Document Object Model 1.0

Απόδοση στα
Ελληνικά της παρουσίασης του Alan Robinson
από τον Γιάννη Παπαδάκη
Νοέμβριος 2007

Ενα τμήμα XML

```
<seq id="my_seq" name="NUCLEAR RIBONUCLEOPROTEIN">
  <dbxref>
    <database>SWISS-PROT</database>
    <unique_id>P09651</unique_id>
  </dbxref>
  <residues type="aa">
SKSESPKEPEQLRKLFIGGLSFETTDESLRSHFEQWGTLTDCVVMRDPNTKRS
RGFGFVTYATVEEVDAAMNARPHKVDGRVVEPKRAVSREDSQRPGAHLTVKKI
FVGGIKEDTEEHHLRDYFEQYGKIEVIEIMTDRGSGKKRGFAFVTFDDHDSVD
KIVIQKYHTVNGHNCEVRKALSKQEMASASSSQRGRSGSGNFGGGRGGGFGGN
DNFGRGGNFSGRGGFGGSRGGGGYGGSGDGYNGFGNDGGYGGGGPGYSGGSRG
YGSGGQGYGNQGSGYGGSGSYDSYNNGGGRGFGGGSGSNFGGGGSYNDFGNYN
NQSSNFGPMKGGNFGGRSSGPYGGGGQYFAKPRNQGGYGGSSSSSSYGSGRRF
  </residues>
</seq>
```

Ενα XML DTD

```
<?xml version='1.0' encoding="US-ASCII"?>

<!DOCTYPE biosequence [
 <!ELEMENT seq (dbxref*, residues?) >
 <!ATTLIST seq      id     ID     #REQUIRED
                    name   CDATA  #IMPLIED
                    length CDATA  #IMPLIED >

<!ELEMENT residues (#PCDATA)>
 <!ATTLIST residues type   (dna | rna | aa) #REQUIRED>
]>
```

Χρησιμοποιώντας έναν XML Parser

- Τρία βασικά βήματα:
  - Δημιουργία του parser object
  - Ανάθεση του XML εγγράφου στον parser
  - Επεξεργασία των αποτελεσμάτων
- Γενικά, το «γράψιμο» XML – XML serialization δεν υποστηρίζεται από τους parsers (αν και ορισμένοι υλοποιούν proprietary μηχανισμούς)

Τύποι Parser

- Υπάρχουν πολλοί τρόποι κατηγοριοποίησης των parsers:
  - Validating και non-validating parsers
  - Parsers που υποστηρίζουν το Document Object Model (DOM)
  - Parsers που υποστηρίζουν το Simple API για XML (SAX)
  - Parsers γραμμένοι σε ορισμένη γλώσσα (Java, C++, Perl, κλπ.)

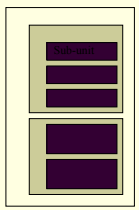
Non-validating Parsers

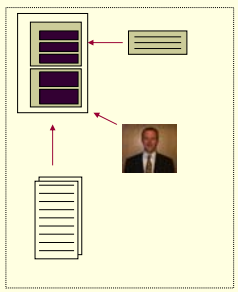
- Γρήγοροι και αποδοτικοί
  - Είναι κοπιαστικό για έναν XML parser να αναλύσει ένα DTD και να πιστοποιήσει ότι κάθε συστατικό στο XML έγγραφο ακολουθεί τους κανόνες του DTD
- Αν το μόνο που απαιτείται είναι ο εντοπισμός συστατικών και η εξόρυξη πληροφορίας συνίσταται η χρήση non-validating

Figure as from "The XML Companion" - Neil Bradley

Δομή ενός XML

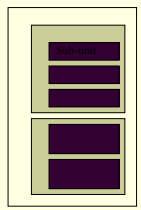
- Λογική δομή
  - Elements
- Φυσική δομή
  - Entities

