UNICA
UNIVERSITÀ
DEGLI STUDI
DI CAGLIARI

sAIfer Lab
Joint lab on Safety and Security of AI

Formati di scambio dati: XML e JSON

Maura Pintor
maura.pintor@unica.it

In questa lezione

Formati di scambio dati

- EXtensible Markup Language (XML)
- JavaScript Object Notation (JSON)

Formati di scambio dati

Abbiamo visto come visualizzare i documenti e applicare degli stili

Adesso ci concentreremo sul contenuto, nello specifico come passare dei dati al documento HTML in modo che questi siano visualizzati dentro esso

EXtensible Markup Language (XML)

L'XML è un formato che consente la rappresentazione di dati strutturati (per esempio dati da inserire in tabelle)

Nasce come sottoinsieme semplificato di SGML (quello stesso insieme a cui appartiene l'HTML)

Anche questo è un **linguaggio a marcatori** (quindi funziona con i **tag**), dove ogni marcatore individua un elemento

I documenti in XML sono leggibili sia da computer che da umani (meglio se formattati con indentazione corretta)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <subject>Greetings</subject>
  <message>Hello!</message>
</note>
```

Caratteristiche di XML

XML è indipendente dalla piattaforma in cui viene utilizzato, e anche dalla applicazione per cui lo si vuole usare

Può essere archiviato in ogni tipo di supporto digitale e aperto con qualunque editor di testo

Può essere facilmente trasmesso via internet

Sono disponibili diverse librerie per la gestione e manipolazione di dati in XML, per esempio per formattarli quando vengono estratti da un **database**

Metalinguaggi

XML è un metalinguaggio (linguaggio per descrivere linguaggi), che vuol dire che definisce delle regole **metasintattiche (sintassi per scrivere le sintassi)** per descrivere un linguaggio

Per questo motivo, si definisce una grammatica, ovvero si stabiliscono delle regole che indicano quali elementi possono essere usati e con quale struttura

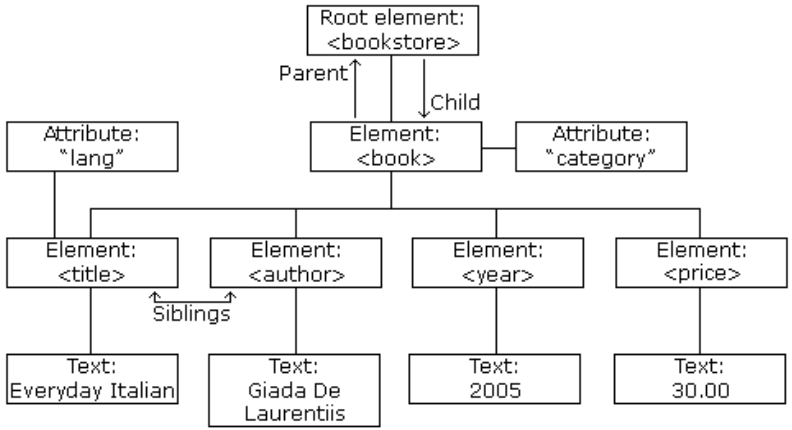
In XML, i tag non sono predefiniti, ma ci sono delle regole che dicono che vanno aperti e chiusi, e ci sono dei caratteri speciali che li delimitano

In XML, l'autore deve definire i tag e la struttura del documento

Documenti XML

I documenti XML sono composti da elementi, che rappresentano componenti logici del documento
Ogni elemento può avere ulteriori sotto-elementi
La forma dei documenti XML parte con un nodo **radice** e tanti rami, fino alle **foglie**
I rapporti tra i nodi vengono spesso descritti con termini quali **"parent", "child", "siblings"**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  </bookstore>
```



Regole di grammatica generali per XML

Il documento è una stringa di caratteri ASCII o Unicode, che può contenere anche elementi utili alla visualizzazione (spazi, a capo, commenti)

I nomi di elementi, attributi ed entità sono case sensitive, che vuol dire che <tag> e <Tag> sono due elementi diversi

I tag sono delimitati da caratteri speciali “<“, “>”, “&” (parentesi angolari e ampersand)

Questi caratteri non possono comparire come contenuto, ma ci sono dei rimpiazzi, come in HTML (rispettivamente <, >, e &)

I commenti sono delimitati dalla sintassi speciale

```
<!-- Commento -->
```

Struttura dei documenti XML

Il documento è composto da due pezzi:

- Prologo: contiene una dichiarazione XML ed eventualmente riferimenti ad altri documenti
  - contiene una dichiarazione XML sulla versione da usare e opzionalmente sul set di caratteri

```
<?xml version="1.0" encoding="UTF-8"?>
```

  - eventualmente può fare riferimento a documenti esterni (CSS, regole per la validazione, etc.)

```
<?xml-stylesheet type="text/css" href="style.css"?>
```

- Corpo: documento XML con i dati e contenuti

Elementi e tag

Anche in questo caso lo start ed end tag racchiudono gli elementi (come in HTLM)
Lo start tag contiene un nome più eventuali attributi racchiusi dai simboli "<" e ">"

```
<NomeTag lista-attributi>
```

Il tag di fine contiene lo stesso nome dello start tag e il carattere di chiusura "</" e ">"

```
</NomeTag>
```

Si possono rappresentare anche tag vuoti come elementi automaticamente chiusi

```
<NomeTag/>
```

