
Elaborazione di documenti XML

Corso di **Applicazioni Telematiche**

A.A. 2006-07 – Lezione n.13

Prof. Roberto Canonico

Università degli Studi di Napoli Federico II

Facoltà di Ingegneria

XML e i linguaggi di programmazione

- XML non è altro che un formato passivo utilizzato per codificare dati
 - I programmatori devono incorporare nelle proprie applicazioni le tecnologie necessarie per generare e interpretare documenti XML
-

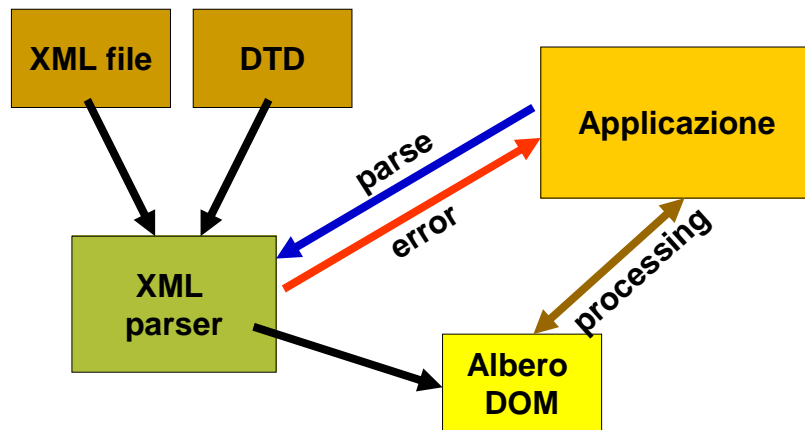
API for XML

- La struttura grammaticale semplice di XML ha dato luogo alla creazione di diverse librerie per generare e interpretare documenti XML
 - Esistono due diversi approcci alla elaborazione di documenti XML:
 - SAX (Simple API for XML)
 - DOM (Document Object Model)
-

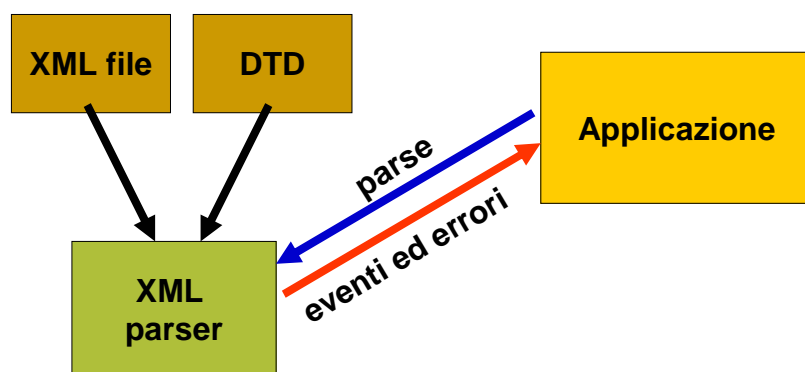
DOM

- Il *Document Object Model* definisce una gerarchia di oggetti in grado di descrivere un documento XML
 - Il parser integrato in DOM analizza il documento XML e genera in memoria la struttura DOM
 - Il programmatore deve quindi solo creare il codice per cercare le informazioni nella struttura DOM
-

Utilizzo di DOM in un'applicazione



Utilizzo di SAX in un'applicazione



Elaborazione del documento

- Dopo il parsing
 - Se si usa un parser DOM
 - Il parser ritorna un oggetto di tipo *Document*
 - Il metodo *GetDocumentElement* restituisce l'elemento root
- Durante il parsing
 - Se si usa un parser SAX
 - Il parser genera eventi durante il parsing
 - L'applicazione elabora i dati mediante appositi metodi *handler*

SAX

- Le *Simple API for XML* definiscono un modello di interpretazione guidato dagli eventi
- Un parser SAX utilizza un meccanismo di callback per inviare delle notifiche durante l'interpretazione dei vari elementi XML
- Il parser quindi non mantiene uno stato, e non crea nessuna struttura in memoria; questa caratteristica lo rende particolarmente adatto per l'utilizzo con documenti molto grandi

Invocazione del parser DOM

```
DOMParser parser = new DOMParser() ;  
parser.setErrorHandler( new MyErrorHandler() ) ;  
  
try  
{  
    parser.parse( fn ) ;  
}  
catch( Exception e )  
{  
    System.out.println( "Exception invoking parser" ) ;  
}  
Document doc = parser.getDocument() ;  
  
if ( doc != null )  
{  
    processTheDocument( doc ) ;  
}
```

Invocazione del parser SAX

```
SAXParser parser = new SAXParser() ;  
parser.setDocumentHandler( this ) ;  
parser.setErrorHandler( new MyErrorHandler() ) ;  
  
try  
{  
    parser.parse( fn ) ;  
}  
catch( Exception e )  
{  
    System.out.println( "Exception invoking parser", 0 ) ;  
}
```

Gestione degli eventi SAX

```
public void startDocument()
{
}
public void endDocument()
{
}
public void startElement( String name, AttributeList attrs )
{
}
public void endElement( String name )
{
}
```

SAX e DOM

- La W3C ha definito soltanto l'interfaccia delle due librerie SAX e COM
 - Questo ha permesso l'implementazione per moltissimi linguaggi di programmazione:
 - Java
 - C / C++
 - Perl
 - Python
 - Tcl
 - ...
-

DOM vs. SAX

- **DOM**
 - Document Object Model (tree structure)
 - Good if you intend to edit the document
 - Good if you plan to process multiple times.
 - Avoids re-parsing each time.
 - Avoids building your own internal tree.
- **SAX**
 - Event based model.
 - Good if you don't need to edit in memory.
 - Good if memory is limited.
 - Good if only parts of document are required.

Domande?