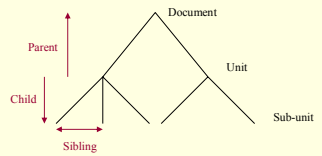




Ιεραρχία XML

Η XML μπορεί να περιγράφει μια δενδρική ιεραρχία





Όλα τα συστατικά πρέπει να είναι ενθυλακωμένα

Parsing XML

- Δυο καθιερωμένα API
  - SAX (Simple API για XML)
    - Ορισμός handlers που περιέχουν μεθόδους κατά τη διάρκεια διάσχισης (parsed) της XML
  - DOM (Document Object Model)
    - Ορίζει ένα λογικό δέντρο που αναπαριστά την parsed XML δομή
- Εφαρμογές που δε χρειάζονται πολύπλοκη διαχείριση μπορούν να χρησιμοποιούν SAX
- Εφαρμογές που χρειάζονται δομική διαχείριση πολλών XML «πραγμάτων» (tokens) να χρησιμοποιούν DOM

DOM

- Document Object Model
- Σύνολο από interfaces για εφαρμογές που αποθηκεύουν ένα αρχείο XML στη μνήμη ως μια δενδρική δομή
- Το αφαιρετικό API επιτρέπει κατασκευή, πρόσβαση διαχείριση και επαναδόμηση της δομής και του περιεχομένου XML και HTML εγγράφων

Πλεονεκτήματα του DOM

- Διασχίζοντας ένα έγγραφο XML με έναν DOM parser, επιστρέφεται μια δομή που περιέχει όλα τα συστατικά του εγγράφου
- Το DOM παρέχει μια ποικιλία συναρτήσεων για την εξέταση της δομής και των περιεχομένων του εγγράφου

DOM αντί SAX

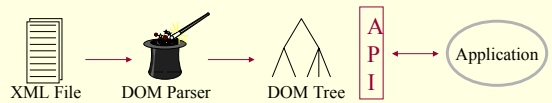
- Αν το έγγραφο είναι πολύ μεγάλο και χρειάζονται μόνο μερικά elements - επέλεξε SAX
- Αν πρέπει να επεξεργαστούν πολλά συστατικά και να εκτελεστούν διαδικασίες πάνω στο XML - επέλεξε DOM
- Αν πρέπει να ανοίξεις το έγγραφο XML πολλές φορές- επέλεξε DOM

DOM Standard

- DOM 1.0 standard από τη www.w3.org
- Αντικειμενοστραφής προσέγγιση
- Αποτελείται από έναν μεγάλο αριθμό διεπαφών
  - org.w3c.dom.*
- Η κεντρική κλάση είναι: '**Document**' (DOM tree)
- Το Standard δεν περιλαμβάνει
  - Παραγωγή στην έξοδο XML format

Δημιουργώντας ένα DOM δέντρο

- Μια υλοποίηση DOM έχει μια μέθοδο ανάθεσης ενός XML αρχείου σε ένα factory object το οποίο θα επιστρέψει ένα Document object που αναπαριστά το συστατικό-ρίζα όλου του εγγράφου
- Το επόμενο βήμα είναι η χρήση του DOM standard interface για αλληλεπίδραση με την XML δομή



Δημιουργώντας ένα DOM δέντρο (2)

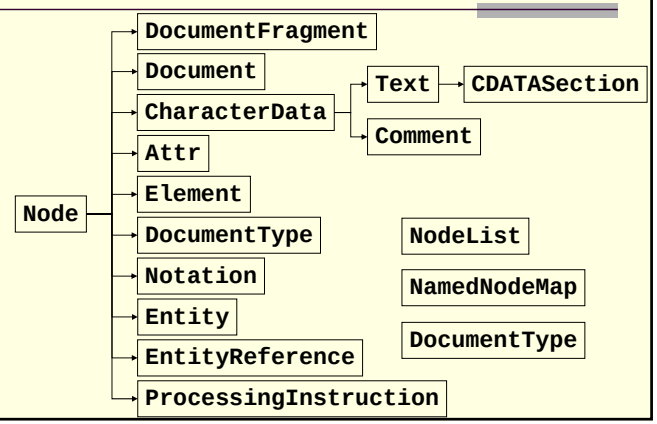
```
import org.w3c.dom.*;          //DOM interfaces
import com.sun.xml.tree.*;     //Using Sun classes
import org.xml.sax.*;          //Need SAX classes

public class myClass {
…
Document myDoc; //Document object
try {
 //if 'true' -> validate
 myDoc =
  XmlDocument.createXmlDocument("file:/doc.xml", true);
} catch (IOException  err) {…}
  catch (SAXException err) {…}
  catch (DOMException err) {…}

//If no exceptions, should have a 'Document' object
```

