



4

Il metalinguaggio della struttura e della semantica: XML

Unimc - BBCC 12/13 - II semestre

Laboratorio di informatica
avanzata per i beni culturali

prof. Pierluigi Feliciati
pierluigi.feliciati@unimc.it

Dai 4 aspetti della pagina web...

- Come ricorderete, abbiamo accennato in apertura del modulo 2, che nella pagina web possiamo individuare 4 aspetti: CONTENUTO, STRUTTURA, PRESENTAZIONE e COMPORTAMENTO
- Con l'esercitazione HTML abbiamo sperimentato un linguaggio di marcatura che ci permettesse di coprire tutti insieme i primi tre aspetti, pur sapendo che la presentazione deve essere gestita a parte, con i fogli di stile
- Ora ci concentreremo su un linguaggio DI MARCATURA che consente SOLO di STRUTTURARE CONTENUTI (non solo per il Web) e non serve a fare presentazioni: **XML**

XML

oltre HTML: XML

XML, acronimo di **eXtensible Markup Language** è un metalinguaggio creato e gestito dal World Wide Web Consortium (W3C). È una semplificazione e adattamento dell'**SGML**, da cui è nato nel 1998, e permette di definire la grammatica di diversi linguaggi specifici derivati.

Rispetto all'HTML, l'XML ha uno scopo ben diverso: **XML è un linguaggio utile allo scambio dei dati, quindi di back-office e non di front-office, o di (rap)presentazione che dir si voglia.** L'XML è oggi molto utilizzato anche per l'esportazione di dati tra DBMS.

La sua versione 1.0 risale al 1998 e la versione italiana del linguaggio è in <http://www.xml.it:23456/XML/REC-xml-19980210-it.html>, la seconda specifica del 2000, la terza e la versione 1.1 (compatibile con Unicode 4) del 2004 <http://www.w3.org/TR/REC-xml>

XML

Il World Wide Web Consortium (W3C), in seguito alla cd. *guerra dei browser* (nei tardi anni '90 Microsoft e Netscape introducevano, con ogni nuova versione del proprio browser, un'estensione proprietaria all'HTML ufficiale), fu costretto a seguire le individuali estensioni a HTML. Il W3C dovette scegliere **quali caratteristiche standardizzare e quali lasciare fuori** dalle specifiche ufficiali dell'HTML. Fu in questo contesto che iniziò a delinearsi la necessità di **un linguaggio di markup che desse maggiore libertà nella definizione dei tag**, pur rimanendo in uno standard.

Il "**progetto XML**" suscitò un così forte interesse a tal punto che si creò un XML Working Group, composto da esperti mondiali delle tecnologie SGML, ed una commissione, XML Editorial Review Board, deputata alla redazione delle specifiche del progetto.

Nel febbraio del 1998 le specifiche divennero una raccomandazione ufficiale con il nome di *Extensible Mark-up Language, versione 1.0*. Ben presto ci si accorse che XML **non era solo limitato al contesto web**, ma era qualcosa di più: uno strumento che permetteva di essere utilizzato nei più diversi contesti, dalla definizione della struttura di documenti, allo scambio delle informazioni tra sistemi diversi, dalla rappresentazione di immagini alla definizione di formati di dati. (da Wikipedia)

XML

La struttura vera e propria è composta dai **tag** (**elementi, marcatori**) **creati dallo sviluppatore**, che hanno essenzialmente due caratteristiche:

- devono essere comprensibili in funzione dello scopo (onde evitare di non capirci nulla ad una seconda lettura e per facilitare la comprensione agli altri utenti);
- devono rispettare delle regole, come la differenza tra maiuscolo e minuscolo, non possono iniziare con numeri o caratteri speciali e non possono contenere spazi.

Ogni **record**, se si vuol pensare ad XML come ad un **database**, viene chiamato **nodo** ed ogni tag può essere corredato da **attributi**, come per HTML.

