

# Computer Science E-259

XML with J2EE

## Lecture 3: DOM Level 3

16 February 2006

David J. Malan

`malan@post.harvard.edu`

1

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Computer Science E-259

## Last Time

- XML 1.1
- SAX 2.0.2
- JAXP 1.3 and Xerces 2.7.1
- Parsing
- My First XML Parser

2

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Last Time

## A Representative Document

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE students SYSTEM "student.dtd">

<!-- This is an XML document that describes students -->
<?studentdb displaydesc="true"?>
<students>
  <student id="0001">
    <name>Jim Bob</name>
    <status>graduate</status>
    <dorm/>
    <major>Computer Science & Music</major>
    <description>
      <![CDATA[ <h1>Jim Bob!</h1>
        Hi my name is jim. I look like
         ]]>
    </description>
  </student>
  <student id="0002">
    ...
  </student>
</students>
```

3

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Last Time

## Grammars

```
S ::= ' '
S ::= SS
S ::= '(' S ')'
```

()

(( ))

() () ()

((((((((((((() )))))))) )) ((((((((((() )))))))) ))

4

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Last Time

## Grammars

```
Sentence ::= Phrase ' ' Verb ' ' Phrase
Phrase   ::= Noun
Phrase   ::= Adj ' ' Phrase
Adj      ::= 'BIG'
Adj      ::= 'GREEN'
Noun     ::= 'JIM'
Noun     ::= 'CHEESE'
Verb     ::= 'ATE'
```

JIM ATE CHEESE

BIG JIM ATE GREEN CHEESE

BIG CHEESE ATE JIM

BIG CHEESE ATE GREEN GREEN BIG GREEN BIG CHEESE

GREEN JIM ATE GREEN BIG JIM

5

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Last Time

## SAX 2.0.2

```
startDocument();
endDocument();
startElement('.',.);
endElement('.');
characters('.');
...
```

6

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Computer Science E-259

## This Time

- DOM Level 3
- JAXP 1.3 and Xerces 2.7.1
- My First XML Parser

7

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

## DOM Level 3

### Why?

- The SAX API has a number of important advantages...
  - You can write very fast SAX parsers
    - No memory to allocate, data structures to link
    - Fire and forget
  - It is useful for large documents
    - Loading the whole document into memory is prohibitive
  - It is easy to use
- ...but it doesn't solve every problem
  - Need to have an internal data structure for some applications
  - To follow links in information (especially backwards ones)
  - To perform operations that require having multiple pieces of the document at the same time
- Enter the DOM...

8

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## By Example

```
<!-- This is an example document -->
```

```
<students>
```

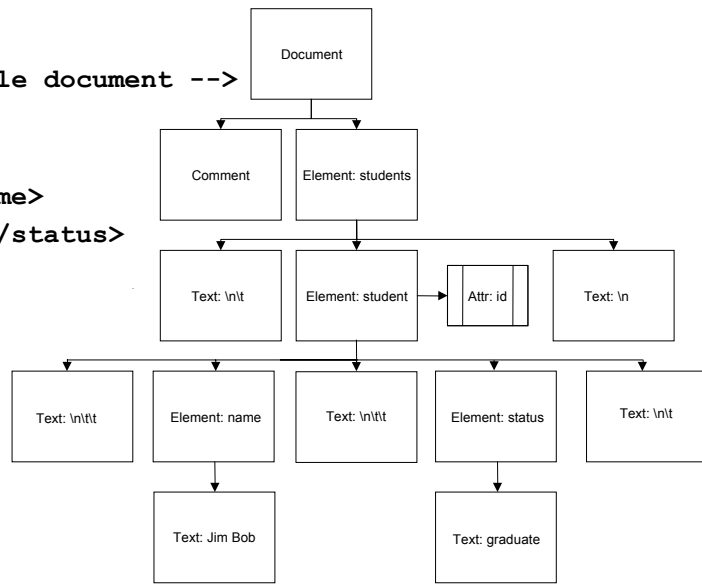
```
  <student id="0001">
```

```
    <name>Jim Bob</name>
```

```
    <status>graduate</status>
```

```
  </student>
```

```
</students>
```



# DOM Level 3

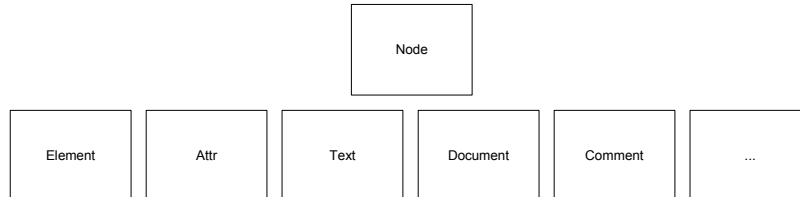
## By Definition

- The result of parsing a document with a DOM parser is a DOM tree that matches the structure of that document
- After parsing is complete, the tree data can be used by application
- A DOM tree may be different than trees you have seen in the past
  - There are different types of nodes in the tree
  - Only some nodes can have children
  - For nodes that are allowed children, there is no limit on the number of child nodes
  - Attributes can grow the tree "horizontally" as well as vertically
  - Can think of a DOM tree as a hybrid of list and tree concepts

# DOM Level 3

## By Definition

- Presents a language-neutral interface for manipulating hierarchical documents
  - Used for both (X)HTML and XML
- Object hierarchy: every object type represents a component of the XML information model



11

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Relationship with SAX

- Although the result of using a DOM parser and a SAX parser may seem very different...
- ...both DOM and SAX are methods for encoding the structure and content of an XML document
  - SAX does this by the type and order of events that are invoked
  - DOM does this by using objects in a tree data-structure
- In fact, it is possible to create a DOM tree from a series of SAX events
  - One of the things you have to do in Project 1!

12

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## A Sample Document