Figure as from "The XML Companion" - Neil Bradley

DOM Interfaces και Classes



DOM Interfaces

- Το DOM ορίζει πολλά interfaces
  - **`Node`** Ο βασικός τύπος δεδομένων για το DOM
  - **`Element`** Αντιστοιχεί στο συστατικό
  - **`Attr`** Αντιστοιχεί στο attribute του συστατικού
  - **`Text`** Το περιεχόμενο ενός συστατικού ή attribute
  - **`Document`** Αντιστοιχεί στο XML έγγραφο. Συχνά, το αντικείμενο Document αναφέρεται ως DOM δέντρο

Node Interface

- Βασικό αντικείμενο του DOM (κόμβος-ρίζα στο δέντρο)
- Ενας κόμβος-Node μπορεί να είναι:

```
Elements              Entity declarations
Attributes            Entity references
Text                  Notation declarations
Comments              Entire documents
CDATA sections        Processing instructions
```

- Node συλλογές-collections
  - **`NodeList`**, **`NamedNodeMap`**, **`DocumentFragment`**

Node μέθοδοι

- Τρεις κατηγορίες μεθόδων
  - Χαρακτηριστικά Node
    - name, type, value
  - Τοποθεσία και πρόσβαση σε συγγενείς
    - parents, siblings, children, ancestors, descendants
  - Τροποποίηση Node
    - Edit, delete, re-arrange child nodes

Node μέθοδοι (2)

```
short       getNodeType();
String      getNodeName();
String      getNodeValue()              throws DOMException;
void        setNodeValue(String value) throws DOMException;
boolean     hasChildNodes();
NamedNodeMap getAttributes();
Document    getOwnerDocument();
```

Node Types - getNodeType()

```
ELEMENT_NODE           = 1  PROCESSING_INSTRUCTION_NODE =  7
ATTRIBUTE_NODE         = 2  COMMENT_NODE                =  8
TEXT_NODE              = 3  DOCUMENT_NODE               =  9
CDATA_SECTION_NODE     = 4  DOCUMENT_TYPE_NODE          = 10
ENTITY_REFERENCE_NODE = 5  DOCUMENT_FRAGMENT_NODE      = 11
ENTITY_NODE            = 6  NOTATION_NODE               = 12
```

```
if (myNode.getNodeType() == Node.ELEMENT_NODE) {
  //process node
  …
}
```

Ονόματα κόμβων και τιμές

- Κάθε κόμβος-node έχει όνομα και πιθανώς τιμή
- Το όνομα δεν είναι unique identifier (μόνο τοποθεσία)

Type	Interface Name	Name	Value
ATTRIBUTE_NODE	Attr	*Attribute name*	*Attribute value*
DOCUMENT_NODE	Document	#document	*NULL*
DOCUMENT_FRAGMENT_NODE	DocumentFragment	#document-fragment	*NULL*
DOCUMENT_TYPE_NODE	DocumentType	*DOCTYPE name*	*NULL*
CDATA_SECTION_NODE	CDATASection	#cdata-section	*CDATA content*
COMMENT_NODE	Comment	*Entity name*	*Content string*
ELEMENT_NODE	Element	*Tag name*	*NULL*
ENTITY_NODE	Entity	*Entity name*	*NULL*
ENTITY_REFERENCE_NODE	EntityReference	*Entity name*	*NULL*
NOTATION_NODE	Notation	*Notation name*	*NULL*
PROCESSING_INSTRUCTION_NODE	ProcessingInstruction	*Target string*	*Content string*
TEXT_NODE	Text	#text	*Text string*