XML

Ad esempio, la struttura precedente può essere modificata come segue:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<utenti>
    <utente nome="Luca" cognome="Ruggiero" />
    <utente nome="Max" cognome="Rossi" />
</utenti>
</xml>
```

I tag che non hanno l'omonimo tag di chiusura vanno chiusi con uno spazio più uno **slash (/)** finale, prima della fine del tag stesso.

XML

La struttura di XML dunque **non ha interfaccia propria** ma è accessibile da qualsiasi linguaggio di programmazione Web (sia lato server che lato client).

Il rispetto delle sue *semplici regole* ed il fatto che *i dati sono accessibili semplicemente insieme alla loro struttura* a qualsiasi tipo di applicazione, locale o remota, fanno dell'XML **uno standard** che si caratterizza per la sua **portabilità**.

L'accesso ad una struttura XML può avvenire attraverso diversi sistemi, il più comune ma rudimentale dei quali è lo sfruttamento del *filesystem* (la gestione dei file in memoria, il gestione risorse di Windows) del server su cui gira il motore del linguaggio in uso.

XML

La semplicità, adattabilità e piena compatibilità di XML con gli altri linguaggi di marcatura, la quasi totale mancanza di caratteristiche opzionali, la leggibilità umana e la facilità per le macchine di elaborarlo ne hanno decretato il successo.

Lo sviluppo del linguaggio DTD prima, di XSD poi, l'adozione dei *NameSpace* (per la definizione delle strutture dati e il richiamo di diverse regole in uno steso file) e l'uso degli *stylesheet* (per la visualizzazione) hanno sostenuto questo percorso. L'unico difetto – volendo - è la poca economicità in termini di spazio di occupazione, dovuta alla ridondanza di elementi oltre ai dati veri e propri.

Sintassi XML

Le regole sintattiche di Xml sono poche:

- Ogni documento deve presentare **un prologo** per la dichiarazione della specifica Xml che si usa, le eventuali informazioni di codifica e il richiamo esplicito alla DTD in uso

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

Questa prima riga, racchiusa tra i simboli `<? e ?>` si dice *processing instruction* e dalla versione 1.1 di XML è divenuta obbligatoria. Se omessa, il documento diventerà automaticamente 1.0

Sintassi XML

Il contenuto di un documento XML deve corrispondere a una gerarchia di **tag** (o **elementi**) **non sovrapposti**, quindi i tag che si aprono per primi debbono chiudersi per primi.

*Per elemento/tag (marcatore) si intende una coppia
<parola>(apertura) e </parola> (chiusura) con
parola=nome del tag*

```
<DOCUMENTO>  
  <TITOLO>Titolo del documento</TITOLO>  
  <TESTO>  
    <PARAGRAFO>testo del primo paragrafo</PARAGRAFO>  
    <PARAGRAFO>testo del secondo paragrafo</PARAGRAFO>  
  </TESTO>  
</DOCUMENTO>
```

Sintassi XML

- Deve esistere un tag che contenga tutti gli altri, che si dice **tag radice** o *root*.
- Tutti i tag devono **sempre essere aperti e chiusi** e se non contengono testo, per economia, possono essere presentati usando la cd. *chiusura implicita*:

<TAGVUOTO/> equivale a <TAGVUOTO></TAGVUOTO>

- I nomi dei tag sono **a nostro piacere**, rispettando però alcune regole:
 - Devono iniziare con un carattere o con un _ (*underscore*), mai con numeri, possono contenere un numero qualsiasi di lettere, numeri, trattini, punti e underscore
 - Non possono contenere spazi e sono case-sensitive: <TAG> e <tag> sono diversi, i tag di apertura e di chiusura devono essere uguali

Sintassi XML

- Possono essere associati **attributi** agli elementi (tag), rappresentandoli nel tag di apertura, racchiudendone **il valore tra apici o virgolette** e non ripetendoli più di una volta:

```
<PREZZO valuta='euro'>15</PREZZO>
```

```
<PREZZO valuta="euro">15</PREZZO>
```

- Per i nomi degli attributi valgono le stesse regole degli elementi
- Esistono **attributi riservati** che possono essere usati solo rispettandone il significato previsto, ad es.

`xml:lang="it | en | fr ..."` indica la lingua del tag

`xml:space="preserve | default"` indica se gli spazi presenti nel contenuto devono essere mantenuti o considerati superflui (quindi ev. eliminati)

Sintassi XML

- Nel caso sia necessario usare **all'interno del contenuto caratteri speciali** (come <, >, &, " etc.) si dovrà usare il cd. **Entity-name** corrispondente, composto da un &, da un nome e da un punto-e-virgola finale, ad es.

&	corrisponde a	&
<	corrisponde a	<
>	corrisponde a	>
"	corrisponde a	"

- Se si deve inserire nel contenuto una porzione di testo **scritta in un codice diverso** (ad es. HTML, JavaScript), si dovrà usare la tipologia di contenuto **CDATA section**

```
<contenuto>
  <![CDATA[
    <html><body> .... </body></html>
  ]]>
</contenuto>
```

XML: definizione degli schemi

Ci sono diversi modi per **dichiarare uno schema di codifica**, vale a dire la grammatica che stabilisce quali elementi sarà possibile usare all'interno di un documento XML e in che modo tali elementi si potranno relazionare fra di loro.

Il modo più conosciuto per dichiarare uno schema di codifica è attraverso una **Document Type Definition** (o **DTD**), una sintassi ereditata direttamente dal linguaggio SGML, da cui l'XML deriva. La DTD è costituita da una sintassi estremamente semplice, ma con varie limitazioni.

Recentemente a questo metodo si sono aggiunti alcuni linguaggi per scrivere **XML Schema** (o **XSD**). Lo schema più usato è appunto quello del W3C

<http://www.w3.org/TR/xmlschema-0/>.

XML e *namespace*

Il rispetto della sintassi XML e dello schema adottato garantisce **documenti *well-formed*** (ben costruiti), leggibili da qualunque software su qualunque piattaforma hardware. I browser visualizzano i file XML ed evidenziano se ci sono errori.

I **software che leggono i documenti XML** si dicono ***parser*** perché scandiscono riga per riga il codice e lo interpretano.

A volte è necessario poter usare dentro al proprio documento XML alcuni elementi che sono **già stati definiti in un qualche schema** di codifica diverso dal nostro. Lo strumento per andare incontro a questa esigenza è chiamato ***namespace***.

XML e namespace

Un *namespace* è indicato da un prefisso distintivo.

Per usarlo è necessario **dichiararlo** all'interno di un qualsiasi elemento di un documento XML, usando un attributo standard, **xmlns** (abbreviazione di XML NameSpace) seguito da un due punti, dal prefisso distintivo del namespace da adottare e dalla URL dove si trova lo schema cui il namespace fa riferimento.

Se, per esempio, vogliamo usare alcuni elementi definiti dallo **schema Dublin Core** per codificare una citazione bibliografica, basterà usare il **namespace dc** nel modo che segue:

```
<citazione xmlns:dc="http://purl.org/dc/elements/1.1/">  
  <dc:creator>Carlo Dionisotti</dc:creator>  
  <dc:title>Geografia e storia della letteratura italiana  
    </dc:title>  
  <dc:publisher>Einaudi</dc:publisher>  
</citazione>
```

XML e *namespace*

..che è più funzionale, internazionale e *interoperabile* rispetto a questo codice:

