Service Oriented Architectures

XML – DOM + PHP

Dario Bonino

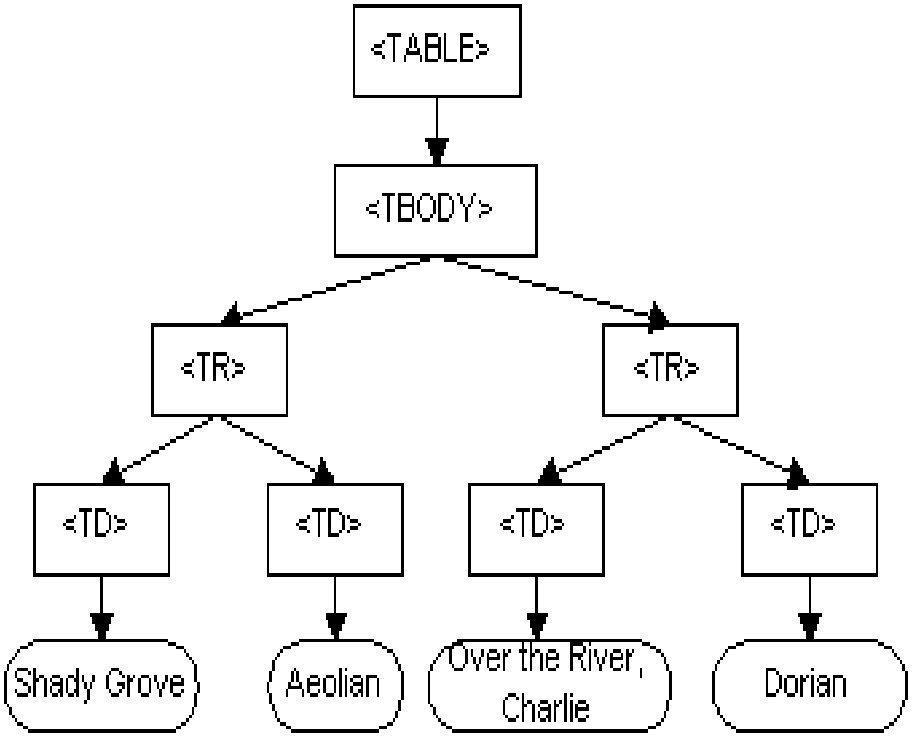
Dipartimento di Automatica e Informatica

Politecnico di Torino

Cos'è il DOM

- “The DOM is a programming API for documents. It closely resembles the structure of the documents it models”.
-

```
<TABLE>
  <TBODY>
      <TR>
      <TD>Shady Grove</TD>
      <TD>Aeolian</TD>
      </TR>
      <TR>
      <TD>Over the River,
Charlie</TD>
      <TD>Dorian</TD>
      </TR>
   </TBODY>
</TABLE>
```



Caratteristiche del DOM

- Struttura molto simile ad un albero, o per essere più precisi ad una foresta
- Il DOM non impone che la struttura dei documenti sia un albero o una foresta
- E' un modello “strutturale” che rappresenta un documento
- E' strutturalmente isomorfo:
  - Se due implementazioni di DOM vengono usate per rappresentare lo stesso documento, esse generano lo stesso modello strutturale, cioè gli stessi oggetti e relazioni

DOM – Document Object Model (1/2)

- “Object Model” viene inteso nell'ottica del paradigma di programmazione ad oggetti
- I documenti sono modellati come insieme di oggetti
- Il modello include
  - la struttura
  - il comportamento del documento
  - il comportamento degli oggetti che compongono la struttura

DOM – Document Object Model (2/2)

- Il DOM è composto da:
  - le interfacce e gli oggetti utilizzati per rappresentare un documento
  - la semantica di interfacce ed oggetti (comportamenti ed attributi)
  - le relazioni e collaborazioni tra oggetti e interfacce

DOM – specifiche (1/2)

- E' composto di due parti principali
  - DOM Core
  - DOM HTML
- DOM Core
  - rappresenta le funzionalità relative ai documenti XML
  - è la base del DOM HTML

DOM – specifiche (2/2)

- Ogni implementazione di DOM deve rispettare le specifiche del DOM Core
- Ogni implementazione di DOM deve supportare almeno una delle interfacce DOM HTML e una delle interfacce DOM estese (XML)
- Specifica
  - http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/cover.html

Cosa non è il DOM (1/3)

- Non è e non supporta completamente il “Dynamic HTML” (DHTML)
- Non è una specifica binaria:
  - i programmi utilizzanti implementazioni del DOM saranno “compatibili” a livello di codice sorgente, su diverse piattaforme.
  - non è garantita la compatibilità a livello di file binari

