XML APIs

DOM, Document Object Model
SAX, Simple API for XML

XML APIs

- základné informácie
- DOM
  - reprezentácia stromu
  - prvky stromu
  - IDL špecifikácia
  - operácie nad stromom
- SAX
  - zverejňované udalosti
  - špecifikácia
  - implementácia

Základné informácie

API zabezpečujúce prístup k informáciám v XML dokumentoch

Štandardizované rozhrania pre XML parsre

Dve základné techniky:

- DOM - reprezentácia celého XML stromu
  (okamžitý prístup ku všetkým prvkom)
- SAX - udalosti parsovania textového streamu
  (začiatok / koniec elementu, …)

Základné informácie - porovnanie DOM a SAX

DOM

štandard W3C
existujúce implementácie
objektová reprezentácia stromu
vytvorenie vs. operácie nad celým dokumentom

SAX

de-facto štandard
aplikačne špecifické implementácie
reprezentácia udalosťami
priebežný prístup k informáciám

pamäťové vs. časové nároky

DOM, Document Object Model

Štandard W3C, http://www.w3.org/DOM

Platformovo a jazykovo neutrálne rozhranie, umožňujúce dynamický prístup a úpravy obsahu, štruktúry a vzhľadu dokumentu

(prístup k celému dokumentu)

DOM podpora v XML parseroch

JAXP, Java API for XML Parsing

Sun Microsystems, java.sun.com/xml, Java 1.4

Xerces, Java/C++/Perl

Apache Foundation, xml.apache.org

msxml

Microsoft, msdn.microsoft.com/xml

4DOM, Python

fourthought.com/4Suite/4DOM

… a mnohé ďalšie

XML dokument v objektovej reprezentacii

```
<?xml version="1.0"?>
<doc name="first">
     <!-- text -->
     <autor>
        <meno>Roman</meno>
        <priez>F</priez>
     </autor>
     ...
</doc>
```

Strom, uzly (nodes):

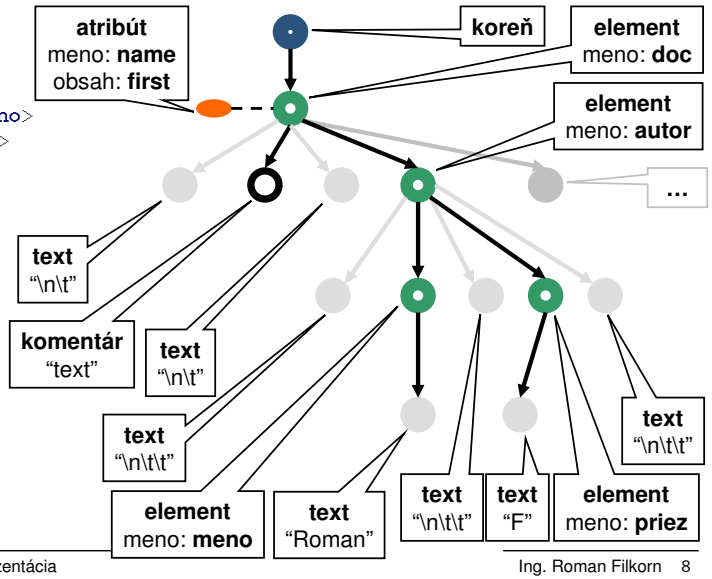
- koreň
- element
- text
- atribút
- namespace
- PI
- komentár

XML dokument v objektovej reprezentacii

```
<?xml version="1.0"?>
<doc name="first">
     <!-- text -->
     <autor>
        <meno>Roman</meno>
        <priez>F</priez>
     </autor>
     ...
</doc>
```

uzly:

- koreň
- element
- text
- atribút
- namespace
- PI
- komentár



DOM komponenty

Document
- top-level komponent
- operácie nad stromom
  - (vytváranie uzlov, vyhľadávanie uzlov)