Gli attributi XML

A ogni elemento possono essere associati degli attributi identificati da un nome e da un valore

```
<immagine file="immagine1.jpg"></immagine>
```

Gli elementi si possono annidare, gli attributi invece no, e si riferiscono solo all'elemento a cui sono associati (non possono esistere senza l'elemento)

Validità dell'XML

Per essere valido (rispettare la grammatica), un XML deve essere ben formato (rispettare le regole sintattiche)

- la dichiarazione deve essere corretta
- ogni elemento deve avere un tag di apertura e uno di chiusura
- l'annidamento deve essere corretto (i tag aperti devono essere richiusi in ordine)
- i valori degli attributi devono essere racchiusi tra singoli o doppi apici

Document Type Definition (DTD) e validazione dei documenti

Il DTD è alla base della definitzione della grammatica

Si compone di un elenco di dichiarazioni (markup declaration) che descrivono la struttura del documento (gli elementi che lo compongono e che attributi possono avere)

Una volta definita la struttura, è possibile **validare** il documento per identificare eventuali errori di formato

Più recentemente, si è passati a XML Schema Definition (XSD), ovvero la descrizione del documento tramite lo stesso XML

XML Schema Definition (XSD)

Un documento XSD può comprendere

- Dichiarazioni di **elementi**: Definiscono proprietà come nome, tipo e vincoli degli elementi.
- Dichiarazioni di **attributi**: Definiscono proprietà come nome, tipo e valori predefiniti.
- **Tipi semplici e complessi**: Strutturano e vincolano elementi e attributi nel documento a dei tipi di dato.
  - semplici: es. intero, stringa, …
  - complessi: es. durata (composto da due date, quella di inizio e quella di fine)
- Definizioni di gruppi di modelli e attributi, e il loro utilizzo
- Altri componenti specializzati

Tutte queste regole consentono poi di verificare se un documento XML le rispetta (validazione)

Document Object Model (DOM)

Il DOM è un modello per creare i contenuti XML

Definisce il contenuto tramite un albero in memoria

Esistono 3 interfacce base:

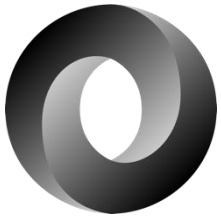
- Node (alla base di tutto)
- NodeList (collezione di nodi)
- NamedNodeMap (collezione di attributi)

Per passare da un XML a un DOM e viceversa si utilizza un'applicazione dedicata chiamata **parser DOM**

Altri formati di scambio dati: JavaScript Object Notation (JSON)

Il JSON, come l'XML, è un formato di testo per la conservazione e il trasporto dei dati
Rispetto all'XML, è stato realizzato per essere più facile da caricare in JavaScript

- XML è un formato più "stringente", il JSON non supporta lo schema
- Rispetto all'XML è però più rapido da realizzare e leggere e più compatto
  - ha meno ridondanza, per esempio rimuove la ripetizione del nome del tag nella chiusura dell'elemento)



JavaScript Object Notation (JSON)

Per chi conosce Python, il JSON è simile ai dizionari
Un esempio di JSON è il seguente:

```
'{"name":"John", "age":30, "car":null}’
```

Questo JSON definisce un oggetto con 3 proprietà:

- nome
- età
- macchina

Come XML, il JSON è indipendente dal linguaggio di programmazione usato nel server
Può facilmente essere letto e compreso da programmi in JavaScript e Python, anche qui tramite applicazioni parser

Sintassi JSON

La sintassi di JSON è un sottoinsieme della sintassi della definizione degli oggetti in JavaScript

- i dati sono organizzati in coppie chiavi-valore
  - il nome e il valore sono contenuti dalle doppie virgolette “, e separati dai due punti :
- i dati sono separati dalle virgole
- le parentesi graffe { e } contengono gli oggetti
- gli array sono contenuti in parentesi quadre [e]
  - gli array sono collezioni ordinate di oggetti

```
'{"name":"John", "age":30, "car":null}’
```

JSON vs XML

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

JSON

```
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>
```

XML

JSON vs XML

Somiglianze

- Entrambi sonon auto-descritti (e facilmente leggibili dagli umani)
- Entrambi sono gerarchici (possono avere valori annidati)
- Entrambi sono letti e usati in molti linguaggi di programmazione
- Entambi possono essere scaricati con delle GET HTTP

Differenze

- JSON non ha il tag di chiusura, ma chiude gli elementi con le virgolette e le parentesi
- JSON è più compatto
- JSON è più facile da leggere e scrivere
- JSON può usare gli array (vedremo più dettagli in seguito)

I tipi di dato JSON

I valori del JSON devono essere di uno dei seguenti tipi

- stringhe
- numeri (interi o in virgola mobile)
- oggetti (altri JSON)
- array
- booleani
- *null*

```
{"employee":
  {"name":"John",
   "age":30,
   "city":"New York”
   }
 }
```

Gli array in JSON

Gli array sono un tipo di dato composto che raggruppano più valori (simili agli array in JavaScript)

In JSON, gli array possono contenere elementi di tipo stringa, oggetto, array, boolean, o null

```
{
"name":"John",
"age":30,
"cars":["Ford", "BMW", "Fiat"]
}
```

Gli array sono particolarmente comodi per scrivere dei cicli (loop) e scorrere tra gli elementi

- es. per ogni macchina, visualizza il nome in una nuova riga