PRG036 – Technologie XML

Přednáší:

Irena Mlýnková (mlynkova@ksi.mff.cuni.cz)

Martin Nečaský (necasky@ksi.mff.cuni.cz)

LS 2011

Stránka přednášky:
http://www.ksi.mff.cuni.cz/~mlynkova/prg036/

Osnova předmětu

- Úvod do principů formátu XML, přehled XML technologií, jazyk DTD
- Datové modely XML, rozhraní DOM a SAX
- Úvod do jazyka XPath
- Úvod do jazyka XSLT
- XPath 2.0, XSLT 2.0
- Úvod do jazyka XML Schema
- Pokročilé rysy jazyka XML Schema
- Přehled standardních XML formátů
- Úvod do jazyka XQuery
- Pokročilé rysy jazyka XQuery, XQuery Update
- Úvod do XML databází, nativní XML databáze, číslovací schémata, structural join
- Relační databáze s XML rozšířením, SQL/XML

Jmenné prostory

Jmenné prostory

- Problém: Nutnost rozlišení elementů a atributů se shodnými jmény v případech, kdy by mohlo dojít ke konfliktům
  - Aplikace potřebuje vědět, které elementy/atributy má zpracovat
  - př. název knihy vs. název firmy
- Myšlenka: Expandované jméno elementu/atributu = jmenný prostor + lokální jméno
  - Jmenný prostor se identifikuje pomocí URI
  - Obvykle se deklaruje pomocí DTD nebo XML Schema (častější)
- URI je příliš dlouhé → zkrácený zápis
  - Deklarace jmenného prostoru = prefix + URI
  - Kvalifikované jméno (QName) = prefix + lokální jméno elementu/atributu

Př. jmenné prostory

Deklarace jmenného prostoru → oblast platnosti

```
<cenik:nabidka
    xmlns:cenik="http://www.ecena.cz/e-cenik"
    xmlns:bib="http://www.book.org/bibliography">
  <cenik:polozka cenik:dph="22%">
    <cenik:nazev>
      <bib:book>
        <bib:author>Jiri Kosek</bib:author>
        <bib:title>HTML - tvorba dokonalych WWW
stranek</bib:title>
      </bib:book>
    </cenik:nazev>
    <cenik:cena mena="CZK">259</cenik:cena>
  </cenik:polozka>
</cenik:nabidka>
```

Použití jmenného prostoru

Př. implicitní jmenný prostor

```
<nabidka
    xmlns="http://www.ecena.cz/e-cenik"
    xmlns:bib="http://www.book.org/bibliography">
  <polozka cenik:dph="22%">
    <nazev>
      <bib:book>
        <bib:author>Jiri Kosek</bib:author>
        <bib:title>HTML - tvorba dokonalych WWW
stranek</bib:title>
      </bib:book>
    </nazev>
    <cena mena="CZK">259</cena>
  </polozka>
</nabidka>
```

Jmenný prostor

- Kolekce identifikátorů neobsahující duplikace
- Jmenný prostor sestává z disjunktních množin:
  - Oddíl jmen elementů (all element partition)
    - Unikátní jméno je dáno identifikátorem prostoru a jménem elementu
  - Oddíl globálních atributů (global attribute partition)
    - Unikátní jméno je dáno identifikátorem prostoru a jménem atributu
      - Lze deklarovat v XML Schema
  - Oddíl za každý element (per element type partitions)
    - Unikátní jméno je dáno identifikátorem prostoru, jménem elementu a lokálním jménem atributu

Př. části jmenných prostorů

```
<nabidka
    xmlns="http://www.ecena.cz/e-cenik"
    xmlns:bib="http://www.book.org/bibliography">
  <polozka cenik:dph="22%">
    <nazev>
      <bib:book>
        <bib:author>Jiri Kosek</bib:author>
        <bib:title xml:lang="cs">HTML - tvorba
dokonalych WWW stranek</bib:title>
      </bib:book>
    </nazev>
    <cena mena="CZK">259</cena>
  </polozka>
</nabidka>
```

Jmenný prostor XML

- Každý XML dokument má přiřazený jmenný prostor XML
  - URI: http://www.w3.org/XML/1998/namespace
  - Prefix: xml
  - Není třeba jej deklarovat
- Obsahuje globální atributy:
  - xml:lang – jazyk obsahu elementu
    - Hodnoty jsou dány normou
  - xml:space – zpracování bílých znaků aplikací
    - preserve = zachovat
    - default = použít nastavení aplikace
      - Obvykle nahradí vícenásobné mezery jednou
  - xml:id – jednoznačný identifikátor (typu ID)
  - xml:base – určení základního URI, další relativně k němu