Node
- uzol stromu
- základ dedenia pre špecifické prvky stromu

NodeList
- návratová hodnota
  - (vyhľadávanie uzlov, atribúty, …)

DOM komponenty - hierarchia vybraných tried

Node
- Element
- Attr
- ProcessingInstruction
- CharacterData
  - Text
    - CDATASection
  - Comment

Prejdeme si ich vo forme IDL špecifikácií

DOM komponenty - Node

Základný typ pre celú DOM reprezentáciu

IDL špecifikácia:
- označenie typu
- meno, hodnota
- operácie s deťmi a atribúty
- rozšírenia (namespace, features, …)

DOM komponenty - Node

```
interface Node {

  // NodeType
  const unsigned short      ELEMENT_NODE                = 1;
  const unsigned short      ATTRIBUTE_NODE              = 2;
  const unsigned short      TEXT_NODE                   = 3;
  const unsigned short      CDATA_SECTION_NODE          = 4;
  const unsigned short      ENTITY_REFERENCE_NODE       = 5;
  const unsigned short      ENTITY_NODE                 = 6;
  const unsigned short      PROCESSING_INSTRUCTION_NODE = 7;
  const unsigned short      COMMENT_NODE                = 8;
  const unsigned short      DOCUMENT_NODE               = 9;
  const unsigned short      DOCUMENT_TYPE_NODE          = 10;
  const unsigned short      DOCUMENT_FRAGMENT_NODE      = 11;
  const unsigned short      NOTATION_NODE               = 12;
```

```
  readonly attribute DOMString       nodeName;
           attribute DOMString       nodeValue;
                                       // raises(DOMException) on setting
                                       // raises(DOMException) on retrieval
```

DOM komponenty - Node

```
readonly attribute unsigned short   nodeType;
readonly attribute Node             parentNode;
readonly attribute NodeList         childNodes;
readonly attribute Node             firstChild;
readonly attribute Node             lastChild;
readonly attribute Node             previousSibling;
readonly attribute Node             nextSibling;
readonly attribute NamedNodeMap     attributes;
// Modified in DOM Level 2:
readonly attribute Document         ownerDocument;
```

V implementáciách (zväčša) metódy **get**

DOM komponenty - Node

```
Node                insertBefore(in Node newChild,
                                 in Node refChild)
                                    raises(DOMException);
Node                replaceChild(in Node newChild,
                                 in Node oldChild)
                                    raises(DOMException);
Node                removeChild(in Node oldChild)
                                    raises(DOMException);
Node                appendChild(in Node newChild)
                                    raises(DOMException);
boolean             hasChildNodes();
```

```
// Introduced in DOM Level 2:
boolean             hasAttributes();
```


DOM komponenty - Node

Jednotlivé typy uzlov:

Interface	nodeName	nodeValue	attributes
Attr	name of attribute	value of attribute	null
CDATASection	#cdata-section	content of the CDATA Section	null
Comment	#comment	content of the comment	null
Document	#document	null	null
DocumentFragment	#document-fragment	null	null
DocumentType	document type name	null	null
Element	tag name	null	NamedNodeMap
Entity	entity name	null	null
EntityReference	name of entity referenced	null	null
Notation	notation name	null	null
ProcessingInstruction	target	entire content excluding the target	null
Text	#text	content of the text node	null

DOM komponenty - Node

Ostatné vlastnosti (features, namespaces, …)

```
Node                cloneNode(in boolean deep);
// Modified in DOM Level 2:
void                normalize();
// Introduced in DOM Level 2:
boolean             isSupported(in DOMString feature,
                                in DOMString version);
// Introduced in DOM Level 2:
readonly attribute DOMString        namespaceURI;
// Introduced in DOM Level 2:
         attribute DOMString        prefix;
                                      // raises(DOMException) on setting

// Introduced in DOM Level 2:
readonly attribute DOMString        localName;
```

DOM komponenty - Document

Reprezentuje celý XML dokument (strom)

IDL špecifikácia:

- základné údaje
- vytváranie nových uzlov
- vyhľadávanie uzlov

```
interface Document : Node {
  readonly attribute DocumentType     doctype;
  readonly attribute DOMImplementation  implementation;
  readonly attribute Element          documentElement;
```


DOM komponenty - Document

```
Element            createElement(in DOMString tagName)
                                        raises(DOMException);
DocumentFragment   createDocumentFragment();
Text               createTextNode(in DOMString data);
Comment            createComment(in DOMString data);
CDATASection       createCDATASection(in DOMString data)
                                        raises(DOMException);
ProcessingInstruction createProcessingInstruction(in DOMString target,
                                                   in DOMString data)
                                        raises(DOMException);
Attr               createAttribute(in DOMString name)
                                        raises(DOMException);
EntityReference    createEntityReference(in DOMString name)
                                        raises(DOMException);
// Introduced in DOM Level 2:
Node               importNode(in Node importedNode,
                              in boolean deep)
                                     raises(DOMException);
// Introduced in DOM Level 2:
Element            createElementNS(in DOMString namespaceURI,
                                   in DOMString qualifiedName)
                                      raises(DOMException);
// Introduced in DOM Level 2:
Attr               createAttributeNS(in DOMString namespaceURI,
                                     in DOMString qualifiedName)
                                      raises(DOMException);
```


DOM komponenty - Document

```
NodeList           getElementsByTagName(in DOMString tagname);
// Introduced in DOM Level 2:
NodeList           getElementsByTagNameNS(in DOMString namespaceURI,
                                          in DOMString localName);
// Introduced in DOM Level 2:
Element            getElementById(in DOMString elementId);
```

Obmedzené možnosti vyhľadávania uzlov…
(ťažkopádne cez NodeList)

… vyhľadávanie pomocou XPath operácií

DOM komponenty - Element

Špeciálny typ uzla (Node), práca s atribútmi

```
interface Element : Node {
  readonly attribute DOMString        tagName;
  DOMString           getAttribute(in DOMString name);
  void                setAttribute(in DOMString name,
                                   in DOMString value)
                                        raises(DOMException);
  void                removeAttribute(in DOMString name)
                                        raises(DOMException);
  Attr                getAttributeNode(in DOMString name);
  Attr                setAttributeNode(in Attr newAttr)
                                        raises(DOMException);
  Attr                removeAttributeNode(in Attr oldAttr)
                                        raises(DOMException);
  NodeList            getElementsByTagName(in DOMString name);
```

DOM komponenty - Element

```
// Introduced in DOM Level 2:
DOMString          getAttributeNS(in DOMString namespaceURI,
                                  in DOMString localName);
// Introduced in DOM Level 2:
void               setAttributeNS(in DOMString namespaceURI,
                                  in DOMString qualifiedName,
                                  in DOMString value)
                                      raises(DOMException);
// Introduced in DOM Level 2:
void               removeAttributeNS(in DOMString namespaceURI,
                                     in DOMString localName)
                                      raises(DOMException);
// Introduced in DOM Level 2:
Attr               getAttributeNodeNS(in DOMString namespaceURI,
                                      in DOMString localName);
// Introduced in DOM Level 2:
Attr               setAttributeNodeNS(in Attr newAttr)
                                      raises(DOMException);
// Introduced in DOM Level 2:
NodeList           getElementsByTagNameNS(in DOMString namespaceURI,
                                          in DOMString localName);
// Introduced in DOM Level 2:
boolean            hasAttribute(in DOMString name);
// Introduced in DOM Level 2:
boolean            hasAttributeNS(in DOMString namespaceURI,
                                  in DOMString localName);
```


DOM komponenty - Attr

Atribút Elementu

name - **value** dvojica

nechápané ako súčasť stromu

(*parentNode*, *siblings* sú **null**)

specified - ak pridelená hodnota v originálnom dok.

```
interface Attr : Node {
  readonly attribute DOMString        name;
  readonly attribute boolean          specified;
           attribute DOMString        value;
                                        // raises(DOMException) on setting

  // Introduced in DOM Level 2:
  readonly attribute Element          ownerElement;
};
```