Table as from "The XML Companion" - Neil Bradley

Child Nodes

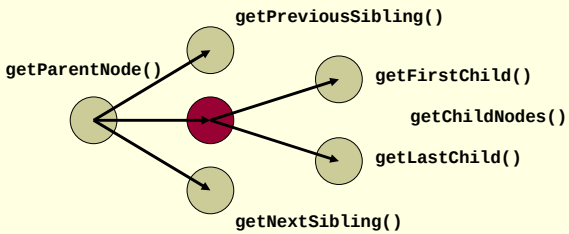
- Οι περισσότεροι κόμβοι δεν μπορούν να έχουν παιδιά, εκτός
  - **Document, DocumentFragment, Element**
- Ελεγχος της παρουσίας παιδιών
  - **if (myNode.hasChildNodes()) {**
    //process children of myNode
    …
    }

Node πλοήγηση

- Κάθε κόμβος έχει συγκεκριμένη θέση στο δέντρο
- Το Node interface ορίζει μεθόδους για να βρει γειτονικούς κόμβους
  - **Node getFirstChild();**
  - **Node getLastChild();**
  - **Node getNextSibling();**
  - **Node getPreviousSibling();**
  - **Node getParentNode();**
  - **NodeList getChildNodes();**

Figure as from "The XML Companion" - Neil Bradley

Node πλοήγηση (2)




```
Node parent = myNode.getParentNode();
if (myNode.hasChildren()) {
  NodeList children = myNode.getChildNodes();
}
```

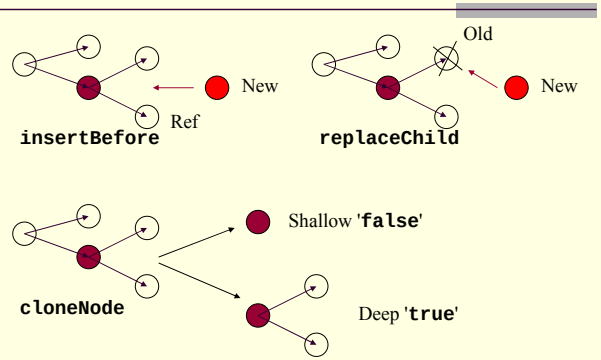
Διαχείριση Node

- Τα παιδιά ενός κόμβου στο δέντρο DOM μπορούν να υποστούν επεξεργασία- added, edited, deleted, moved, copied, κλπ.

```
Node removeChild(Node old)            throws DOMException;
Node insertBefore(Node new, Node ref) throws DOMException;
Node appendChild(Node new)            throws DOMException;
Node replaceChild(Node new, Node old) throws DOMException;
Node cloneNode(boolean deep);
```

Figure as from "The XML Companion" - Neil Bradley

Διαχείριση Node (2)



Document::Node Interface

- Αναφέρεται σε όλο το έγγραφο XML (ρίζα δέντρου)
- Μέθοδοι

```
//Information from DOCTYPE - See 'DocumentType'
DocumentType      getDocumentType();

//Information about capabilities of DOM implementation
DOMImplementation getImplementation();

//Returns reference to root node element
Element           getDocumentElement();

//Searches for all occurrences of 'tagName' in nodes
NodeList          getElementsByName(String tagName);
```

Document::Node Interface (2)

- Μέθοδοι για δημιουργία κόμβων

```
Element createElement(String tagName) throws DOMException;

DocumentFragment createDocumentFragment();

Text createTextNode(String data);

Comment createComment(String data);

CDATASection createCDATASection(String data) throws
                                              DOMException;
ProcessingInstruction createProcessingInstruction(
       String target, String data) throws DOMException;
Attr createAttribute(String name) throws DOMException;

EntityReference createEntityReference(String name)
                                        throws DOMException;
```

