name	null
Processing Instruction	target	rest of content
Text	#text	text content

Methods for Different Interfaces



Traversing Documents

- To get root element:

```
root = xmlDoc.documentElement
```

- To get child elements and name/value:

```
firstChild
nodeName
nodeValue
childNodes
```

Modifying Documents

- To modify document:

```
createElement
createTextNode
appendChild
removeChild
```

- For example:

```
first = root.firstChild
second = root.ChildNodes[1]
firstName = first.nodeName
firstText = first.firstChild.nodeValue
new = createElement(‘book’)
root.appendChild(new)
```

Summary

- DOM provides W3C standard for accessing elements on XML document.
- Allows access to all elements, based on standard methods to traverse tree or find elements by ID or tag name.
- Allows modification of tree, creating new elements or replacing content.

Read More in

- W3C
  - http://www.w3.org/DOM/