Technologie XML

Syntaxe XML

Jiří Měska (jiri.meska@gmail.com)

MIB008, Datové a procesní modely,

MFF UK Praha, 2020

Syntaxe a sémantika XML:

Řekneme si něco o syntaxi a sémantice následujících objektů XML:

- Elementů
- Atributů
- Kódování Base 64
- Jmenných prostorech

Element v XML:

- Element je pojmenovaní datového objektu
- Element uzavírá data mezi počátečním a koncovou značkou:

Příklad elementu "jmeno":

```
<jmeno>Přemysl Otakar II.</jmeno>
```

Element obsahuje textovou informaci

Příklad elementu "panoval":

```
<panoval>
        <od>1253</od>
        <do>1278</do>
</panoval>
```

Element obsahuje strukturovanou informaci tvořenou jinými elementy

Element nemusí obsahovat žádnou informaci:

```
<zemrel/>
```

Takový element lze zapsat ve zkrácené syntaxi

```
<seznam_panovniku>
    <panovnik rod="přemyslovec">
        <jmeno>Přemysl Otakar II.</jmeno>
        <titul>král český</titul>
        <panoval>
            <od>1253</od>
            <do>1278</do>
        </panoval>
    </panovnik>
</seznam_panovniku>
```

Element v XML:

Element může obsahovat:

- elementy,
- fragmenty textu
- atributy

```
<seznam_panovniku>
    <panovnik rod="přemyslovec">
            <jmeno>Přemysl Otakar II.</jmeno>
            <titul>král český</titul>
            <panoval>
                <od>1253</od>
                <do>1278</do>
            </panoval>
    </panovnik>
</seznam_panovniku>
```

Elementy se nemohou navzájem prolínat,
tj. elementy jsou buď disjunktní
nebo je jeden vnořen do druhého.

Struktura elementů tak tvoří strom s kořenovým elementem jako vrcholem.

Syntaxe jména elementu:

- Jméno musí být shodné v počáteční a koncové značce (tag)):
- Jméno je case sensitivní (malá a velká písmena mají význam)
- Může obsahovat písmena (i s diakritikou), číslice a znaky tečka, pomlčka, podtržítko
- Počátečním znakem musí být písmeno nebo podtržítko

Atribut v XML:

- Součástí úvodní značky elementu může být jeden nebo více atributů
- Atribut je tvořen dvojicí jméno a hodnota

<panovnik rod="přemyslovec">

- Hodnota může být uzavřena ve dvojitých nebo jednoduchých apostrofech

<panovnik rod='přemyslovec'>

<panovnik rod="přemyslovec">

- Syntaxe jména atribut je obdobná syntaxi jména elementu.
- Pořadí atributů nemá na rozdíl od elementů žádný význam

```
<seznam_panovniku>
   <panovnik rod="přemyslovec">
         <jmeno>Přemysl Otakar II.</jmeno>
         <titul>král český</titul>
         <panoval>
               <od>1253</od>
               <do>1278</do>
         </panoval>
      </panovnik>
</seznam_panovniku>
```

Co může dělat problémy:

Filozofickou otázkou je jaké texty ukládat do atributů a jaké do elementů. Databázové pojetí často svádí mapovat tabulky do elementů a sloupce do atributů. Doporučená filosofie je používat atributy pouze k metainformacím souvisejícím s obsahem elementu, ne k věcným informacím (např. id (odkaz), jazyk (obsahu elementu) etc.)

Pojem white space: mezera, tabulátor, odřádkování.

XML zpracování může nahradit sekvenci white spaces v atributu jednou mezerou. Důsledek je, že se ztrácí formátování. Platí to pouze o obsahu atributu ne o textovém obsahu v elementu!

