XML – XSD

MGR. TOMÁŠ HAVETTA, MCT

Představení

- Jméno
- Firma
- Pozice
- Odpovědnost
- Aktuální znalost XML, XSD, XSLT, Xpath
- S jakým nástrojem pracujete
- Očekávání od kurzu

Osnova

- Historie
- Základy XSD
- XML namespace
- Komplexní typy v XSD
- Použití schémat

Technické informace

- Čas výuky
- Přestávky, nápoje
- Oběd
- Kuřáci
- Kód pro vstup
- Toalety
- ???

1. Historie

- XML
- DTD
- XSD

Začátky XML

- Konec 80. let 20. století se používá SGML *(Standard Generalized Markup Language)*
- XML vzniklo jako podmnožina SGML v letech 1995 – 1997
- O jeho standardizaci se staralo W3C konsorcium firem
- Verze 1.0 byla specifikována W3C dne 10. února 1998
- Verze 1.1 byla publikována 4. února 2004, reálně se ale moc nepoužívá

XML – základní prvky

```
<?xml version="1.0" encoding="UTF-8"?>
<expense-report xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="ExpReport.xsd"
currency="USD" detailed="false"
total-sum="244.23">
    <Person>
        <First>Fred</First>
        <Last>Landis</Last>
        <Title>Project Manager</Title>
        <Phone>123-456-7890</Phone>
        <Email>f.landis@nanonull.com</Email>
    </Person>
    <expense-item type="Lodging" expto="Sales">
        <Date>2003-01-01</Date>
        <expense>122.11</expense>
    </expense-item>
    <expense-item type="Lodging" expto="Development">
        <Date>2003-01-02</Date>
        <expense>122.12</expense>
        <description>Played penny arcade</description>
    </expense-item>
</expense-report>
```

Základní pojmy XML

- Element
  - Musí být uzavřený a správně vnořený
  - Platí case sensitivita
- Attribut
  - Hodnota musí být v uvozovkách
  - Musí být v elementu jedinečný
- Root element => v dokumentu může být jen jeden!!!!
- **Well – formed XML document** => splňuje XML pravidla
- **Valid XML document** => splňuje navíc pravidla dané specifikací dokumentu

XML technologie

- XSD
- XSLT
- XPath
- XQuery
- WSDL, SOAP
- OpenXML

DTD

- Vzniklo spolu s XML 1.0
- Popisuje XML dokument pomocí textového dokumentu splňujícího SGML
- http://www.w3.org/XML/1998/06/xmlspec-report-19980910.htm

```
<!DOCTYPE bookstore [
 <!ELEMENT bookstore (topic+)>
 <!ELEMENT topic (name,book*)>
 <!ELEMENT name (#PCDATA)>
 <!ELEMENT book (title,author)>
 <!ELEMENT title (#CDATA)>
 <!ELEMENT author (#CDATA)>
 <!ELEMENT isbn (#PCDATA)>
 <!ATTLIST book isbn CDATA "0">
 ]>
```

DTD - použití

- Klady
  - Starší systémy dodnes preferují DTD před jiným schématem
  - Velké použití hlavně v akademickém světě
  - Jednoduchost
- Zápory
  - Nepodporuje složitější omezení hodnot elementů a atributů
  - Nepodporuje namespace
  - DTD se dá použít pro DoS útok
  - .NET umožňuje ignorovat DTD při parsování XML

XSD - Historie

- Nedostatky DTD vedly ke snaze vytvořit nový jazyk popisující schémata XML dokumentu.
- Na základě XDR, DDML, DTD, SOX a W3C vydává v roce 2001 specifikaci XSD 1.0
- V dubnu 2012 vydává W3C doporučení XSD 1.1
- XSD je zapsáno jako XML, takže umožňuje definovat i sama sebe

XSD - použití

- Klady
  - Standard v nových aplikacích
  - Podpora omezení hodnot elementů mnoha způsoby
  - Plná podpora namespace
  - XQuery dokáže využít XSD pro optimalizaci dotazu do XML
- Zápory
  - Složitá syntaxe (specifikace má několik set stránek)
  - Omezená podpora pro obsah nevyžadující konkrétní pořadí elementů
  - Obsah a atributy nemůžou záviset na obsahu jiných elementů a atributů (1.0)

