Jmenné prostory

Spolufinancováno Evropskou unií

Ministerstvo školství, mládeže a tělovýchovy

Projekt: UčiTUL
CZ.02.02.XX/00/23_019/0008386

Jmenné prostory

- počítá se s modularitou XML
  - jeden dokument může používat značky z několika specializovaných jazyků (např. pro matematické a chemické vzorce)
- problém: konfliktní jména prvků
- řešení: jmenné prostory
  - prostor zahrnuje příbuzné prvky (z jednoho modulu)
  - kompletní identifikace: jmenný prostor + prvek

Definice (1)

- **jmenný prostor**
  - je **identifikován lokátorem (URI)** = jméno prostoru
  - zahrnuje jména prvků a atributů
  - jméno (URI) příliš dlouhé, bývá definována **zkratka (prefix)** pro zařazení identifikátoru do jmenného prostoru
  - URI musí být absolutní
- **expandované jméno**
  - pár zahrnující jméno jmenného prostoru a lokální jméno

Definice (2)

- **kvalifikované jméno**
  - jméno prvku v XML dokumentu
  - je interpretováno z hlediska příslušnosti ke jmennému prostoru
  - **bez prefixu** (např. h1) – náleží do implicitního jmenného prostoru, závisí na kontextu
  - **s prefixem** (např. html:h1) – jmenný prostor je určen prefixem

Deklarace jmenného prostoru

- speciálním atributem prvku, platí pro daný prvek a jeho potomky (nejčastěji kořenový prvek)
- **xmlns="*URI*"**
  deklaruje **implicitní jmenný prostor**, platí pro jména prvků bez prefixu; ne pro atributy – atribut bez prefixu není v žádném jmenném prostoru
- **xmlns:*prefix*="*URI*"**
  deklaruje jmenný prostor označený v kvalifikovaných jménech daným ***prefixem***

Příklad dokumentu

```
<h:html   xmlns:xdc="http://www.xml.com/books"
          xmlns:h="http://www.w3.org/1999/xhtml">
 <h:head><h:title>Book Review</h:title></h:head>
 <h:body>
  <xdc:bookreview>
   <xdc:title h:style="color: red;">XML: A Primer</xdc:title>
   <h:p>Autor: <xdc:author>Simon St. Laurent</xdc:author></h:p>
   <h:p>Vydáno: <xdc:date>1998/01</xdc:date></h:p>
   <h:p>Dobrý úvod do XML.</h:p>
  </xdc:bookreview>
 </h:body>
</h:html>
```

Implicitní prostor

```
<html      xmlns:xdc="http://www.xml.com/books"
           xmlns="http://www.w3.org/1999/xhtml">
 <head><title>Book Review</title></head>
 <body>
  <xdc:bookreview>
   <xdc:title>XML: A Primer</xdc:title>
   <p>Autor: <xdc:author>Simon St. Laurent</xdc:author></p>
   <p>Vydáno: <xdc:date>1998/01</xdc:date></p>
   <p>Dobrý úvod do XML.</p>
  </xdc:bookreview>
 </body>
</html>
```

Rezervované prefixy

- **xml**
  - vázán na http://www.w3.org/XML/1998/namespace
  - nesmí být přiřazen jinému URI a toto URI nesmí být přiřazeno jinému prefixu
- **xmlns**
  - vázán na http://www.w3.org/2000/xmlns/
  - nesmí být deklarován
- všechny prefixy začínající **xml...**

XML Schema

XML Schema

- označováno **XSD** (XML Schema Definition)
- specifikace vyvíjená WWW konsorciem
- jmenný prostor xmlns:xsd="http://www.w3.org/2001/XMLSchema"
- výrazově dost silné, ale kritizováno za složitost
- řada předdefinovaných typů
- hlavní síla ve schopnosti definovat si vlastní typy

Jednoduché datové typy (1)

- celkově přes 50
- slouží **pro koncové hodnoty** (žádné vnořené prvky)
- **celá čísla:** integer, positiveinteger, unsignedint (32 bitů), short (16 bitů), byte (se znaménkem),...
- **necelá čísla:** decimal (s desetinnou částí), float (32 bitů), double (64 bitů),...
- **časy:** date (2006-10-23), time (14:20:00), duration (délka časového intervalu, P2H15M),...