DocumentType::Node Interface

- Πληροφορίες για το ενθυλακωμένο έγγραφο DTD
- Το DOM 1.0 δεν επιτρέπει επεξεργασία του κόμβου

```
//Returns name of document
String              getName();

//Returns general entities declared in DTD
NamedNodeList       getEntities();

//Returns notations declared in DTD
NamedNodeList       getNotations();
```

Element::Node Interface

- Δυο κατηγορίες μεθόδων
  - Γενικές μέθοδοι element

```
String   getTagName();
NodeList getElementsByTagName();
void     normalize();
```

  - Διαχείριση Attributes

```
String   getAttribute(String name);
void     setAttribute(String name, String value)
                                  throws DOMException;
void     removeAttribute(String name)
                                  throws DOMException;
Attr     getAttributeNode (String name);
void     setAttributeNode(Attr new)
                                  throws DOMException;
void     removeAttributeNode(Attr old)
                                  throws DOMException;
```

Element::Node Interface (2)

- Προφανώς, μόνο αντικείμενα Element έχουν attributes αλλά οι μέθοδοι για attribute του **`Element`** είναι απλοϊκές
  - Πρέπει να ξέρεις το όνομα του attribute
  - Δεν μπορείς να ξεχωρίσεις μεταξύ της default τιμής που υπάρχει στο DTD και σε αυτή που υπάρχει στο αρχείο XML
- Χρησιμοποίησε τη μέθοδο **`getAttributes()`** του **`Node`**
  - Επιστρέφει αντικείμενα **`Attr`** σε μορφή **`NamedNodeMap`**

Attr::Node Interface

- Interface σε αντικείμενα που έχουν δεδομένα attribute

```
//Get name of attribute
String getName();

//Get value of attribute
String getValue();

//Change value of attribute
void setValue(String value);

//if 'true' - attribute defined in element, else in DTD
boolean getSpecified();
```

Attr::Node Interface (2)

- **`parentNode`**, **`previousSibling`** και **`nextSibling`** έχουν null τιμή για το Attr αντικείμενο

```
//Create the empty Attribute node
Attr newAttr = myDoc.createAttribute("status");

//Set the value of the attribute
newAttr.setValue("secret");

//Attach the attribute to an element
myElement.setAttributeNode(newAttr);
```

CharacterData::Node Interface

- Χρήσιμες γενικές μέθοδοι για επεξεργασία κειμένου
- Δε χρησιμοποιείται άμεσα
  - sub-classed σε Text και Comment Node types

```
String getData()                 throws DOMException;
void   setData(String data)      throws DOMException;
int    getLength();
void   appendData(String data)   throws DOMException;
String substringData(int offset, int length)
                                 throws DOMException;
void   insertData(int offset, String data)
                                 throws DOMException;
void   deleteData(int offser, int length)
                                 throws DOMException;
void   replaceData(int offset, int length, String data)
                                 throws DOMException;
```

Text:: CharacterData Interface

- Αναφέρεται σε περιεχόμενο τύπου «κείμενο» σε **`Element`** ή **`Attr`**
  - Συνήθως παιδί αυτών των κόμβων
- Μια μόνο μέθοδος προστίθεται στο CharacterData interface
  - `Text splitText(int offset) throws DOMException`
- Εκτελώντας **`normalize()`** σε ένα **`Element`** συγχωνεύονται τα αντικείμενα **`Text`** του

CDATASection::Text Interface

- Αναφέρεται σε κείμενο (**CDATA)** που δε θέλουμε να αναγνωριστεί ως σημείωση (το μόνο που αναγνωρίζεται είναι το "**]]>**" που τερματίζει την περιοχή CDATA)
- Το **DOMString** attribute του κόμβου **Text** έχει το κείμενο του CDATA
- Δεν προστίθενται μέθοδοι στο **CharacterData**
- Μέθοδος κατασκευής (Factory method) στο **Document**
  - `CDATASection newCDATA = myDoc.createDATASection("press <<<ENTER>>>");`