ALTOVA MissionKit

- XMLSpy – XML editor
- MapForce – grafické generování XSLT, nebo kódu pro konverzi XML
- StyleVision – generování reportů, PDF, Word z dat
- UModel – generování UML
- DatabaseSpy – nástroj na dotazy, generování a porovnávání DB
- DiffDog – porovnávání souborů, složek a databází
- SchemaAgent – udržuje vazby mezi XML dokumenty
- Authentic – vytváření elektronických formulářů pro XML a DB data
- www.altova.com
- Cena za komplet Pro 690 € nebo Enterprise 1.390 € (single licence)

Lab 1

- Najděte chyby v XML dokumentu „Lab1\Soubor1.docx“ a označte je. K nalezení chyb nepoužijte prosím žádnou aplikaci.

2. Základy XSD

- Ověření XML dokumentu pomocí XSD
- Vytvoření vlastního XSD
- Definování základních schémat
- Použití jednoduchých omezujících podmínek
- Generování XML z XSD

XSD – první dojmy

- XSD popisuje strukturu a omezení hodnot elementů a atributů
- XSD je modulární
- XSD globální komponenty – flat design
- Vazba na XML je
  - Přímo definovaná v XML
  - Nebo kontrolu řeší aplikace propojením schématu a XML

XSD – první vlastní XSD

- Vytvořme (společně) XSD k tomuto XML

```
<Knihovna>
    <Kniha ISBN= "123456" Jazyk="cs" >
        <Autori>
            <Autor Jmeno="Juraj Červenák" />
        </Autori>
        <Nazev>Ďáblova pevnost</Nazev>
        <Nakladatel>BROKILON</Nakladatel>
        <RokVydani>2011</RokVydani>
    </Kniha>
    <Kniha> … </Kniha>
</Knihovna>
```

Definování základních schémat

- Varianta s globálními elementy a referencemi
  - Kterýkoliv prvek může být root dokumentu
- Pevná stavba XML schématu s minimem referencí
- Zamyslete se nad výhodami/nevýhodami jednotlivých variant!
- Kdy se nehodí globální elementy?
- Kdy musíte použít globální elementy odkazované referencí?

Předdefinované datové typy

Zpracování mezer

- Whitespace mimo elementů se ignorují
- Whitespace v elementu se zpracovávají dle typu elementu
- xs:string => nic se neignoruje
- xs:normalizedString => pouze Tab, LF, CR se nahradí mezerou
- xs:token => vše se nahradí mezerami a více mezer jednou

Další typy odvozené z xs:token

- xs:language => může obsahovat jen string definující jazyk
  - en, en-US, cs-CZ, fr, fr-FR, …
- xs:NMTOKEN => tzv. Name token. String neobsahující mezery, čárky a další definované znaky
  - Maxipes, VAT, 1965-02-18, 12345678, …
- xs:Name => jako NMTOKEN, ale může začít písmenem, „:“ a „-“
  - Maxipes, VAT, -1965-02-18, :10-00, …
- xs:NCName => „noncolonized name“. Neobsahuje „:“

Speciální typy odvozené z xs:NCName

- xs:ID => nesmí mít duplicitu v XML dokumentu
- xs:IDREF => hodnota musí odpovídat některému ID v dokumentu
- xs:ENTITY => kvůli kompatibilitě s DTD ENTITY

Specializované typy

- xs:QName => podporuje použití namespace prefix
- xs:anyURI => očekává URI a podporuje převod znaků na URI standard (náhrada mezer, znaků s diakritikou, …)
- xs:NOTATION => pro zápis výčtu typů non-XML dat (nelze použít přímo)
- xs:hexBinary => umožní zadat binární data pomocí hexa zápisu
- xs:base64Binary => binární data kódované base64 dle RFC2045

Numerické typy odvozené z xs:decimal

- Desetinná čárka je vždy „.“
- Délka čísla je libovolná. Limitovaná je pouze pro xs:long, xs:int, xs:short a xs:byte a jejich unsigned verze
- Číslo nesmí obsahovat oddělovač tisíců, ani mezery