Více o jmenných prostorech

- W3C specifikace:
  - http://www.w3.org/TR/REC-xml-names/
- Přednášky o XML Schema

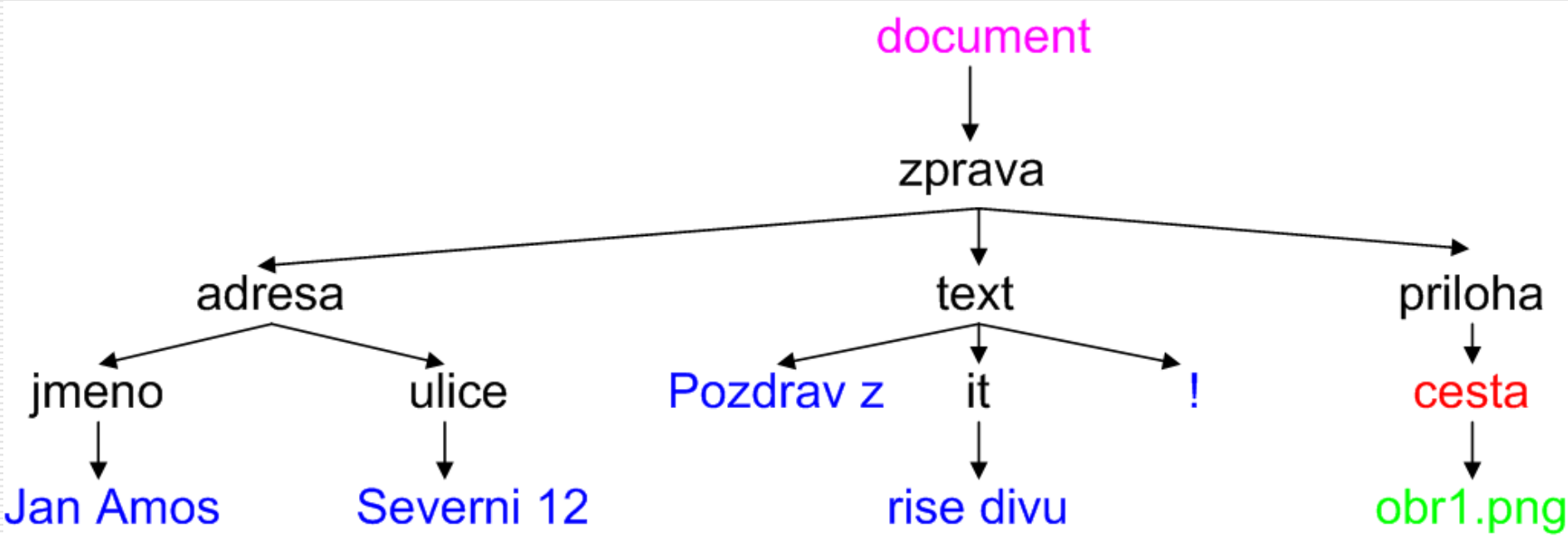
Datový model XML

XML Infoset

- Správně strukturovaný XML dokument → hierarchická stromová struktura = XML Infoset
  - Abstraktní datový model XML dat
- Information set = množina informací
- Information item = uzel stromu
  - Typy položek: Dokument, element, atribut, řetězec znaků, instrukce pro zpracování, komentář, notace, DTD deklarace, …
  - Vlastnosti položek: jméno, rodiče, potomci, obsah, …
- Je využíván v dalších standardech
- DTD může „modifikovat“ infoset
  - Př. defaultní hodnoty atributů

Př.

```
<zprava>
  <adresa>
    <jmeno>Jan Amos</jmeno>
    <ulice>Severni 12</ulice>
  </adresa>
  <text>Pozdrav z <it>rise divu</it>!</text>
  <priloha cesta="obr1.png"/>
</zprava>
```



Př. Element Information Item

- [namespace name]
  - (Případně prázdný) název jmenného prostoru
- [local name]
  - Lokální část jména elementu
- [prefix]
  - (Případně prázdný) prefix jmenného prostoru
- [children]
  - (Případně prázdný) uspořádaný seznam dětských položek
    - Document order
    - Elementy, instrukce pro zpracování, neexpandované reference na entity, znakové řetězce a komentáře
- [attributes]
  - (Případně prázdná) neuspořádaná množina atributů (Attribute Information Items)
    - Deklarace jmenných prostorů zde nejsou
  - Každá položka (atribut) je deklarovaná nebo určená schématem (defaultní atributy)