```
<citazione>  
  <autore>Carlo Dionisotti</autore>  
  <titolo>Geografia e storia della letteratura  
italiana</titolo>  
  <editore>Einaudi</editore>  
</citazione>
```

perché si adotta **uno standard usato da molti** e che quindi molti (intendendo soprattutto le macchine) possono decodificare facilmente.

Identificazione e descrizione delle risorse digitali

per garantire le funzioni basilari (**trovare, identificare, selezionare, ottenere**) i *metadati* devono rispondere a **requisiti strutturali e formali condivisi**. L'**adozione di standard** in questo senso diventa infatti particolarmente importante:

- per facilitare lo scambio tra soggetti diversi
- perché in ambiente digitale parte dell'elaborazione delle informazioni è svolta da *software*, che non sono sempre abbastanza furbi da sciogliere le differenze sintattiche e logiche...

Identificazione e descrizione delle risorse digitali

Lo standard più usato per esprimere informazioni di base relative alla forma e al contenuto delle risorse digitali (**metadati descrittivi**) per le risorse digitali è il:

Dublin Core Metadata Element Set

sviluppato in ambito statunitense, dall'OCLC (*Online Computer Library Center*) che ha stabilito un vocabolario semantico per descrivere informazioni sulle caratteristiche *core* (**principali**) di un oggetto digitale e categorizzarlo ai fini di una ricerca semplificata da parte dell'utente (*umano o automatico* che sia).

Dublin indica la città dell'Ohio dove il gruppo di lavoro si è riunito per la prima volta, **Core** indica la basilarità del set

Identificazione e descrizione delle risorse digitali

Il set minimo, del dicembre 1996, è costituito da **15 elementi di base** e si è esteso anche a sottoelementi o qualificatori

Lo standard è in via di sviluppo ma il cosiddetto *core* dei 15 elementi della descrizione è rimasto stabile.

Ciascun elemento è definito usando un set di **10 attributi** ricavati dalla norma ISO 11179 (*Specification and standardization of data elements*).

Dublin Core

I 15 elementi di Dublin Core (1-8)

<u>nome</u>	<u>identificatore</u>	<u>definizione</u>
Titolo	DC.Title	Un nome dato alla risorsa
Creatore	DC.Creator	Un'entità che ha la responsabilità principale della produzione del contenuto della risorsa
Soggetto	DC.Subject	L'argomento della risorsa.
Descrizione	DC.Description	Una spiegazione del contenuto della risorsa
Editore	DC.Publisher	Un'entità responsabile della produzione della risorsa, disponibile nella sua forma presente
Autore di contr. subordinato	DC.Contributor	Un'entità responsabile della produzione di un contributo al contenuto della risorsa
Data	DC.Date	Una data associata a un evento del ciclo di vita della risorsa, ad es. nella forma YYYY-MM-DD
Tipo	DC.Type	La natura o il genere del contenuto della risorsa, ad es. la lista dei Dublin Core Types

Dublin Core

I 15 elementi di Dublin Core (9-15)

<u>nome</u>	<u>identificatore</u>	<u>definizione</u>
Formato	DC.Format	La manifestazione fisica o digitale della risorsa (cfr. Internet Media Types [MIME])
Identificatore	DC.Identifier	Un riferimento univoco alla risorsa nell'ambito di un dato contesto (URL, DOI, ISBN...)
Fonte	DC.Source	Un riferimento a una risorsa dalla quale è derivata la risorsa in oggetto
Lingua	DC.Language	La lingua del contenuto intellettuale della risorsa (codice di 2 chr ISO 639 o 2+2 ISO 3166 x paese)
Relazione	DC.Relation	Un riferimento alla risorsa correlata
Copertura	DC.Coverage	L'estensione o scopo del contenuto della risorsa (sua estensione spaziale, temporale o giurisdiz.)
Gestione dei diritti	DC.Rights	Informazione sui diritti esercitati sulla risorsa

Dublin Core in XML

- Fin dal 2003 la *Dublin Core Metadata Initiative* ha prodotto una raccomandazione per l'implementazione di dichiarazioni DC (cioè di metadati descrittivi) usando il linguaggio XML
<http://www.dublincore.org/documents/dc-xml-guidelines/>
- Le raccomandazioni sono queste:
 1. Le applicazioni XML devono essere basate su [XMLSCHEMA] e non su XML DTD
 2. L'uso dei XML Namespaces [XMLNS] deve servire a identificare univocamente gli elementi DC, gli eventuali *refinements* e gli schemi di codifica

Dublin Core in XML

- **Raccomandazione 3:** gli sviluppatori dovrebbero codificare le proprietà di una risorsa come elementi XML e i valori come il contenuto di tali elementi. Il nome dell'elemento XML dovrebbe essere un nome XML completo che associa il nome dell'elemento con il nome appropriato DCMI namespace. Ad esempio, si deve usare

`<dc:title> Dublin Core in XML </ dc: title>`

piuttosto che

`<dc:title value="Dublin Core in XML" />`

Dublin Core in XML

- **Raccomandazione 4:** I nomi delle proprietà per i 15 elementi DC dovrebbero essere tutti espressi in minuscolo.

Ad esempio, utilizzare