Comment::Text Interface

- Αναφέρεται στα σχόλια
- όλοι οι χαρακτήρες μεταξύ '**<!--**' και '**-->**'
- Δεν προστίθενται μέθοδοι στο **CharacterData**
- Factory method in **Document** for creation
  - `Comment newComment = myDoc.createComment(" my comment "); //Note spaces`

ProcessingInstruction::Node Interface

- Αναφέρεται σε δηλώσεις processing instruction
  - Το όνομα του κόμβου είναι η pi
  - Η τιμή του κόμβου είναι το κείμενο μεταξύ του ονόματος της pi και του '?>'

```
//Get the content of the processing instruction
String getData()
//Set the content of the processing instruction
void   setData(String data)
//The target of this processing instruction
String    getTarget();
```

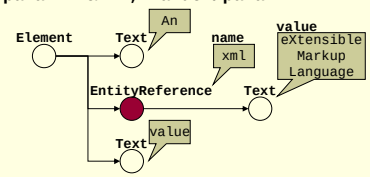
- Μέθοδος κατασκευής (Factory method) στο **Document**
  - `ProcessingInstruction newPI = myDoc.createProcessingInstruction("ACME", "page-break");`

Figure as from "The XML Companion" - Neil Bradley

EntityReference::Node Interface

- Το DOM περιέχει interfaces για διαχείριση entities και entity references

```
<!ENTITY xml "eXtensible Markup Language">
<para>An &xml; value</para>
```



NodeList Interface

- Εχει τη συλλογή ταξινομημένων **Node** αντικειμένων
- 2 μέθοδοι

```
//Find number of Nodes in NodeList
int  getLength();

//Return the i-th Node
Node item(int index);
-----------------------------------------------
Node child;
NodeList children = element.getChildNodes()'
for (int i = 0; i < children.getLength(); i++) {
  child = children.item(i);
  if (child.getNodeType() == Node.ELEMENT_NODE) {
    System.out.println(child.getNodeName());
  }
}
```

NamedNodeMap Interface

Εχει τη συλλογή μη ταξινομημένων **Node** αντικειμένων
Π.χ. **Attribute**, **Entity**

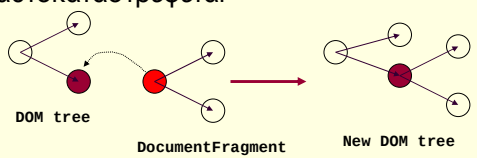
- Τα μοναδικά ονόματα είναι απαραίτητα καθώς οι κόμβοι προσπελαύνονται με το όνομά τους

```
NamedNodeMap myAttributes = myElement.getAttributes();
NamedNodeMap myEntities   = myDocument.getEntities();
-----------------------------------------------------
int getLength();
Node item(int index);
Node getNamedItem(String name);
Node setNamedItem(Node node) throws DOMException;//Node!
Node removeNamedItem(String name) throws DOMException;
```

Figure as from "The XML Companion" - Neil Bradley

DocumentFragment::Node Interface

- Ενα τμήμα ενός κειμένου μπορεί να αποθηκευτεί προσωρινά σε κόμβο **DocumentFragment**
  - Π.χ. Για 'cut-n-paste'
- Οταν προστίθεται σε άλλο κόμβο, αυτοκαταστρέφεται



DOMImplementation Interface

- Interface για τον καθορισμό βαθμού υποστήριξης από τον DOM parser
  - `hasFeature(String feature, String version);`
  - ```
    if (theParser.hasFeature("XML", "1.0") {
      //XML is supported
      …
    }
    ```
    ```

DOM αντί XSL

- Αν απαιτείται πολύπλοκη ταξινόμηση ή αναδόμηση, προτιμάται το DOM
- Η XSL είναι πιο lightweight