Félig strukturált adatok: XML, JSON

Imre Gábor

Q.B224

gabor@aut.bme.hu

BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék

XML

XML: eXtensible Markup Language

- Adatok szöveges, platformfüggetlen reprezentációja
- Emberi szemmel és programmal is olvasható
- Célja: egyszerű, általános használat
- Eredetileg dokumentum leírásnak készült
- Sok más helyen is használják: pl. RSS, Atom, SOAP, OpenXML, XHTML, OpenDocument, …
- Önleíró, pl.

```
<?xml version="1.0"?>
<note>
    <to>John</to>
    <from>Peter</from>
    <heading>Reminder</heading>
    <body>Don't forget the homework!</body>
</note>
```

Jól-formált XML

- Minden elem <elem_név> nyitótag és </elem_név> zárótagből, vagy egy üres <elem_név/> tagből áll
- Minden elemnek lehetnek attribútumai, amelyeket idézőjelek közé kell tenni
- Egy gyökér eleme van
- Az elemek következhetnek szekvenciálisan vagy egymásba ágyazva
- Az elemek nem lapolódhatnak át
- Case sensitive

Érvényes (valid) XML

- Érvényes, ha megfelel egy előre definiált sémának:
  - különféle szabályok előírása pl.
    - lehetséges elemek,
    - elemek sorrendje,
    - egymásba ágyazása,
    - lehetséges attribútumok
- Az XML be tudja hivatkozni a saját sémáját, azzal szemben validálható
- Legelterjedtebb nyelv a séma definícióra: XSD (XML Schema Definition)
  - Alap típusokat is definiál, melyekből saját összetett típusokat állíthatunk össze
  - Maga is XML formátumot követ

XML feldolgozása 1.

- Document streaming:
  - nem kell az egész dokumentumot beolvasni, csak egy részét
  - pull parsing: az API-t meg kell hívni, hogy jöjjön adat
  - push parsing:
    - a parszer nyomja az adatokat, ha a kliens nem is dolgozza fel
    - a kliens által írt eseménykezelőket hívja meg a parszer
- DOM: Document Object Model
  - az XML dokumentum egészének beolvasása után előáll egy memóriabeli fa (Pl. Element, Attribute csomópontokkal), ebben
    - szabadon navigálhatunk (ehhez XPath is használható)
    - CRUD (Create, Retrieve, Update, Delete) is lehetséges
  - nagyobb erőforrásigény
- XSLT: eXtensible Stylesheet Language
  - Szabályalapú, eseményvezérelt programnyelv
  - XML formátumban írhatunk le vele egy transzformációt
  - XML + XSLT + XSLT engine = Transzformált (nem feltétlen XML!) kimenet

XML feldolgozása 2.

- XPath: lekérdező nyelv az XML csomópontjainak kiválasztására, pl.

/konyvtar/konyv[ar>5000]

```
<konyvtar>
  <cim>1118 Budapest ...</cim>
  <konyv>
    <cim nyelv="en">Harry Potter</cim>
    <ar>1234</ar>
  </konyv>
  <konyv>
    <cim nyelv="hu">Aranysárkány</cim>
    <ar>5678</ar>
  </konyv>
</konyvtar>
```

XML feldolgozása 3.

- XML (de)szerializáció (binding): objektum és annak XML reprezentációja közti oda-vissza konvertálás
  - A DOM általános Element/Attribute/… típusai helyett alkalmazásspecifikus típusok, pl.

```
var ser = new XmlSerializer(typeof(C));

ser.Serialize(<stream>, <obj>);

myobj = (C)ser.Deserialize(<stream>);
```

  - Testreszabás attribútumokkal

```
[XmlElement("Cim")]
public class Address
{
   [XmlAttribute("Varos")]
   public string City;
   [XmlIgnore]
   public int SzamoltTavolsag;
}
```

XML API-k

- Java
  - JAXP (Java API for XML Parsing), a JDK része
    - Streaming (push: SAX, pull: StAX)
    - DOM, XSLT, XPath
  - JAXB (Java API for XML Binding)
    - Java 6 tette a Java SE-be
    - Java 9 óta deprecated, a Java 11 ténylegesen ki is vette
      → külön libraryként használható
- .NET
  - A LINQ to XML plusz lehetőség

JSON

JavaScript Object Notation

- Kliensalkalmazások tantárgyból már ismert, de nemcsak JavaScript-ben használható!
- Az XML-hez hasonlóan önleíró, platformfüggetlen
- Kisebb méret, pl.

```
{
   "to": "John",
   "from": "Peter",
   "heading": "Reminder",
   "body": "Don't forget the homework!"
}
```

- Gyorsabb feldolgozás

JSON API-k

- Tipikusan (de)szerializálás (binding)
- De léteznek egyéb API-k is
  - JSON Object Model (~DOM)
  - JSON Streaming
  - JSONPath, pl. **$.konyv[?(@.ar > 5000)]**

JSON és XML tipikus használata

- Objektumszerializációs formátum frontend-backend vagy backend-backend kommunikáció során
  - Frontend-backend szinte mindig JSON, mert mind böngészőkben, mind mobil platformokon kézenfekvő a használata
- Konfigurációs fájlok (önleíró + ember számára is használható)
- Változékony sémájú adatok perzisztens tárolása
  - Relációs modellben egy varchar oszlop
    - gyártóspecifikus SQL kiterjesztésekkel akár kereshető is XPath/JSONPath kifejezésekkel
  - NoSQL dokumentum-adatbázisokban (pl. MongoDB)