Jednoduché datové typy (2)

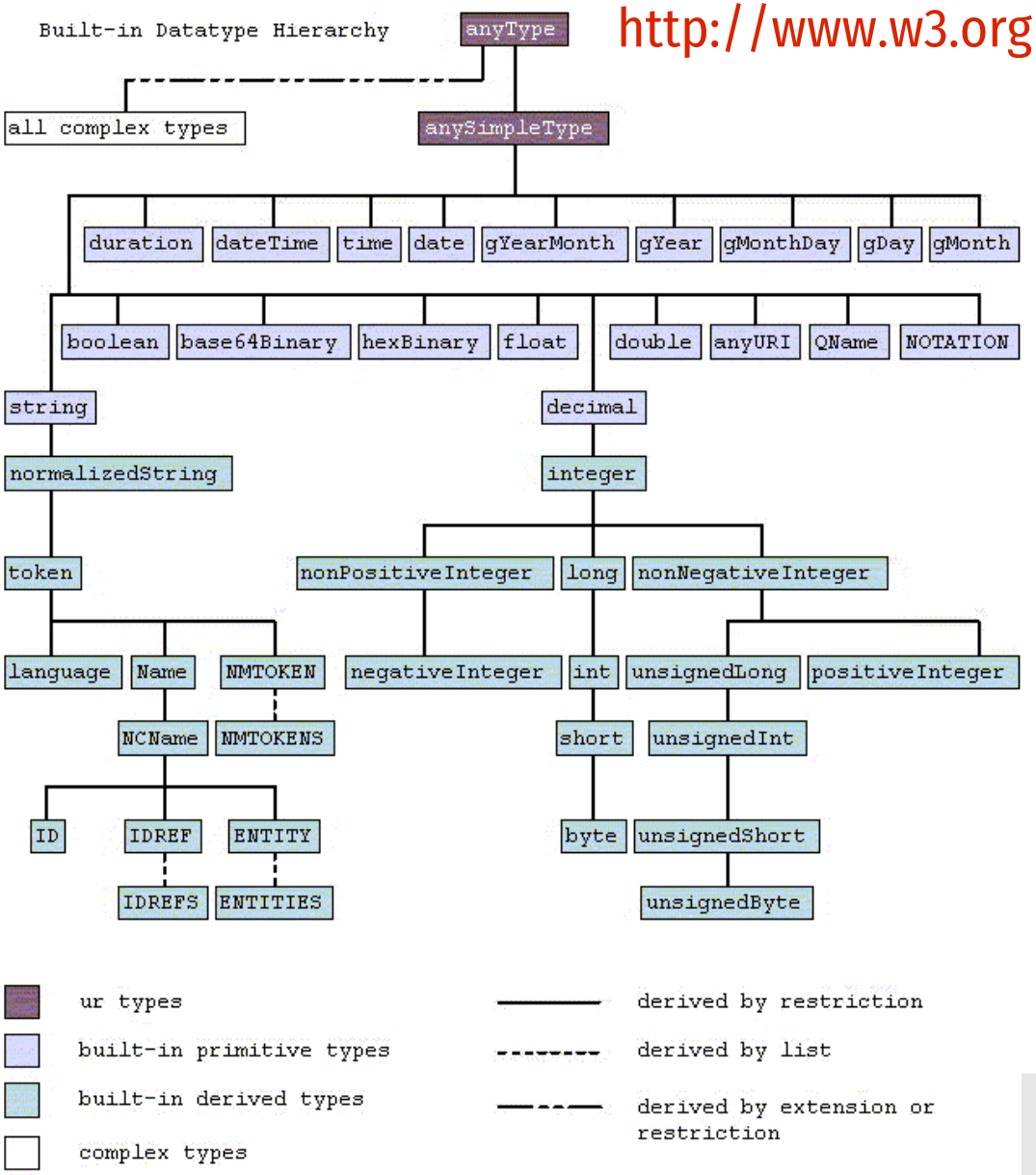
- **řetězce:** string, normalizedString (tabulátory a konce řádků nahrazeny mezerami), token (navíc sloučeny skupiny mezer a odstraněny mezery na začátku/konci řetězce)
- **z DTD:** ID, IDREF, IDREFS, ENTITY, ENTITIES, NMTOKEN, NMTOKENS, NOTATION