Př. Element Information Item

- [namespace attributes]
  - (Případně prázdná) neuspořádaná množina deklarací jmenných prostorů
  - Každá položka (atribut) je deklarovaná nebo určená schématem (defaultní atributy)
- [in-scope namespaces]
  - Neuspořádaní množina jmenných prostorů, které jsou platné pro daný element
  - Vždy obsahuje jmenný prostor XML
  - Vždy obsahuje prvky množiny [namespace attributes]
- [base URI]
  - URI elementu
- [parent]
  - Document/Element Information Item v jehož vlastnosti [children] je obsažen

- Další položky viz http://www.w3.org/TR/xml-infoset/

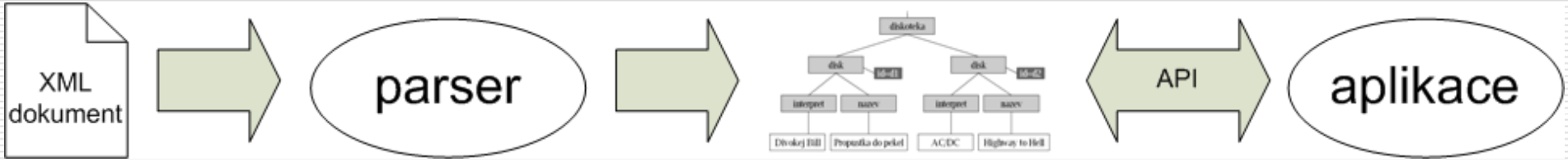
Post Schema Validation Infoset (PSVI)

- Otypovaný infoset
- Vznikne přiřazením datových typů na základě validace oproti schématu
  - Můžeme pracovat přímo s typovými hodnotami
  - Bez PSVI máme jen textové řetězce
    - DTD: minimum datových typů
    - XML Schema: int, long, bzte, date, time, boolean, positiveInteger, …
- Využití: v dotazovacích jazycích (XQuery, XPath)
  - Např. máme funkce specifické pro řetězce, čísla, datumy, ….

XML parsery

XML parsery

- Problém: XML data chceme zpracovávat
  - Programově načítat
- XML dokument = textový dokument → můžeme načítat jako text se závorkami
  - Pracné, nepohodlné, neefektivní
- Řešení: Při zpracování nás zajímá co je element, atribut, text, komentář, … → zajímá nás Infoset XML dokumentu
- Parser = SW, který umí přes vhodné rozhraní poskytovat infoset dokumentu aplikaci

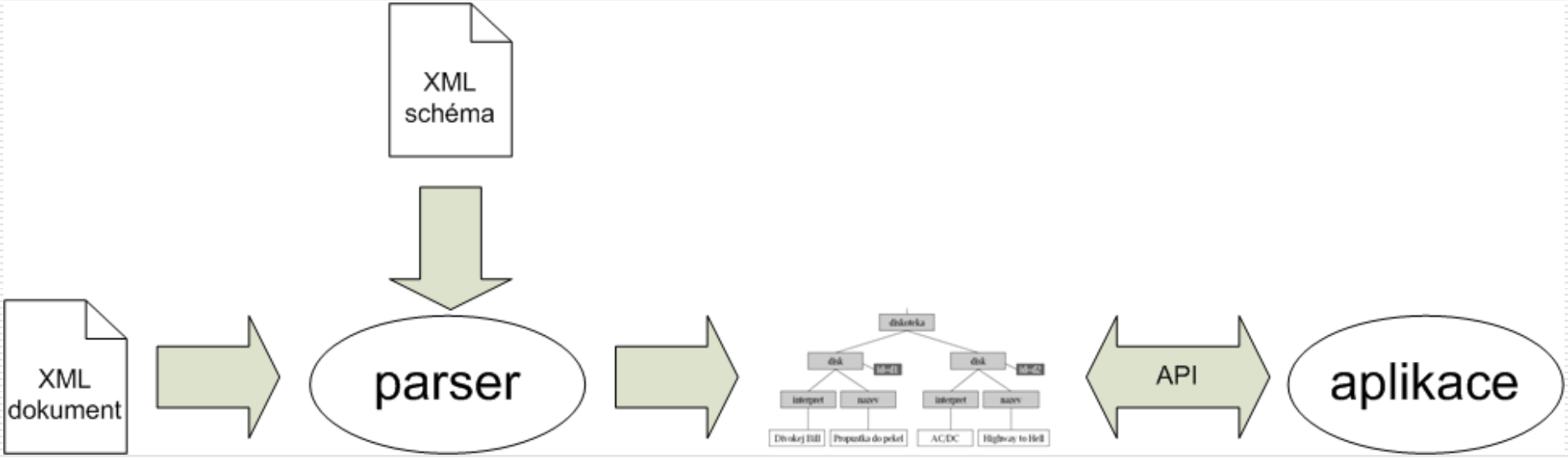


Typy parserů (1)

