Jemný úvod do spracovania XML

XML štandardy

- nevyhnutná súčasť moderného SW,
- umožňujú lepšiu integráciu rôznych aplikácií,
- štruktúrovanie údajov (typovanie, pomenovanie polí,...)
- vysoká flexibilita
- výhoda: XML súbory sú čitateľné aj pre ľudí (ručná úprava, konfigurácia),
- využitie štandardizovaných knižníc
- možnosť kontroly správnosti dokumentu (validácia)
- možnosť zaslať partnerovi bez toho, aby sme mu museli poslať aj parser...

XML

```
<zoznamDovoleniek>
–
<dovolenka id="1">
<miesto krajina="Slovensko">Chopok</miesto>
<rok>1994</rok>
</dovolenka>
–
<dovolenka id="2">
<miesto krajina="Taliansko">Catania</miesto>
<rok>1995</rok>
</dovolenka>
–
<dovolenka id="3">
<miesto krajina="Španielsko">Costa Brava</miesto>
<rok>1993</rok>
</dovolenka>
</zoznamDovoleniek>
```

XML

Textový obsah so značkami (tagmi), nemusí byť súbor – akýkoľvek “stream” - napríklad správa prijatá na sériovom porte;

Otváracia a uzatváracia značka, alebo prázdny element:
<zmrzlina></zmrzlina> alebo <zmrzlina /> alebo <zmrzlina/>

Hierarchicky usporiadané – jeden element je koreňový

Atribúty, <zmrzlina prichut=”cola”/>

Znakové entity, < & > " '

Alternatíva: sekcia CDATA:

```
<usekProgramu jazyk=”Java”>
<![CDATA[
      for i = 0; i < 5; i++) {
        System.out.format(“%d < 5\n”, i);
      }
  ]]>
</usekProgramu>
```

XML...

Sada viacerých formátov...

Formát XML súborov definujú schémové jazyky (DTD = document type definition)
- poradie, prítomnosť elementov
- ich typ, typy atribútov
- dovolené hodnoty, intervaly, ...

Použitie: validácia, definícia, XML editory, vytvorenie zodpovedajúceho objektového modelu

1. **DTD**: historicky prvý, nie je v XML, je široko podporovaný, avšak slabý: nedá sa validovať, nepopisuje dátové typy, nepodporuje menné priestory)
Alternatívy: W3C XML Schema (**XSD**) a Relax NG
Výhody: popisujú dátové typy a obmedzenia, sú XML

Transformácia do iných formátov (napr. HTML, PDF) pomocou **XSL** transformácií (eXtensible Stylesheet Language)

XML...

Komentáre:
`<!-- poznamka -->`

Skripty/kód:
`<?  ... ?>`

Deklarácia:

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

Špeciálne znaky:

`&#xčíslo;` (v hexa)

www.unicode.org/charts

Základné prístupy:

Prísna hierarchia, príklad
Plochá štruktúra, príklad

XML...

Spôsob spracovania:

Spracovanie “stream”-u: dáta sa neukladajú v pamäti iba sa priebežne parsujú, aplikácia si vyberá, čo potrebuje – príklad cvičenie 1, SAX parser – pozri 4. kapitola v knižke

Document Object Model (DOM): celý XML sa načíta do pamäte (interne väčšinou zasa pomocou SAX parsera), kde sa s ním dá pracovať a znovu uložiť do súboru – príklad: nasledujúce cvičenie.

DOM...

Je tiež súčasťou Java Core API: javax.xml.parsers

Podstatné triedy: DocumentBuilderFactory, DocumentBuilder

Interfejsy: org.w3c.dom
- viac ako 10 typov uzlov (Node), podstatné sú:
    Node – predchodca všetkých ostatných
    Document – začiatok dokumentu (nie koreňový element)
    Attr - atribút
    Element - element
    Text – hodnota elementu

Kolekcie uzlov: NodeList, NamedNodeMap

DOM...

Príklad – Jedlo z knižky Pavel Herout: Java a XML, 5.kapitola; zdrojáky on-line, kópia na

dai.fmph.uniba.sk/courses/java2/xml/

Node: getNodeType(), getNodeValue(), getParentNode(), getPreviousSibling(), getNextSibling(), hasChildNodes(), getFirstChild(), getLastChild(), getChildNodes(), hasAttributes(), getAttributes()

NodeList: getLength(), item(index)

NamedNodeMap: getLength(), getNamedItem(name), item(index)

DOM...

Príklady z knižky:

`jidlo.xml` – parsovaný súbor
`VerifikatorJidloDOM.java` – iba skontroluje, že súbor je korektný XML
`VahaJidloDOM.java` – zistí a vypíše celkovú váhu – ale obsahuje chybný predpoklad, že prvý poduzol uzlu <vaha> je textový, čo nemusí byť a text nemusí byť v celku!
`VahaJidloBezpecneDOM.java` – to isté, ale bez chybného predpokladu
`CenaJidloDOM.java` – vypočíta celkovú cenu – musí pristupovať k atribútom
`TransformaceJidloDOM1_6.java` – konvertuje XML na iné kódovanie
`ZmenaJidloDOM.java` – príklad ako meniť štruktúru v pamäti (mazať, meniť, pridávať) a zapísať výsledok