Jednoduché datové typy (3)

- **z XML:** Name (à la jméno prvku/atributu), NCName (jméno bez dvojtečky), language (kód jazyka)
- **binární:** hexBinary (bajt=2 šestnáctkové číslice), base64Binary (data v MIME kódování BASE64)
- **z webu:** anyURI (lib. URI adresa, absolutní i relativní)

Hierarchie typů

http://www.w3.org/TR/xmlschema-2/#built-in-datatypes



Odvozování nových typů

- **restrikce (restriction):**
  - omezí se možné hodnoty
- **seznam (list):**
  - vytváření seznamů hodnot daného typu
- **spojení (union):**
  - spojuje několik typů
- **rozšíření (extension):**
  - přidává děti a/nebo atributy (pro složené typy)

Restrikce řetězců

- prvek **restriction**
- atribut **base** udává výchozí typ
- obsahem je konkrétní omezení
- <xsd:**restriction base="xsd:token**">

  ...

  </xsd:restriction>

Length

- délka (**length**) udává přesný počet znaků řetězce
- hodnota je v atributu **value**
- příklad: telefonní číslo ČR jako řetězec 9 znaků

```
<xsd:simpleType name="telefonType">
  <xsd:restriction base="xsd:token">
    <xsd:length value="9"/>
  </xsd:restriction>
</xsd:simpleType>
```

minLength, maxLength

- rozmezí délek: **minLength** a **maxLength**
- příklad: tří až pětipísmenná zkratka

```
<xsd:simpleType name="zkratkaType">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="3"/>
    <xsd:maxLength value="5"/>
  </xsd:restriction>
</xsd:simpleType>
```

Enumeration

- výčet hodnot: **enumeration**
- příklad: fakulty naší univerzity

```
<xsd:simpleType name="fakultaTULType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="FS"/>
  <xsd:enumeration value="FT"/>
  <xsd:enumeration value="FP"/>
  <xsd:enumeration value="EF"/>
  <xsd:enumeration value="FM"/>
  <xsd:enumeration value="FUA"/>
  <xsd:enumeration value="FZS"/>
 </xsd:restriction>
</xsd:simpleType>
```

Pattern

- regulární výrazy: **pattern**
- syntaxe podle jazyka Perl
- příklad: telefon – 3 skupiny 3 číslic, lze s mezerami

```
<xsd:simpleType name="telefonType">
  <xsd:restriction base="xsd:token">
    <xsd:pattern value="\d{3} ?\d{3} ?\d{3}"/>
  </xsd:restriction>
</xsd:simpleType>
```

Restrikce čísel

- rozmezí hodnot: **minInclusive**, **minExclusive**, **maxInclusive**, **maxExclusive**
- počet platných číslic: **totalDigits**
- přesnost: **fractionDigits**

Příklad číselné restrikce

- cena, do 1 000 000, uvádí se na 2 desetinná místa
-
```
<xsd:simpleType name="cenaType">
  <xsd:restriction base="xsd:decimal">
    <xsd:minInclusive value="0"/>
    <xsd:maxExclusive value="1000000"/>
    <xsd:fractionDigits value="2"/>
  </xsd:restriction>
</xsd:simpleType>
```

Použití typu v definici prvku

- základní – pojmenovaný typ:

```
<xsd:element  name="telefon"  type="telefonType"/>
```