Cosa non è il DOM (2/3)

- Non è un modo di dare persistenza ad oggetti software
- Non è un set di strutture dati ma di interfacce
  - non assume che le relazioni logiche tra interfacce corrispondano ad una implementazione fisica

Cosa non è il DOM (3/3)

- Il DOM non definisce la semantica di XML o HTML (definita dalle relative specifiche W3C)
- Il DOM non è un competitore dello standard COM (Common Object Model).
  - è un set di interfacce e oggetti progettati per rappresentare documenti XML e HTML

Elementi del DOM (1/2)

- Il DOM rappresenta un documento come gerarchia di elementi `Node`
- Ciascun Node può essere ulteriormente specializzato

Document -- Element (maximum of one), ProcessingInstruction, Comment, DocumentType

DocumentFragment -- Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference

DocumentType -- no children

EntityReference -- Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference

Element -- Element, Text, Comment, ProcessingInstruction, CDATASection, EntityReference

Elementi del DOM (2/2)

Attr -- Text, EntityReference

ProcessingInstruction -- no children

Comment -- no children

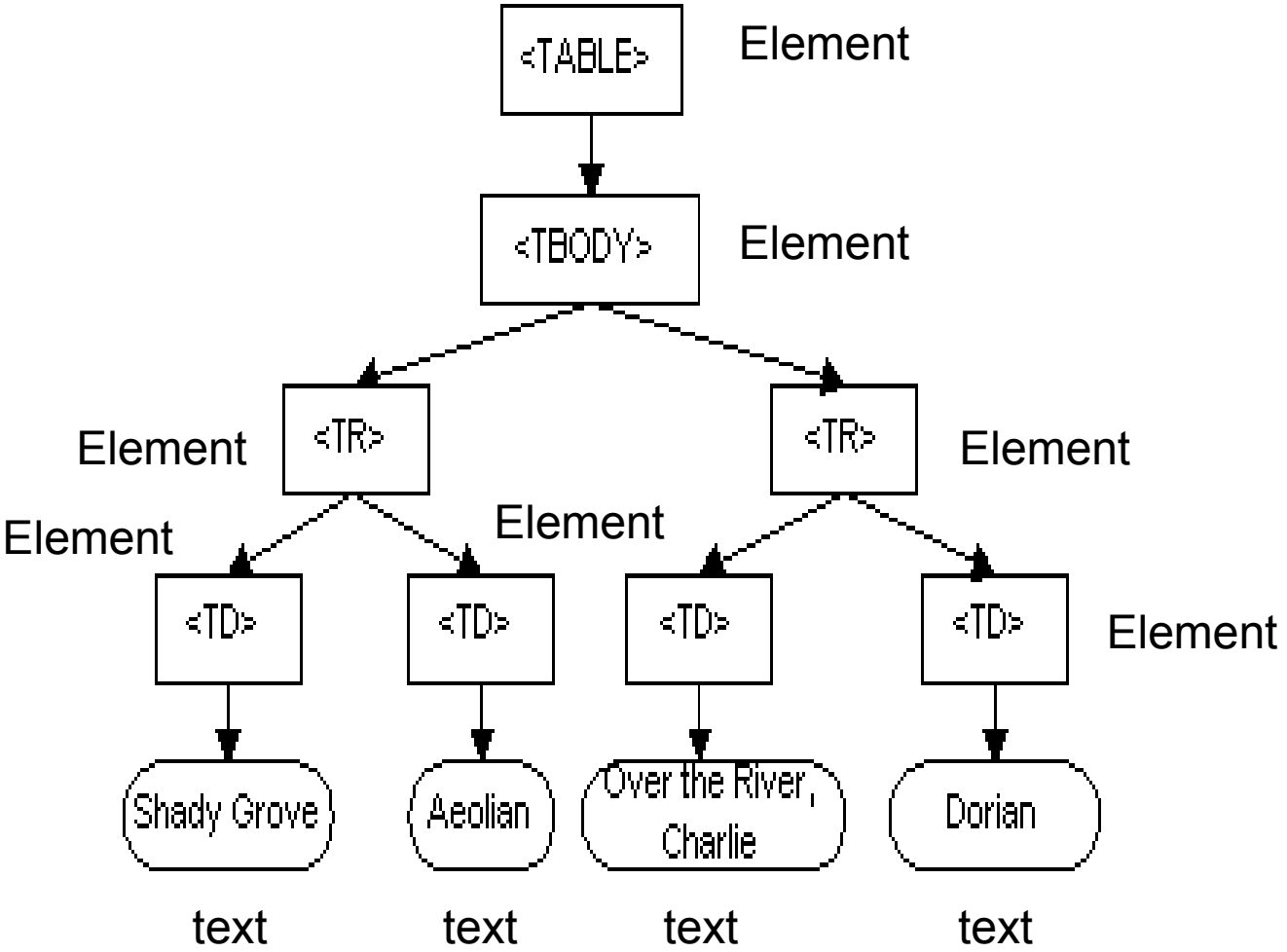
Text -- no children

CDATASection -- no children

Entity -- Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference

Notation -- no children

Esempio


<TABLE>
Element
<TBODY>
Element
Element
<TR>
<TR>
Element
Element
Element
<TD>
<TD>
<TD>
<TD>
Element
Shady Grove
Aeolian
Over the River, Charlie
Dorian
text
text
text
text

Interfacce / metodi (1/3)

- Per ogni elemento il DOM definisce una serie di interfacce (cioè metodi e attributi) che possono essere utilizzati per la manipolazione del documento

Interfacce / metodi (2/3)

```
interface Document : Node {
  readonly attribute  DocumentType         doctype;
  readonly attribute  DOMImplementation    implementation;
  readonly attribute  Element              documentElement;
  Element                  createElement(in DOMString tagName)
                                        raises(DOMException);
  DocumentFragment         createDocumentFragment();
  Text                     createTextNode(in DOMString data);
  Comment                  createComment(in DOMString data);
  CDATASection             createCDATASection(in DOMString data)
                                        raises(DOMException);
  ProcessingInstruction    createProcessingInstruction(in DOMString
target,
                                                    in DOMString data)

raises(DOMException);
  Attr                     createAttribute(in DOMString name)
                                        raises(DOMException);
  EntityReference          createEntityReference(in DOMString name)
                                        raises(DOMException);
  NodeList                 getElementsByTagName(in DOMString tagname);