```
<dc:title> Dublin Core in XML </ dc: title>
```

piuttosto che

```
<dc:Title> Dublin Core in XML </ dc: title>
```

Dublin Core in XML

- **Raccomandazione 5:** I valori delle proprietà multiple dovrebbero essere codificati ripetendo l'elemento XML che serve a esprimere quella proprietà.
- Per esempio:

```
<dc:title> Primo titolo </ dc: title>
```

```
<dc:title> secondo titolo </ dc: title>
```

Dublin core in XML: esempio

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<metadata
```

```
  xmlns="http://example.org/myapp/"
```

```
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
  xsi:schemaLocation="http://example.org/myapp/
```

```
http://example.org/myapp/schema.xsd"
```

```
  xmlns:dc="http://purl.org/dc/elements/1.1/">
```

<!-- questa parte serve a dichiarare quale schema si adotta e che si userà il namespace di Dublin Core →

```
<dc:title> UKOLN </dc:title>
```

```
<dc:description> UKOLN is a national focus of expertise in digital  
information management. It provides policy, research and awareness services  
to the UK library, information and cultural heritage communities. UKOLN is  
based at the University of Bath. </dc:description>
```

```
<dc:publisher> UKOLN, University of Bath </dc:publisher>
```

```
<dc:identifier> http://www.ukoln.ac.uk/ </dc:identifier>
```

```
...
```

```
</metadata>
```

Dublin core in XML

- **Ora tocca a voi...**
- Scegliete un oggetto digitale di ambito culturale di cui sapete un po' di informazioni (**una foto, un video, un documento digitalizzato..**) e provate a creare i metadati descrittivi scrivendo un file XML che adotti l'XMLNS Dublin Core.
- L'editor è sempre BLOCCO NOTE, e per visualizzarlo – in mancanza di fogli di stile – possiamo usare il browser, che ci evidenzierà soltanto i tag e i contenuti

Verso il WWW semantico: RDF

Qualunque cosa descritta da RDF è detta **risorsa**

Ogni risorsa è identificata da un **URI**.

Il modello di dati RDF è formato da **risorse, proprietà e valori**.