Reálná čísla

- xs:double => reálné číslo reprezentující IEEE double (64 bit)
- xs:float => reálné číslo reprezentující IEEE single (32 bit)
- Podporuje zápis s mocninou 10 pomocí E (e)
- Podporuje hodnoty INF, -INF, NaN, 0 a -0
- Pozor na nepřesnost při práci s reálnými čísly

Boolovská hodnota

- xs:boolean => podporuje hodnoty „true“ a „false“
- Podporuje i hodnoty 1 a 0

Datum a čas

- Zápis času vychází z ISO 8601
- xs:dateTime => bod v čase (2001-10-26T22:21:10+04:00)
- xs:date => definuje pouze den, může obsahovat i časové pásmo
- xs:gYearMonth => měsíc v roce dle Gregoriánského kalendáře
- xs:gYear => pouze rok (případně s časovým posunem)
- xs:time => definuje pouze bod pomocí času dne
- xs:gDay => opakující se den měsíce (---01, ---31, ---05+02:00)
- xs:gMonth => definuje měsíc (--05, --11Z, --08-04:00)
- xs:gMonthDay => opakující se den v měsíci (--01-05, --11-15Z)

Časový úsek

- xs:duration => definuje časový úsek speciálním zápisem
- PnYnMnDTnHnMnS
- Není vhodné míchat dny a měsíce, protože délka času u měsíců je proměnlivá
- P2Y30D
- -P1Y
- PT123456S

Seznamy

- xs:NMTOKENS => seznam hodnot v xs:NMTOKEN
- xs:IDREFS => seznam hodnot v xs:IDREF
- xs:ENTITIES => seznam hodnot v xs:ENTITY
- Prvky seznamů jsou oddělené pomocí whitespace

xs:anySimpleType

- Může obsahovat cokoliv
- Nejde ho použít pro vytvoření vlastního typu
- Není ani v Recommendation specifikaci! Proto je jeho použití nedoporučeno
- Používá se pouze ve specifických případech

Lab 2a

- Upravte první XSD popisující knihovnu a zpřesněte definici typů
- Přidejte knize
  - atribut definující ID knihy v knihovně
  - atribut definující zda je kniha zapůjčená
  - atribut počet stránek
  - atribut cena
  - element popis
  - nepovinný element odkaz na web stránku s ukázkou

Vytváření vlastních typů

- Restriction
- List
- Union

Použití restriction - 1

- ```
  <xs:simpleType name="myInteger">
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="-2"/>
      <xs:maxExclusive value="5"/>
    </xs:restriction>
  </xs:simpleType>
  ```
- ```
  <xs:simpleType name="schemaRecommendations">
    <xs:restriction base="xs:anyURI">
      <xs:enumeration value="http://www.w3.org/TR/xmlschema-0/"/>
      <xs:enumeration value="http://www.w3.org/TR/xmlschema-1/"/>
      <xs:enumeration value="http://www.w3.org/TR/xmlschema-2/"/>
    </xs:restriction>
  </xs:simpleType>
  ```

Použití restriction - 2

- xs:length => definuje pevnou délku řetězce nebo binárních dat
- xs:maxLength => definuje maximální délku řetězce nebo dat
- xs:minLength => definuje minimální délku řetězce nebo dat
- xs:pattern => definuje regulární výraz pro řetězec nebo reálné číslo. Pokud se použije vícenásobně, platí mezi omezeními OR
- xs:whiteSpace => definuje mechanizmus zpracování whitespace znaků

Restriction pro datum a čas

- xs:enumeration
- xs:maxExclusive, xs:maxInclusive
- xs:minExclusive, xs:minInclusive
- xs:pattern => reg. výraz definuje povolený zápis času

Restriction pro číselné typy

- Podporuje vše co jsme už použili
- xs:totalDigits => maximální počet číslic (ignoruje počáteční 0)
- xs:fractionDigits => maximální počet desetinných číslic (pouze pro typ xs:decimal)

Pravidla pro odvozování u restrikce

- Odvozením nelze zmenšit restrikci původního typu
- Vůbec nelze změnit restrikci xs:length
- Pokud použijete opakovaně xs:pattern při odvození nového typu, aplikuje se mezi požadavky AND

List

- Umožňuje definovat prvek, obsahující výčet hodnot oddělených pomocí whitespace znaků
- <xs:simpleType name="integerList">
  <xs:list itemType="xs:integer"/>
  </xs:simpleType>
- Omezení xs:length, xs:minLength a xs:maxLength se vztahují na počet prvků
- Nelze definovat seznam seznamů