- Sekvenční
  - Rychlé, paměťově nenáročné
  - Jeden lineární průchod
- Stromová reprezentace
  - Celý dokument je načten v paměti
  - Opakovaný a nesekvenční průchod
  - Paměťově náročné, neefektivní
- Push vs. pull parser
  - Proud událostí vs. čtení na vyžádání

Typy parserů (2)

- Validující × nevalidující
  - Umí kontrolovat validitu dokumentu oproti schématu

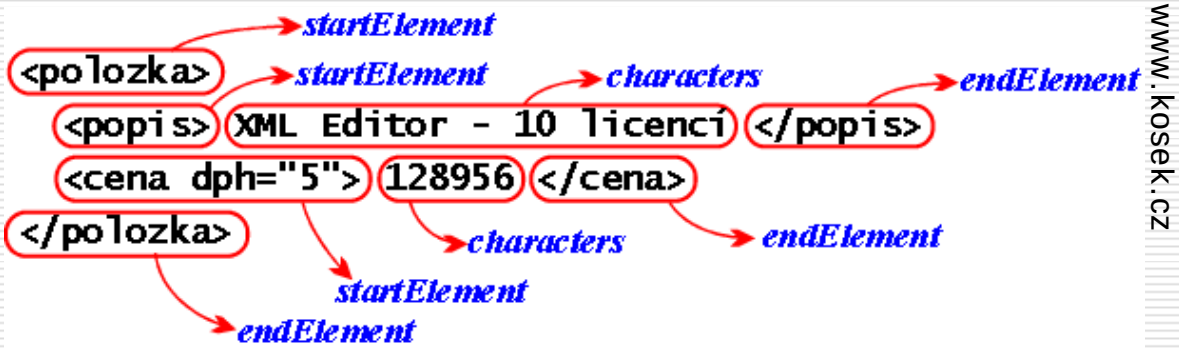


- S/bez podpory PSVI

Rozhraní SAX

SAX

- SAX = Simple API for XML
- Načtení části dokumentu = událost
  - Můžeme definovat obsluhu
- Hlavní události:



  - Atributy jsou součástí parametrů startElement ()

Java: Interface ContentHandler

- void startDocument ()
- void endDocument ()
- void startElement (String uri, String localName, String qName, Attributes atts)
- void endElement (String uri, String localName, String qName)
- void characters (char[] ch, int start, int length)
- void processingInstruction (String target, String data)
- void ignorableWhitespace (char[] ch, int start, int length)
- void startPrefixMapping (String prefix, String uri)
- void endPrefixMapping (String prefix)
- void skippedEntity (String name)
- void setDocumentLocator (Locator locator)

ContentHandler: startElement ()

- String uri
  - URI jmenného prostoru
- String localName
  - Lokální jméno
- String qName
  - Kvalifikované jméno
- Attributes atts

```
for (int i = 0; i < atts.getLength (); i++ ) {
  System.out.println (atts.getQName (i));
  System.out.println (atts.getValue (i));
  }
```

Interface Attributes (1)

- int getLength ()
  - Počet atributů v seznamu
- int getIndex (String qName)
  - Index atributu s daným (kvalifikovaným) jménem
- int getIndex (String uri, String localName)
  - Index atributu s daným lokálním jménem a URI jmenného prostoru
- String getLocalName (int index)
  - Lokální jméno atributu s daným indexem
- String getQName (int index)
  - Kvalifikované jméno atributu s daným indexem
- String getURI (int index)
  - URI atributu s daným indexem

Interface Attributes (2)

```
CDATA
ID
IDREF
IDREFS
NMTOKEN
NMTOKENS
ENTITY
ENTITIES
NOTATION
```

- String getType (int index)
  - Typ atributu s daným indexem
- String getType (String qName)
  - Typ atributu s daným (kvalifikovaným) jménem
- String getType (String uri, String localName)
  - Typ atributu s daným lokálním jménem a URI jmenného prostoru
- String getValue (int index)
  - Hodnota atributu s daným indexem
- String getValue (String qName)
  - Hodnota atributu s daným (kvalifikovaným) jménem
- String getValue (String uri, String localName)
  - Hodnota atributu s daným lokálním jménem a URI jmenného prostoru

ContentHandler: characters ()

- SAX parser může znaková data dávkovat libovolně → nelze počítat s tím, že je celý text doručen v rámci jednoho volání
- char[] ch
  - Pole v němž jsou znaky uloženy
- int start
  - Počáteční pozice znaků
- int length
  - Počet znaků