Text v XML:

- Textové data jsou součástí elementu nebo atributu

- XML dokument je textovým souborem, proto binární data je nutno uložit v nějakém textovém formátu

- XML dokument musí být strojově zpracovatelný a proto ze syntaktických důvodů textová data nesmí obsahovat znaky: <,>:

    znak menší < nahradit <

    znak větší > nahradit >

- Výjimkou tvoří CDATA sekce nahrazující textový obsah. Vnitřek CDATA sekce není parsován. Textová data jsou v CDATA sekci

uzavřena ve speciální konstrukci:

    <![CDATA[ ]]>

    CDATA sekce se využívá např. pro Java script

- Znakové reference představují další způsob jak zapsat v textu libovolný znak. Znaková reference odkazuje na číselný kód znaku ze znakové sady ISO/IEC 10646, což je nadmnožina Unicode:

    - decimální formát &#číslo;

    - hexadecimální formát &#xčíslo;

Příklad CDATA sekce:

```
<script>
    < ![CDATA[
    function matchwo(a,b)
    {
    if (a < b && a < 0) then
      {
      return 1;
      }
    else
      {
      return 0;
      }
    }
    ]]>
</script>
```

Ukládání binárních dat pomocí BASE64:

Kódování textové informace. Kódují se vždy 3 byte pomocí 4 znaků.

6 bitů reprezentuje 64 základních tisknutelných znaků, index 0…63 - znaky 'A' ... '/'

Binární data `01001101|01100001|01101110`

Rozložení `010011|010110|00001|101110 do 4x6bitů`

Index `19 |22 |5 |46`

Znaky `T |W |F |u` dle kódovací tabulky:

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Na konci se doplňuje/doplňují do trojice =

Jmenné prostory

Pomocí jmenných prostorů

můžeme definovat "XML jazyky".

Příklad: V našem dokumentu se prolínají data ze dvou zdrojů:

1. Historická data o panovníkovi, označený jmenným prostorem:

   xmlns:pekar="http://www.historie.cz/pekar"

   pojmenovaný po českém historikovi Josefu Pekařovi

   Pekař byl současníkem Masaryka a současně jeho odpůrce ve věci filosofie českých dějin, doporučuji přečíst jeho dílko o Bílé hoře.

2. Organizační data určená pro vytvoření rejstříku, označená jmenným prostorem:

   xmlns:rejstrik="http://www.historie.cz/rejstrik"

Jmenný prostor je definován pomocí **URI** (Uniform Resource Identifier)

Poznámka: Hodnota URI jmenného prostoru nemá jiný než jednoznačný význam! Proto se používá URI odvozené od doménové adresy

```
<seznam_panovniku
    xmlns:pekar="http://www.historie.cz/pekar"
    xmlns:rejstrik="http://www.historie.cz/rejstrik">

 <pekar.panovnik pekar.rod="přemyslovec">
      <jmeno><rejstrik:jmeno>Vratislav II</rejstrik:jmeno></jmeno>
      <tituly><titul>král český</titul></tituly>
      <panoval titul="král český">
        <od>1061</od> <do>1092</do>
      </panoval>
      <smrt>Umírá na následky zranění při pádu z koně na lovu.</smrt>
      <manželky>
        <manželka><rejstrik:jmeno>Marie</rejstrik:jmeno></manželka>
        <manželka><rejstrik:jmeno>Adleyta</rejstrik:jmeno>, dcera uherského krále
    Ondřeje I.</manželka>
        <manželka><rejstrik:jmeno>Svatava</rejstrik:jmeno>, dcera polského krále
    Kazimiera</manželka>
      </manželky>
      <poznamka> Co se týče zahraniční politiky, před rokem 1075 se Vratislav II. v době
    bojů o investituru sbližuje s císařem <rejstrik:jmeno>Jindřichem IV.</rejstrik:jmeno> ...
    </poznamka>

  </pekar.panovnik>

</seznam_panovniku>
```

Proč potřebujeme dva jmenné prostory?