Union

- Umožňuje definovat typ s různými datovými typy
- ```
  <xs:simpleType name="myIntegerUnion">
    <xs:union>
      <xs:simpleType>
        <xs:restriction base="xs:integer"/>
      </xs:simpleType>
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="undefined"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:union>
  </xs:simpleType>
  ```

Lab 2b

- Upravte definici knihovny podle těchto požadavků
  - Cena má maximálně 2 desetinná místa
  - ID knihy je ve formátu X-00000, kde X může být B, D, V, M, P, Y
  - Rok vydání má číselnou hodnotu nebo „neznámý“
  - Hodnota vydavatelství musí být pouze z povoleného výčtu
  - EXTRA: Poznámka nesmí mít víc než 10 slov

3. XML Namespace

- XML namespace
- Deklarace namespace a jeho použití
- Pravidla pro namespace
  ```

XML Namespace

- XML Namespace umožňuje rozlišit dva prvky stejného jména na různých místech XML dokumentu
- Namespace se dostal do XML rok po uvedení XML 1.0
- DTD nebylo nikdy adaptováno na použití namespace
- XSD obsahuje plnou podporu pro definování namespace pro všechny prvky (element, attribut, skupina, typ)

Defaultní namespace

- Je uveden pomocí atributu xmlns
- <?xml version="1.0"?>
<Kniha ISBN="978-80-7456-064-4" Jazyk="cs"
xmlns="http://www.gopas.cz/kurz/xmlxsd" >
<Autori>
<Autor Jmeno="Juraj Červenák"/>
</Autori>
<Nazev>Ďáblova pevnost</Nazev>
<Nakladatel>BROKILON</Nakladatel>
<RokVydani>2011</RokVydani>
<Popis>Třetí kniha příběhů kapitána Báthoryho</Popis>

<WebUkazka>http://www.xxxx.cz/kniha1.php</WebUkazka>
</Kniha>

Namespace pomocí prefixu

- Je uveden pomocí atributu xmlns:prefix
- <?xml version="1.0"?>
<kn:Kniha ISBN="978-80-7456-064-4" Jazyk="cs"
xmlns:kn="http://www.gopas.cz/kurz/xmlxsd" >
<kn:Autori>
<kn:Autor Jmeno="Juraj Červenák"/>
</kn:Autori>
<kn:Nazev>Ďáblova pevnost</kn:Nazev>
<kn:Nakladatel>BROKILON</kn:Nakladatel>
<kn:RokVydani>2011</kn:RokVydani>
<kn:Popis>Třetí kniha příběhů kapitána Báthoryho</kn:Popis>

<kn:WebUkazka>http://www.xxxx.cz/kniha1.php</kn:WebUkazka>
</kn:Kniha>

Použití více namespace

- V dokumentu může být definováno několik namespace
- Liší se prefixem a parser dokáže korektně hledat podle jména a namespace
- Je možno míchat defaultní a prefix namespace

Deklarace namespace v XSD

- Atribut targetNamespace v elementu xs:schema
- „qualified“ prvky závislé na namespace
- „unqualified“ prvky nezávislé na namespace
- Zda se uplatní na definované prvky určí atributy
  - elementFormDefault
  - attributeFormDefault
- Defaultní nastavení lze změnit atributem form v definici elementu nebo atributu

XML doporučení => Atributy neumísťovat do namespace

Import namespace

- Pokud chceme použít v našem schématu prvky z jiného schématu
- Import (umožňuje použít reference na globální prvky z jiného namespace)
  <xs:import namespace="http://www.gopas.cz/kurz/net1" schemaLocation=„net1base.xsd"/>
- Include (schéma bez targetNamespace, ale qualified prvky). Prvky se stanou součástí cílového namespace.
  <xs:include schemaLocation=„net2base.xsd"/>
- Redefine (umožňuje změnit typ z jiného schématu při zachování jména)

Preferujte Import před Include a Redefine

Jakýkoliv element z jiného namespace

- Občas je nutno zadat, že na daném místě bude nějaký element z jiného namespace
- Element
  <xs:any namespace="http://www.gopas.cz/kurz/net" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
- Atribut
  <xs:anyAttribute namespace="http://www.gopas.cz/kurz/net" processContents="skip"/>
- Do namespace lze zadat ##local, ##any, ##other, ##targetNamespace
- Do processContents lze zadat skip, strict, lax

Lab 3

- Doplňte do knihovny namespace
- Elementy Popis a WebUkazka přesuňte do jiného namespace
- Umožněte do elementu Kniha přidat jakýkoliv element z namespace http://www.gopas.cz/kurz/test2019

4. Komplexní typy v XSD

- Co to je komplexní typ
- Tvorba vlastních komplexních typů
- Typy omezení
- Použití hotových knihoven typů

Komplexní typ

- Simple type => definuje obsah a je nezávislý na zbytku XML
- Complex type => definuje celou strukturu
- Komplexní typy se vytvářejí zadefinováním, nebo odvozením z existujícího komplexního typu

Extension a Restriction

- Extension => může přidat nové atributy a elementy
- Restriction => může omezit samotná data elementu, jeho atributů a některé atributy výchozího typu zakázat

Simple Content - extension