- lze i anonymně:

```
<xsd:element name="telefon">
  <xsd:simpleType>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="\d{3} ?\d{3} ?\d{3}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Posuzování hodnot

- normalizace
  - nahrazení tabulátorů a konců řádků mezerami
  - sloučení skupin mezer do jedné
  - odstranění mezer na začátku a konci hodnoty
  - neprovádí se pro string a normalizedString a jejich odvozeniny
- po normalizaci se převede na interní hodnotu
  - další omezení se ověřují pro ni

Složené datové typy

Složené datové typy

- popisují strukturu dokumentu
- definují **vzájemné vztahy jednotlivých prvků**
- tři základní druhy:
  - **sequence** – prvky se musí vyskytovat v daném pořadí
  - **choice** – vyskytuje se jeden z uvedených prvků
  - **all** – prvky se mohou vyskytovat v libovolném pořadí
- lze vnořovat (kromě all)

sequence

- prvky musí být obsaženy v daném pořadí
- počet výskytů jim lze předepsat atributy minOccurs a maxOccurs
  - hodnotou číslo od 0 nebo unbounded (neomezeně)

Příklad sekvence

- prvek kontakt obsahující jméno, nepovinný mail a alespoň jeden telefon

```
<xsd:element name="kontakt">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="jmeno" type="xsd:string"/>
      <xsd:element name="email" type="xsd:string"
                   minOccurs="0"/>
      <xsd:element name="telefon" type="telefonType"
                   maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

choice

- vyskytuje se právě jeden z uvedených prvků
- příklad: osoba s rodným číslem nebo číslem pasu

```
<xsd:element name="osoba">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="jmeno" type="xsd:string"/>
   <xsd:choice>
     <xsd:element name="rodne" type="xsd:string"/>
     <xsd:element name="pas" type="xsd:string"/>
   </xsd:choice>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

all

- prvky se mohou vyskytovat v libovolném pořadí
- maxOccurs dětí musí být jedna
  - de facto má smysl uvažovat jen o minOccurs="0" pro nepovinné
- lze použít jen přímo uvnitř složeného typu
- **smí obsahovat jen definice prvků (element)**, nikoli složené typy

Volnější obsah

- diskoteka obsahuje libovolný počet cd a dvd v libovolném pořadí

```
<xsd:element name="diskoteka">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:choice minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="cd" type="cdType"/>
    <xsd:element name="dvd" type="dvdType"/>
   </xsd:choice>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

Smíšený obsah

- atributem mixed="true" u prvku complexType
- povoluje text mezi definovanými prvky (přísnější než v DTD)
- chceme-li skutečně volný obsah

```
<xsd:complexType mixed="true">
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
    povolené prvky
  </xsd:choice>
</xsd:complexType>
```

Prvky

- prvek **element**; atributy:
  - **name** – jméno prvku, povinný
  - **type** – typ prvku (co je jeho obsahem);
    pokud chybí, může být typ obsahu definován přímo na místě jako obsah prvku element
  - **default** – implicitní obsah prvku
  - **fixed** – pevně definovaný obsah prvku

Atributy

- prvek **attribute**; jeho atributy:
  - **name** – jméno atributu, povinný
  - **type** – typ atributu (omezuje hodnotu), povinný
  - **use** – hodnota "required" stanoví povinný atribut
  - **default** – implicitní hodnota
- **musí být na konci (za prvky)**
- do složených typů se přidává normálně

Příklad

- přidáme kontaktu povinný identifikátor:

```
<xsd:element name="kontakt">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name="jmeno" type="xsd:string"/>
   <xsd:element name="email" type="xsd:string"
          minOccurs="0"/>
   <xsd:element name="telefon" type="telefonType"
          maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
 </xsd:complexType>
</xsd:element>
```

Prázdný prvek

- složený typ bez obsahu:

```
<xsd:element name="br">
  <xsd:complexType/>
</xsd:element>
```

Rozšíření jednoduchého typu

- přidání atributu je rozšířením jednoduchého typu
- příklad: odkaz

```
<xsd:element name="odkaz">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute name="url" type="xsd:anyURI"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

Vytvoření schématu

Obal schématu

- celé schéma obaluje prvek **schema** z jmenného prostoru přiděleného XML Schema:

```
<xsd:schema
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
... definice schématu ...
</xsd:schema>
```

- atributy umožňují určit různé vlastnosti

Připojení schématu k dokumentu

- pomocí atributů z prostoru http://www.w3.org/2001/XMLSchema-instance
- obvyklým prefixem je xsi
- atribut noNamespaceSchemaLocation pokud nepoužíváme jmenné prostory, hodnotou lokátor

```
<mujdokument
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="schemadokumentu.xsd">
...
</mujdokument>
```

Připojení se jmenným prostorem

- atribut schemaLocation
- hodnotou dvojice (i několik)
  URI_prostoru lokátor_schématu

```
<book isbn="0836217462"
    xmlns="http://example.org/ns/books/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
        "http://example.org/ns/books/  library.xsd">
...
</book>
```

Jak vytvářet schéma?

- obvyklé jsou tři základní metody:
  - matrjoška
  - plátkování
  - žaluziová metoda (slepý Benátčan)
- volba závisí především na složitosti a charakteru popisovaného jazyka
- lze kombinovat (ale s mírou)

Příklad: ceník zboží

- poslouží k ilustraci jednotlivých přístupů
- definice XML prvku zbozi, který má mít při použití následující tvar:

```
<zbozi kod="1267">
    <nazev>Žízeň Killer – voda 1,5 l</nazev>
    <cena>8.50</cena>
</zbozi>
```

Matrjoška

- definice prvků se do sebe ve schématu vkládají přímo, stejně jako prvky v dokumentu
- anonymní typy
- na nejvyšší úrovni vždy jen jeden prvek
- krátké a kompaktní schéma
- pro složitější jazyky nepřehledné
- nelze opakovaně využívat dílčí definice

Matrjoška – příklad

```
<xsd:element name="zbozi" maxOccurs="unbounded">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="nazev" type="xsd:string"/>
            <xsd:element name="cena" type="xsd:decimal"/>
        </xsd:sequence>
        <xsd:attribute name="kod" type="xsd:integer"/>
    </xsd:complexType>
</xsd:element>
```

Plátkování

- všechny prvky a atributy se definují na stejné (nejvyšší) úrovni
- odkazují se na sebe pomocí ref="*jméno*"
- definované prvky/atributy lze používat opakovaně
- koncepce blízká DTD
- obsah prvku se nemůže lišit podle kontextu (rodiče)

Plátkování – příklad

```
<xsd:element name="nazev" type="xsd:string"/>
<xsd:element name="cena" type="xsd:decimal"/>
<xsd:attribute name="kod" type="xsd:integer"/>

<xsd:element name="zbozi">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="nazev"/>
            <xsd:element ref="cena"/>
        </xsd:sequence>
        <xsd:attribute ref="kod"/>
    </xsd:complexType>
</xsd:element>
```

Žaluziová metoda

- předem se (na globální úrovni) definují typy
- prvky a atributy se definují lokálně (jako v matrjošce), ovšem triviálně s využitím připravených typů
- nejflexibilnější
- typy lze používat opakovaně
- ale schéma je delší
- vhodné pro složité jazyky

Žaluzie – příklad (1)

```
<xsd:simpleType name="nazevTyp">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="50"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="cenaTyp">
    <xsd:restriction base="xsd:decimal">
        <xsd:fractionDigits value="2"/>
    </xsd:restriction>
</xsd:simpleType>
```

Žaluzie – příklad (2)

```
<xsd:simpleType name="kodTyp">
    <xsd:restriction base="xsd:integer"/>
</xsd:simpleType>

<xsd:complexType name="zboziTyp">
    <xsd:sequence>
        <xsd:element name="nazev" type="nazevTyp"/>
        <xsd:element name="cena" type="cenaTyp"/>
    </xsd:sequence>
    <xsd:attribute name="kod" type="kodTyp"/>
</xsd:complexType>

<xsd:element name="zbozi" type="zboziTyp"/>
```

Spolufinancováno Evropskou unií

Ministerstvo školství, mládeže a tělovýchovy

Projekt: UčiTUL
CZ.02.02.XX/00/23_019/0008386