```
<students>
  <student id="0001"/>
</students>
```

13

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

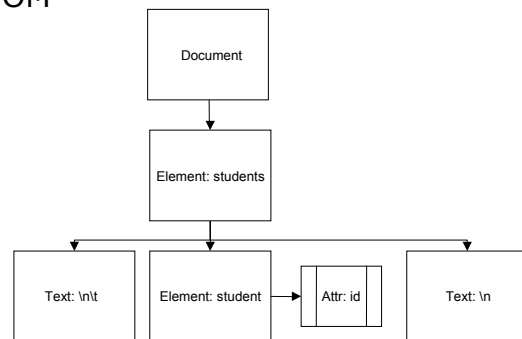
# DOM Level 3

## Relationship with SAX

Document

```
<students>
  <student id="0001"/>
</students>
```

DOM



Handler

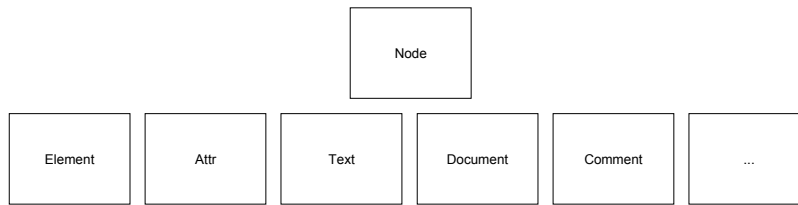
```
startDocument();
startElement("students", {});
characters("\n\t");
startElement("student", {("id", "0001")});
endElement("student");
characters("\n");
endElement("students");
endDocument();
```

14

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Nodes



15

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Nodes

- All objects in the DOM tree implement a **Node** interface
- The **Node** interface contains methods to get
  - a name (used to store the name of the node)
  - a value (used to store the value of the node, if any)
  - a child list (a list of nodes that are children of the current node)
  - a list of attributes
  - the parent of the node
- Not every node subtype has meaningful data to return from these methods (e.g., only **Element** has attributes)
- Provides most of the functionality you ever want on a node
  - Get the children of an element
  - Get the value of a text node
  - Modify the DOM tree by adding or removing elements
  - ...

16

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Interfaces

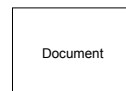
- The W3C defined the DOM interfaces for a language-neutral data structure
  - In Java, these interfaces are in the `org.w3c.dom` package
- In any one language, applications can use the interfaces without ever "seeing" the actual implementation
  - In Java, you program against `org.w3c.dom.Node` and not, e.g., `org.apache.xerces.dom.NodeImpl`
- In My First XML Parser, we
  - don't use the `org.w3c.dom` interfaces
  - simplify by using a `Node` base class and subclasses instead of separating an interface from an implementation

17

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Document



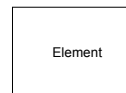
- At the root of the XML DOM is a **Document** object
  - This is not the same as the root element!
- Can have content that is valid at the top level of an XML document
  - Processing instructions, comments
- Also contains the (one and only one) document element
- Contains functions for creating other types of DOM Nodes
  - Remember, the DOM specifies an interface, not an implementation!
  - This design pattern is known as a *factory*

18

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Element



- The most "interesting" object in the DOM tree, as it makes up most of the structure
- Adds a few additional utility functions on top of the Node interface for manipulating attributes

19

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

## Attr



- Somewhat special in the DOM hierarchy in that it is not part of the DOM tree proper
- Elements have a list of attributes attached

20

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# DOM Level 3

...



- Most of the other DOM types are relatively simple, and use the name and value fields defined by the base Node interface
- `CDataSection`, `Comment`, `ProcessingInstruction`, and `Text`, for instance, all fall into this category

21

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

## JAXP 1.3 and Xerces 2.7.1

`DocumentBuilderDemo`

```
javax.xml.parsers.DocumentBuilderFactory
    javax.xml.parsers.DocumentBuilder
        org.w3c.dom.*
        ...
```

22

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# JAXP 1.3 and Xerces 2.7.1

## Namespaces

- Many of JAXP's APIs mention XML namespaces
- Namespaces are a way to specify groupings of tag and attribute names so that two names with different meanings don't "collide"
  - For example, the element "name" may refer to a person in a student markup language, but may refer to a book in a library markup language
- Allow you to specify a namespace, local name, and fully qualified name

▪ **studentml:name**

Namespace URI      Local Name

QName

- More to come...

23

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# My First XML Parser

DOMBuilderDemo

`cscie259.project1.mf.*`

24

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Computer Science E-259

## Next Time

- CSS Level 2
- XPath 1.0
- XSLT 1.0
- TrAX
- Project 2

25

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.

# Computer Science E-259

## XML with J2EE

### Lecture 3: DOM Level 3

16 February 2006

David J. Malan

`malan@post.harvard.edu`

26

Copyright © 2006, David J. Malan <malan@post.harvard.edu>. All Rights Reserved.