ContentHandler: ignorableWhitespace ()

- Ignorované bílé znaky
- char[] ch
  - Pole v němž jsou znaky uloženy
- int start
  - Počáteční pozice znaků
- int length
  - Počet znaků

ContentHandler: setDocumentLocator ()

```
class MujContentHandler implements ContentHandler {
  Locator locator;

  public void setDocumentLocator (Locator locator) {
    this.locator = locator;
    }
  ...
```

- Zacílení místa v dokumentu, kde vznikla událost
- Interface Locator
  - int getColumnNumber () – číslo sloupce
  - int getLineNumber () – číslo řádku
  - String getPublicId () – veřejný identifikátor dokumentu
  - String getSystemId () – systémový identifikátor dokumentu

Inicializace zpracování

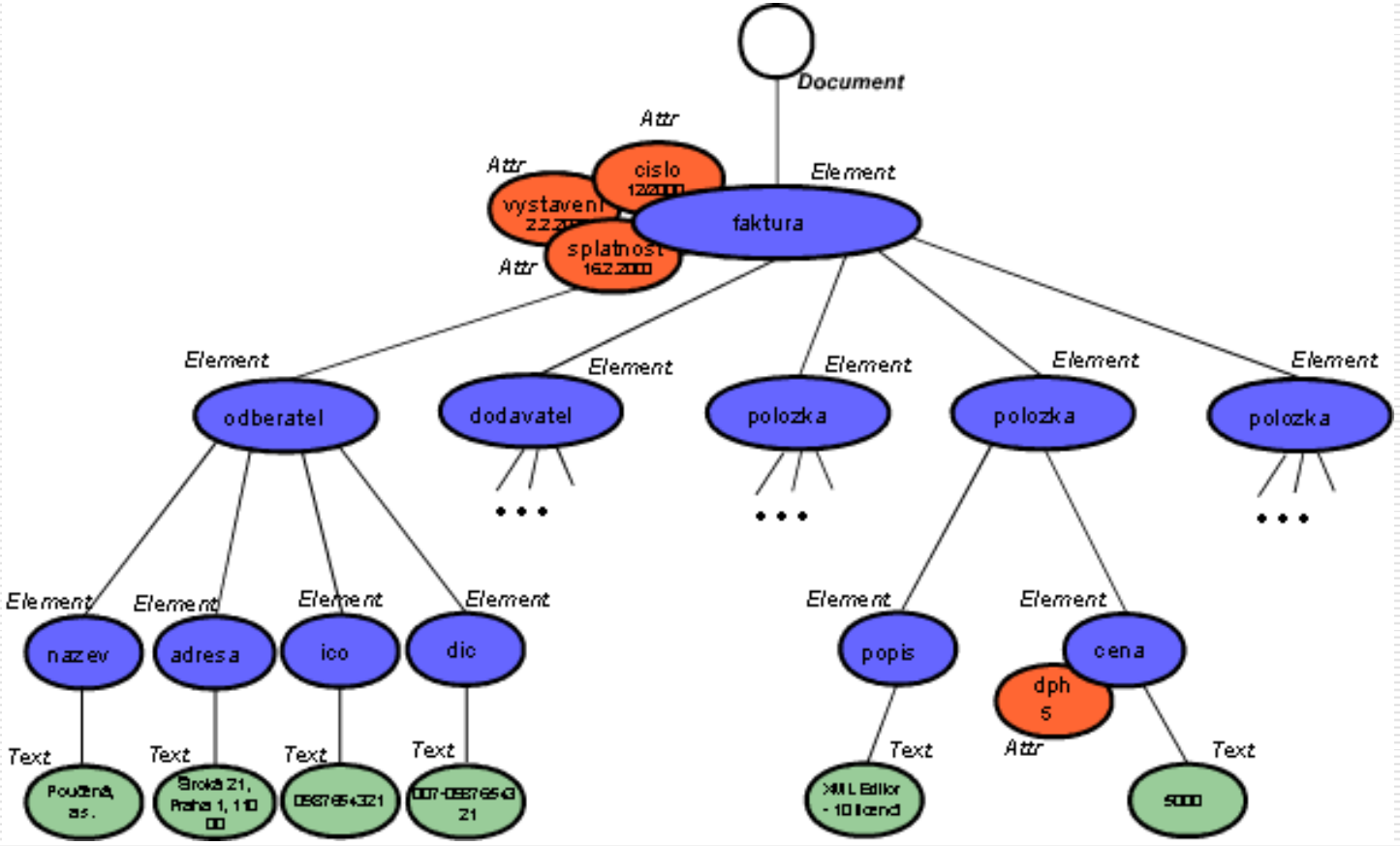
```
// Vytvorime instanci parseru
XMLReader parser = XMLReaderFactory.createXMLReader ();

// Vytvorime vstupni proud XML dat
InputSource source = new InputSource ("mujDokument.xml");

// Nastavime vlastni content handler pro obsluhu udalosti
parser.setContentHandler (new MujContentHandler ());

// Zpracujeme vstupni proud XML dat
parser.parse (source);
```