};
interface NodeList {
  Node                     item(in unsigned long index);
  readonly attribute  unsigned long        length;
};
```

Interfacce / metodi (3/3)

```
interface Element : Node {
  readonly attribute  DOMString           tagName;
  DOMString                   getAttribute(in DOMString name);
  void                        setAttribute(in DOMString name,
                                           in DOMString value)
                                           raises(DOMException);
  void                        removeAttribute(in DOMString name)
                                            raises(DOMException);
  Attr                        getAttributeNode(in DOMString name);
  Attr                        setAttributeNode(in Attr newAttr)
                                             raises(DOMException);
  Attr                        removeAttributeNode(in Attr oldAttr)

raises(DOMException);
  NodeList                    getElementsByTagName(in DOMString name);
  void                        normalize();
};

interface Attr : Node {
  readonly attribute  DOMString           name;
  readonly attribute  boolean             specified;
           attribute  DOMString           value;
};
```

Esempio – valore di un attributo (1/3)

- Dato un elemento “professore” si vuole leggere il valore del suo attributo “name”

```
<corso>
   <nome>01KTF</nome>
   <professore name=  Fulvio   surname=  Corno  />
</corso>
```

Esempio – valore di un attributo (2/3)

- La prima operazione necessaria è ottenere l'oggetto Element corrispondente a “professore”
  - `NodeList list = Document.getElementsByTagName('professore')`
- poiché esiste un solo elemento professore la NodeList “list” conterrà un solo elemento (Node)
  - `Element prof = list.item(0)`

Esempio – valore di un attributo (3/3)

- per leggere il valore dell'attributo name
  - `DOMString profName = prof.getAttribute('name');`
- in versione compatta
  - `Document.getElementsByTagName('professore').item(0).getAttribute('name')`

DOM in PHP

- PHP fornisce una implementazione del DOM
  - http://www.php.net/manual/it/ref.dom.php

W3C DOM	PHP DOM
Document	DOMDocument
Node	DOMNode
Element	DOMElement
NodeList	DOMNodeList
Document.getElementsByTagName()	DOMDocument->getElementsByTagName()
NodelList.item()	DOMNodeList->item()

- esempio
  - $list = DOMDocument::getElementsByTagName('professore');
  - $name = $list.intem(0).getAttribute('name');

DOM in Javascript

- Javascript fornisce una implementazione del DOM

W3C DOM	JS DOM
Document	document
Node	node
Element	element
NodeList	node list
Document.getElementsByTagName()	document.getElementsByTagName()
NodeList.item()	nodelist[index]

- esempio
  - var list = document.getElementsByTagName('professore');
  - var name = list[0].getAttribute('name');