V našem dokumentu se prolínají data ze dvou zdrojů:

1. Historická data o panovníkovi, označený jmenným prostorem:

xmlns:pekar="http://www.historie.cz/pekar"

2. Organizační data určená pro vytvoření rejstříku, označená jmenným prostorem:

xmlns:rejstrik="http://www.historie.cz/rejstrik"

Můžeme mít programy:

1. Program který pracuje s historickými daty, tj. zpracovává elementy z jmenného prostoru

xmlns:pekar="http://www.historie.cz/pekar"

2. Program, který vytváří rejstřík jmen:

xmlns:rejstrik="http://www.historie.cz/rejstrik"

Poznámka: Můžeme si představit definovat třídu XML elementů popisující živočišné druhy. V rámci popisu druhů lze využít značek rejstříku k popisu názvu druhů a mít jeden program, který bude dělat jmenný rejstřík pro různá XML.

Poznámka: Podobně to může být s formátováním

Výchozí jmenný prostor elementu lze definovat

Jmenný prostor elementu <panovnik>

lze definovat:

- odkazem pomocí aliasu (vlevo)
- nebo explicitně (dole)

```
<seznam_panovniku
xmlns:pekar="http://www.historie.cz/pekar"
xmlns:rejstrik="http://www.historie.cz/rejstrik"
>

...
  <pekar.panovnik rod="přemyslovec">
```

```
<panovnik xmlns ="http://www.historie.cz/pekar"  rod="přemyslovec">
```

V obou případech všechny podelementy elementu **dědí** výchozí jmenný prostor.

V našem úvodním příkladě elementy jmeno, tituly, titul, panoval, od, do, smrt, hodnocení zdědí jmenný prostor popsaný aliasem pekar.

Výchozí jmenný prostor lze v podelementu **předefinovat** (explicitně nebo pomocí aliasu), což jsme udělali v případě elementu <rejstrik.jmeno>.

Jmenný prostor atributu

Atributy nepatří do žádného jmenného prostou, pokud jim není jmenný prostor explicitně přiřazen prostřednictvím aliasu.

```
<tomek.panovnik pekar.rod="přemyslovec">
    <jmeno>Vratislav II</jmeno>
    <tituly>
      <titul>král český</titul>
    </tituly>
    <panoval titul="král český">
      <od>1061</od>
      <do>1092</do>
    </panoval>
  </tomek.panovnik>
```

Element panovnik a všechny jeho podelementy patří do jmenného prostoru pod aliasem tomek. Atribut pekar.rod patří do téhož jmenného prostoru. Atribut titul nepatří do žádného jmenného prostoru.

XML technologie (oblasti použití, standardy a nástroje)

Persistence (ukládání XML dat)

Souborový systém

Relační databáze – XML datový typ, na rozdíl od varchar podpora SQL/XML

Nativní XML DB – XML databáze

Validace XML (popis syntaxe třídy XML dokumentů):

DTD – součást standardu XML

XML Schémata – nejrozšířenější

a další

Parsování XML (programové zpracování XML dat):

SAX Parser – událostně řízený parser, výhodou je možnost streamového zpracování a v důsledku toho možnost zpracování neomezeně velkých dokumentů

DOM (Document Object Model) – programové rozhraní pro práci s XML, výhodou je přístup k celému dokumentu, nevýhodou omezení na velikost dokumentu, který se musí vejít do paměti

XML technologie (oblasti použití, standardy a nástroje)

SQL/XML (budeme se podrobněji zabývat v dalších přednáškách)

Integrovaná technologie pro generaci XML dat z databázových struktur

Zpracování XML dat (budeme se podrobněji zabývat v dalších přednáškách)

XPath – adresace dat v XML

XSL – formátování XML dat

XSLT – deklarativní jazyk pro transfomaci XML dat

XQuery– dotazování jazyk pro vyhledávání dat v XML, obdoba SQL

Konec

Obsah XML Kurzu:

Úvod do XML

Syntaxe XML

XML Validace

SQL/XML

XPath

XSLT