Rozhraní DOM

DOM

- DOM = Document Object Model
- Standard W3C
  - Verze: DOM1, DOM2 a DOM3
  - http://www.w3.org/DOM/DOMTR
- Celý dokument je načten do paměti
- Stromová reprezentace
- Jednotlivé uzly stromu jsou reprezentovány objekty
  - Dokument, fragment dokumentu, deklarace DTD, element, atribut, text, sekce CDATA, komentář, entita, reference na entitu, notace, instrukce pro zpracování
  - Metody objektů jsou dány specifikací DOM
  - Potomci objektů jsou dány XML Infosetem

Př. DOM strom



Java: Vybudování DOM stromu

```
// DocumentBuilderFactory vytvari DOM parsery
DocumentBuilderFactory dbf =
  documentBuilderFactory.newInstance ();

// nebudeme validovat
dbf.setValidating (false);

// vytvorime si DOM parser
DocumentBuilder builder =
  dbf.newDocumentBuilder ("mujDokument.xml");

// parser zpracuje soubor a vytvori strom DOM objektu
Document doc = builder.parse ();

// zpracujeme DOM strom
processTree (doc);
```

Document doc

Java: Uložení DOM stromu

```
// TransformerFactory vytvari serializatory DOM stromu
TransformerFactory tf = TransformerFactory.newInstance ();

// Transformer serializuje DOM stromy
Transformer writer = tf.newTransformer ();

// nastavime kodovani
writer.setOutputProperty
  (OutputKeys.ENCODING, "windows-1250");

// spustime transformaci DOM stromu do XML dokumentu
writer.transform
  (new DOMSource (doc),
   new StreamResult (new File ("vystupniDokument.xml")));
```

Třídy v Javě (1)

- Node – základ pro další rozhraní reprezentující uzly DOM stromu:
  - Document – potomci: Element (maximálně jeden), ProcessingInstruction, Comment, DocumentType (maximálně jeden)
  - DocumentFragment – potomci: Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
  - Element – potomci: Element, Text, Comment, ProcessingInstruction, CDATASection, EntityReference
  - Attr – potomci: Text, EntityReference
  - Text – bez potomků
  - CharacterData – bez potomků

Třídy v Javě (2)

- ProcessingInstruction – bez potomků
- Comment – bez potomků
- CDATASection – bez potomků
- Entity – potomci: Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
- EntityReference – potomci: Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
- Notation – bez potomků
- DocumentType – bez potomků

Interface Node (1)

- String getNodeName ()
- short getNodeType ()
- String getNodeValue ()

- String getBaseURI ()
- String getPrefix ()
- void setPrefix (String prefix)
- String getLocalName ()
- String getNamespaceURI ()
- String lookupNamespaceURI (String prefix)
- String lookupPrefix (String namespaceURI)
- boolean isDefaultNamespace (String namespaceURI)

- boolean hasAttributes ()
- boolean hasChildNodes ()

Interface Node (2)

- Node getParentNode ()
- Node getPreviousSibling ()
- Node getNextSibling ()
- NodeList getChildNodes ()
- Node getFirstChild ()
- Node getLastChild ()

- NamedNodeMap getAttributes ()
- String getTextContent ()
- Document getOwnerDocument ()
  - Vrací Document asociovaný s uzlem

- Node removeChild (Node oldChild)
- Node replaceChild (Node newChild, Node oldChild)
- Node appendChild (Node newChild)
- Node insertBefore (Node newChild, Node refChild)

Interface Node (3)

- Node cloneNode (boolean deep)
- void setNodeValue (String nodeValue)
- void setTextContent (String textContent)

- void normalize ()
  - Normalizuje všechny textové poduzly, tj. sloučí sousední textové uzly a eliminuje prázdné

- boolean isEqualNode (Node arg)
- boolean isSameNode (Node other)
- short compareDocumentPosition (Node other)
  - Porovná pozice uzlů v dokumentu

Interface Node (4)

- Object getUserData (String key)
  - Vrací objekt s daným klíčem asociovaný s uzlem
- Object setUserData (String key, Object data, UserDataHandler handler)
  - Asociuje objekt s daným klíčem s uzlem