- ```
  <xs:complexType name="tokenWithLang">
      <xs:simpleContent>
          <xs:extension base="xs:token">
              <xs:attribute ref="lang"/>
          </xs:extension>
      </xs:simpleContent>
  </xs:complexType>

  <xs:element name="title" type="tokenWithLang"/>
  ```
  ```

Simple Content - restriction

```
<xs:element name="title">
  <xs:complexType>
    <xs:simpleContent>
      <xs:restriction base="tokenWithLangAndNote">
        <xs:maxLength value="255"/>
        <xs:attribute name="lang"
type="xs:language"/>
        <xs:attribute name="note" use="prohibited"/>
      </xs:restriction>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

Komplexní obsah

- Sequence => pevně dané pořadí elementů
- Choice => výběr ze seznamu
- All => seznam elementů s libovolným pořadím

Pravidla pro vytváření komplexních typů

```
<xs:element name="Root">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element name="Element" minOccurs="1" maxOccurs="unbounded">
        <xs:complexType mixed="true">
          <xs:attribute name="First" use="required"/>
          <xs:attribute name="SomeAttribute" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="Element" minOccurs="0" maxOccurs="1">
        <xs:complexType mixed="true">
          <xs:attribute name="Second" use="required"/>
          <xs:attribute name="SomeOtherAttribute" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Omezení pro xs:all

- Nelze ho použít jako část jiného seznamu
- Samotný xs:all nesmí mít více výskytů než 1
- Žádný prvek seznamu nesmí mít více výskytů než 1
- Může obsahovat pouze elementy

- Cesty jak omezení obejít
  - Používat kontejnery pro vícenásobné elementy nebo skupiny
  - Nahradit xs:all pomocí xs:choice s neomezeným počtem výskytů

Extenze komplexních typů

- Sequence => bez problémů
- Choice => nelze rozšířit seznam choice, pouze přidat další prvky jako následnou sekvenci
- All => do typu definovaného xs:all nelze přidat další elementy, pouze atributy

Restriction pro komplexní typy

- Umožňuje změnit počet instancí jednotlivých prvků rodičovské struktury
- Instance odvozeného typu musí být validní i jako rodičovský typ!

Mixed content

- Používá se v případě, když element obsahuje text včetně podelementů
- Např. pokud obsah elementu může obsahovat HTML formátování

```
<Poznamka lang="cs">Autor <a href="http://wiki.org/xxxx">
Xxxx </a> patří k hlavním protagonistům <em> fantasy
</em> literatury v ČR </Poznamka>
```

```
<xs:complexType name= "formatovanyText" mixed="true">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="em" type="xs:token"/>
            <xs:element ref="a"/>
        </xs:choice>
        <xs:attribute ref="lang"/>
</xs:complexType>
```

Prázdný element

- Element, který má pouze atributy

```
<xs:element name="br">
   <xs:complexType>
       <xs:attribute name="id" type="xs:ID"/>
       <xs:attribute name="class" type="xs:NMTOKEN"/>
   </xs:complexType>
</xs:element>
```