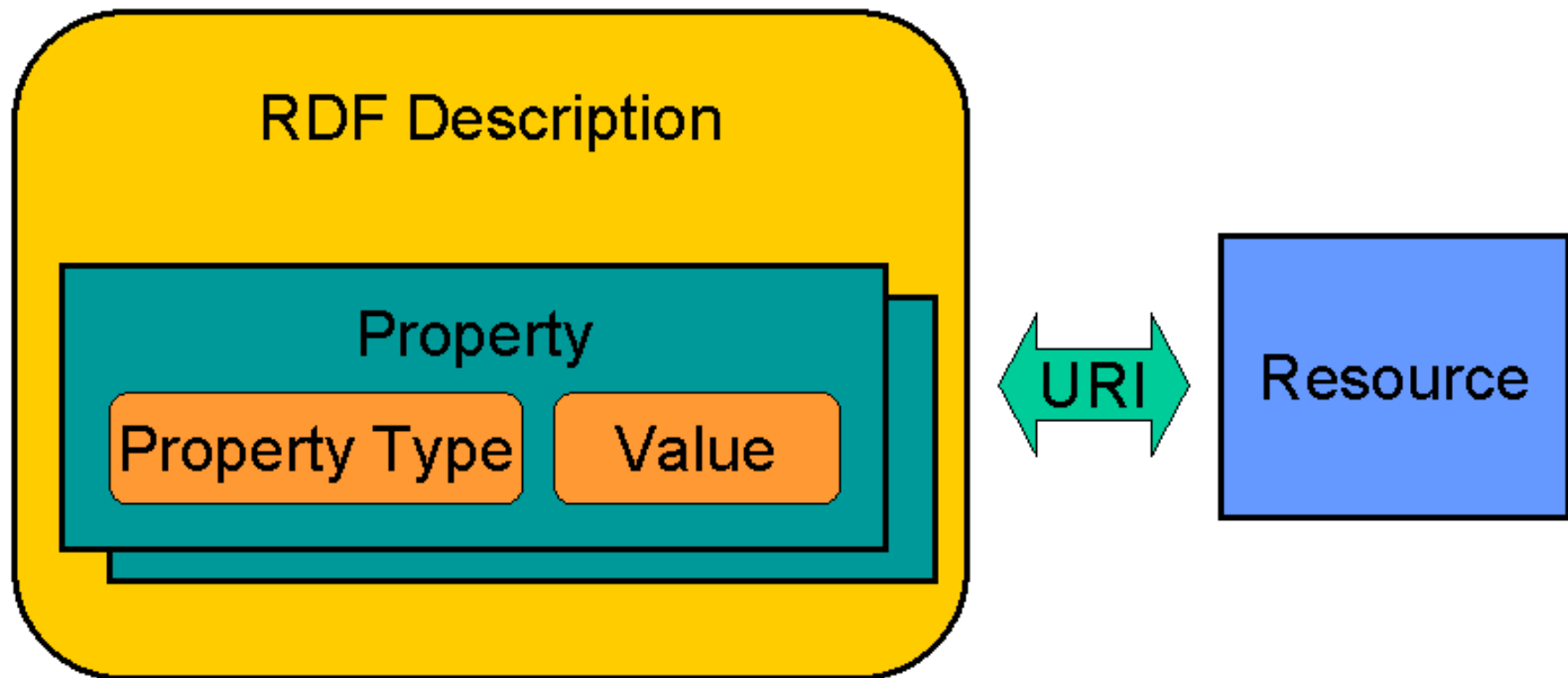
Le **proprietà** sono delle **relazioni che legano tra loro risorse e valori**, e sono anch'esse identificate da URI.

Un **valore**, invece, è **una risorsa o è un dato** primitivo.

L'unità base per rappresentare un'informazione in RDF è una tripletta del tipo: **Soggetto – Predicato – Oggetto** dove il soggetto è una risorsa, il predicato è una proprietà e l'oggetto è un valore.

Verso il WWW semantico: RDF

Una rappresentazione di conoscenza in RDF



Verso il WWW semantico: RDF

Le principali modalità adottabili per esprimere RDF sono:

XML: con metodo classico o abbreviato, più leggibile per l'uomo

N3: si descrive una risorsa e tutte le sue proprietà

"Alessandro_Manzoni" "è autore di"

"I_promessi_sposi" in RDF/XML si esprime così:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:au="http://description.org/schema/">
  <rdf:Description
    about="http://www.book.it/I_promessi_sposi/">
    <au:author>Alessandro_Manzoni</au:author>
  </rdf:Description>
</rdf:RDF>
```

Dublin Core in RDF

Trattandosi sempre di XML, in RDF si possono esprimere le proprietà di un oggetto digitale anche adottando standard descrittivi come Dublin Core.

Vedi: <http://dublincore.org/documents/dcmes-xml/> e un esempio:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF PUBLIC "-//DUBLIN CORE//DCMES DTD
  2002/07/31//EN"
  "http://dublincore.org/documents/2002/07/31/dcmes-xml/dcmes-
  xml-dtd.dtd">
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
  ns#" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description
    rdf:about="http://www.ilrt.bristol.ac.uk/people/cmdjb/">
    <dc:title>Dave Beckett's Home Page</dc:title>
    <dc:creator>Dave Beckett</dc:creator>
    <dc:publisher>ILRT, University of Bristol</dc:publisher>
    <dc:date>2002-07-31</dc:date>
  </rdf:Description>
</rdf:RDF>
```