- Object getFeature (String feature, String version)
  - Vrací objekt implementující danou verzi dané vlastnosti
- boolean isSupported (String feature, String version)
  - Testuje jestli implementace podporuje danou verzi dané vlastnosti

Interface	nodeName	nodeValue	attributes
Attr	jako Attr.name	jako Attr.value	null
CDATA-Section	"#cdata-section"	jako CharacterData.data, obsah sekce CDATA	null
Comment	"#comment"	jako CharacterData.data, obsah komentáře	null
Document	"#document"	null	null
Document-Fragment	"#document-fragment"	null	null
Document-Type	jako DocumentType.name	null	null
Element	jako Element.tagName	null	Named-NodeMap
Entity	jméno entity	null	null
Entity-Reference	jméno referencované entity	null	null
Notation	jméno notace	null	null
Processing-Instruction	jako ProcessingInstruction.target	jako ProcessingInstruction.data	null
Text	"#text"	jako CharacterData.data, obsah textového uzlu	null

Př. dětské uzly vs. atributy

```
for (Node child = n.getFirstChild();
     child != null;
     child = child.getNextSibling())
  {
  processChildNode(child);
  }
```

```
NamedNodeMap atts = n.getAttributes();
for (int i = 0; i < atts.getLength(); i++)
  {
  Node att = atts.item(i);
  processAttribute(att);
  }
```

Interface Document (1)

- Attr createAttribute (String name)
- Attr createAttributeNS (String namespaceURI, String qualifiedName)
- CDATASection createCDATASection (String data)
- Comment createComment (String data)
- DocumentFragment createDocumentFragment ()
- Element createElement (String tagName)
- Element createElementNS (String namespaceURI, String qualifiedName)
- EntityReference createEntityReference (String name)
- ProcessingInstruction createProcessingInstruction (String target, String data)
- Text createTextNode (String data)

- Element getElementById (String elementId)
  - Vrací element s danou hodnotou atributu typu ID
- NodeList getElementsByTagName (String tagname)
- NodeList getElementsByTagNameNS (String namespaceURI, String localName)

Interface Document (2)

- Element getDocumentElement ()
- DocumentType getDoctype ()

- Node renameNode (Node n, String namespaceURI, String qualifiedName)
- Node adoptNode (Node source)
  - Připojí daný uzel do aktuálního dokumentu
- Node importNode (Node importedNode, boolean deep)
  - Importuje daný uzel do aktuálního dokumentu, tj. vytvoří jeho kopii

- String getInputEncoding ()
  - Vrací kódování použité při parsingu
- String getXmlEncoding ()
- DOMImplementation getImplementation ()
  - Vrací implementaci (DOMImplementation), k níž přísluší dokument
- DOMConfiguration getDomConfig ()
  - Vrací konfiguraci pro normalizaci různých typů uzlů

Interface Document (3)

- boolean getXmlStandalone ()
- String getXmlVersion ()
- String getDocumentURI ()
- void setXmlStandalone (boolean xmlStandalone)
- void setXmlVersion (String xmlVersion)
- void setDocumentURI (String documentURI)

- void normalizeDocument ()
  - Normalizuje XML dokument, tj. nahradí všechny referecne na entity a normalizuje textové hodnoty

- boolean getStrictErrorChecking ()
  - Zjišťuje zda je kontrola chyb daná specifikací nebo závislá na implementaci
- void setStrictErrorChecking (boolean strictErrorChecking)
  - Určuje zda je kontrola chyb daná specifikací nebo závislá na implementaci

Interface Element (1)

- String getTagName ()
  - Jméno elementu
- NodeList getElementsByTagName (String name)
  - Seznam (NodeList) všech potomků daného jména
- NodeList getElementsByTagNameNS (String namespaceURI, String localName)
  - Seznam (NodeList) všech potomků daného lokálního jména a URI
- String getAttribute (String name)
  - Hodnota atributu daného jména
- Attr getAttributeNode (String name)
  - Atributový uzel daného jména
- Attr getAttributeNodeNS (String namespaceURI, String localName)
  - Atributový uzel daného lokálního jména a URI
- String getAttributeNS (String namespaceURI, String localName)
  - Hodnota atributu daného lokálního jména a URI

Interface Element (2)

- boolean hasAttribute (String name)
  - true = element má atribut daného jména
- boolean hasAttributeNS (String namespaceURI, String localName)
  - true = element má atribut daného lokálního jména a URI

- void removeAttribute (String name)
  - Odstraní atribut daného jména
- Attr removeAttributeNode (Attr oldAttr)
  - Odstraní atributový uzel
- void removeAttributeNS (String namespaceURI, String localName)
  - Odstraní atribut daného lokálního jména a URI