- Nebo vytvořit simple typ s restrikcí

```
<xs:simpleType name="empty">
        <xs:restriction base="xs:string">
            <xs:enumeration value=""/>
        </xs:restriction>
</xs:simpleType>
```

Použití skupin

- Skupiny slouží k seskupení elementů nebo atributů
- Tam kde je to povoleno lze následně referencovat skupinu

Doporučení pro život

- Deklarujte elementy, skupiny atributů, skupiny elementů a simple typy
- Používejte XML namespace všude kde to má smysl
- Používejte pouze jednoduché komplexní typy
- Při lokální deklaraci pozor na „qualified“ elementy
- Schéma bez targetNamespace použijte výhradně k definici simple typů
- Preferujte jednoduchost před „XSD guru code“

Lab 4

- Upravte schéma knihovny tak, aby:
  - Atributy umístěte do skupin
  - Nastavte, že kniha má buď Popis nebo webový odkaz
  - Vytvořte typ KnihaType a pomocí extenze udělejte typ KnihaSHodnocenimType, který bude mít element pro zadání hodnocení číslem od 1 do 10

5. Unikátnost, reference

- ID/IDREFS
- xs:unique
- xs:key/xs:keyref

ID/IDREF

- Vychází z DTD ID a IDREF
- Odvozeno z xs:NCName => nemůže začínat číslicí a nesmí obsahovat mezeru
- ID a IDREF by se mělo používat pouze v atributech

xs:unique

- Validátor ověřuje jedinečnost hodnoty daného prvku
- Rozsah kontroly je dán umístěním podmínky
- Selector => určuje kde se bude hledat jedinečnost
- Field => definuje atribut, subelement, kterého hodnota musí být jedinečná
- Composite Fields

```
<xs:element name="library" maxOccurs="unbounded">
  <xs:complexType>
    .../...
  </xs:complexType>
  <xs:unique name="book">
    <xs:selector xpath="book"/>
    <xs:field xpath="isbn"/>
  </xs:unique>
</xs:element>
```

xs:key / xs:keyref

- xs:key se definuje stejně jako xs:unique, ale je povinné mít v něm hodnotu
- xs:keyref odkazuje na name xs:key

Lab 5

- Upravte schéma a vzorový XML soubor (bigKnihovna), který obsahuje seznam knih, seznam nakladatelství a seznam autorů tak, aby kontroloval zadané reference
  - Kniha pro autora a nakladatele obsahuje referenci do seznamu nakladatelů/autorů
- Zajistěte jedinečnost ISBN v dokumentu

6. Využití schémat a XSD 1.1

- Validace dokumentů pomocí schémat
- Konverze do čitelné podoby pomocí schémat
- Konverze mezi schématy
- Generování kódu pro konverze

Validace XML dokumentů

- Speciální aplikace
  - Většinou vyžaduje vazbu na konkrétní soubor
- Aplikace napsané v jazyce s XML/XSD knihovnou (C#, C++, Java, …)
  - Schéma lze načíst samostatně a XML soubor nemusí na schéma odkazovat
  - Schéma lze udržet v paměti pro více kontrol
- Jedno schéma může validovat XML s různým root elementem

Konverze do čitelné podoby

- Pomocí XSLT lze generovat jakékoliv dokumenty z XML
- Při tvorbě XSLT je vhodné mít k dispozici XSD
- Altova StyleVision umožňuje generovat HTML, RTF, PDF, Word 2007

Konverze z XML na XML

- Opět pomocí XSLT
- Při práci s XSLT se investice do komerčních produktů vrátí ve zvýšené produktivitě
- Altova MapForce

XSD 1.1

- XSD 1.1 přineslo několik významných změn
  - Assert
  - Alternative
  - Podpora verzí
  - Open content
- .NET Framework zatím nepodporuje XSD 1.1, je nutno použít knihovny třetích stran

Lab 5 – jen kdo chce

- Zkuste pomocí Altova MapPath vygenerovat XSLT pro konverzi z XML do XML (například mezi verzemi knihoven)
- Zkuste pomocí Altova StyleVision vygenerovat HTML dokument obsahující data o knihách
- Vytvořte schéma verze 1.1 a využijte alternate pro různý obsah elementu na základě hodnoty atributu, nebo použijte assert pro vazbu mezi hodnotami

Děkuji za účast

tomas@havetta.cz