- TypeInfo getSchemaTypeInfo ()
  - Typová informace pro daný element

Interface Element (3)

- void setAttribute (String name, String value)
  - Přidá nový atribut
- Attr setAttributeNode (Attr newAttr)
  - Přidá nový atribut
- Attr setAttributeNodeNS (Attr newAttr)
  - Přidá nový atribut
- void setAttributeNS (String namespaceURI, String qualifiedName, String value)
  - Přidá nový atribut

- void setIdAttribute (String name, boolean isId)
  - Zmena datového typu atributu z/na ID
- void setIdAttributeNode (Attr idAttr, boolean isId)
  - Zmena datového typu atributu z/na ID
- void setIdAttributeNS (String namespaceURI, String localName, boolean isId)
  - Zmena datového typu atributu z/na ID

Př. vytvoření elementu

```
public Node createEmployee(Document document) {
  Element firstName = document.createElement("FirstName");
  firstName.appendChild(document.createTextNode("Shawn"));

  Element lastName = document.createElement("LastName");
  lastName.appendChild(document.createTextNode("Michaels"));

  Attr genderAttribute = document.createAttribute("gender");
  genderAttribute.setValue("M");

  Element employee = document.createElement("Employee");
  employee.setAttributeNode(genderAttribute);
  employee.appendChild(firstName);
  employee.appendChild(lastName);

  return employee;
  }
```

Interface Attr

```
NamedNodeMap attrs =
  node.getAttributes();
Attr attr = (Attr)attrs.item(0);
System.out.print(
  attr.getNodeName() + "=\"" +
  attr.getNodeValue() + "\"");
```

- String getName ()
  - Název atributu
- String getValue ()
  - Hodnota atributu
- void setValue (String value)
  - Nastaví hodnotu atributu
- Element getOwnerElement ()
  - Elementový uzel, k němuž atribut přísluší
- TypeInfo getSchemaTypeInfo ()
  - Informace o typu atributu
- boolean getSpecified ()
  - true = atribut byl explicitně uveden v dokumentu
- boolean isId ()
  - true = atribut je typu ID

Interface CharacterData

- String getData ()
  - Aktuální znaková data
- int getLength ()
  - Délka znakových dat
- String substringData (int offset, int count)
  - Podčást aktuálních dat
- void setData (String data)
  - Nastaví aktuální znaková data
- void insertData (int offset, String arg)
  - Přidá daný text na danou pozici
- void appendData (String arg)
  - Přidá daný text na konec
- void deleteData (int offset, int count)
  - Odstraní text v daném rozsahu
- void replaceData (int offset, int count, String arg)
  - Nahradí text v daném rozsahu

Interface Text

- Metody CharacterData
- String getWholeText ()
  - Textový obsah všech logicky sousedících textových uzlů daného uzlu spojený v jeden celek
- Text replaceWholeText (String content)
  - Nahradí textový obsah všech logicky sousedících textových uzlů daného uzlu
- boolean isElementContentWhitespace ()
  - true = textový uzel obsahuje nevýznamné bílé znaky
- Text splitText (int offset)
  - Na daném místě rozdělí textový uzel do dvou.

Interface ProcessingInstruction

- String getData ()
  - Textový obsah PI
- void setData (String data)
  - Nastavení textového obsahu PI
- String getTarget ()
  - Cíl PI

Interface Notation

- String getPublicId ()
  - Veřejný identifikátor notace
- String getSystemId ()
  - Systémový identifikátor notace

Interface Entity

- String getNotationName ()
  - Jméno notace pro neparsované entity
- String getPublicId ()
  - Veřejný identifikátor entity
- String getSystemId ()
  - Systémový identifikátor entity
- String getXmlVersion ()
  - Specifikovaná verze externí entity
- String getXmlEncoding ()
  - Specifikované kódování externí entity
- String getInputEncoding ()
  - Kódování použité pro externí entitu při parsingu

Interface DocumentType

- String getName ()
  - Jméno kořenového elementu DTD
- String getPublicId ()
  - Veřejný identifikátor DTD
- String getSystemId ()
  - Systémový identifikátor DTD
- String getInternalSubset ()
  - Definice DTD ve formě textového řetězce
- NamedNodeMap getEntities ()
  - Seznam deklarovaných entit
- NamedNodeMap getNotations ()
  - Seznam deklarovaných notací

Další rozhraní

- Interface DocumentFragment
  - Pouze metody Node
- Interface EntityReference
  - Pouze metody Node
- Interface CDATASection
  - Metody Node, Text a CharacterData
- Interface Comment
  - Metody Node a CharacterData

Konec