DOM, návratové typy

```
interface NodeList {
  Node               item(in unsigned long index);
  readonly attribute unsigned long    length;
};
```

```
interface NamedNodeMap {
  Node               getNamedItem(in DOMString name);
  Node               setNamedItem(in Node arg)
                                      raises(DOMException);
  Node               removeNamedItem(in DOMString name)
                                      raises(DOMException);
  Node               item(in unsigned long index);
  readonly attribute unsigned long    length;
  // Introduced in DOM Level 2:
  Node               getNamedItemNS(in DOMString namespaceURI,
                                    in DOMString localName);
  // Introduced in DOM Level 2:
  Node               setNamedItemNS(in Node arg)
                                      raises(DOMException);
  // Introduced in DOM Level 2:
  Node               removeNamedItemNS(in DOMString namespaceURI,
                                       in DOMString localName)
                                      raises(DOMException);
};
```


DOM, Java implementácia

Majme “databázu” používateľov

```
<?xml version="1.0"?>
<Users>
    <User ID="u1">
        <Name>anka</Name>
    </User>
    <User ID="u2">
        <Name>robo</Name>
    </User>
    <User ID="u3">
        <Name>roman</Name>
    </User>
</Users>
```

DOM, Java implementácia

Modul na sprístupnenie dát

```
// imports
import ...;

public class UserManager
{
    // inicializácia (static, konštruktor)

    public Element getUser (int index)
    {   }

    public Element findByName (String name)
    {   }

    public String getID (Element user)
    {   }

    public String getName (Element user)
    {   }
}
```


DOM, Java implementácia

Java implementácia (Apache) rozhraní

```
// parser (DOM, Apache Xerces)
import org.apache.xerces.parsers.DOMParser;
import org.xml.sax.InputSource;

// implementacia DOM
import org.w3c.dom.*;

// XPath API (Apache)
import org.apache.xpath.XPathAPI;

public class UserManager
{
    private static Document users;
```


DOM, Java implementácia

Parsovanie dokumentu na DOM reprezentáciu

```
try
{
    DOMParser parser = new DOMParser();
    parser.parse (new InputSource ("users.xml"));

    users = parser.getDocument();

    parser = null;
}
catch (Exception e)
{
```


DOM, Java implementácia

```
public Element getUser (int index)
{
    Element elm = null;

    try
    {
        NodeList nl = users.getElementsByTagName ("User");

        if (index < nl.getLength())
            return (Element) nl.item (index);
    }
    catch (Exception e){}

    return null;
}
```

DOM, Java implementácia

```
public Element findByName(String key)
{
    Element elm = null;

    try
    {
        elm = (Element)
              XPathAPI.selectSingleNode
                        (
                          users.getDocumentElement(),
                          "//User[Name='" + key + "']"
                        );
        return elm;
    }
    catch (Exception e){}

    return null;
}
```


DOM, Java implementácia

```
public String getID (Element user)
{
    return user.getAttribute ("ID");
}

public String getName (Element user)
{
    NodeList nl = user.getElementsByTagName ("Name");

    return ((Element) nl.item(0)).getFirstChild().getNodeValue();
}
```


DOM, zhrnutie

- Document Object Model
- Odporúčanie vo vlastníctve W3C
- Objektová reprezentácia XML dokumentu
  - Node, Document, Element, Attr, …
- Existujúce implementácie (Apache, …) API

Simple API for XML, SAX

- Vytvorené používateľmi (XML-DEV mailing list)
- Udalosti zasielané počas parsovania dokumentu
  - časovo aj pamäťovo menej náročné než DOM
  - typicky používané na čítanie dokumentov
    - (nie modifikácie, …)

SAX podpora v XML parseroch

JAXP, Java API for XML Parsing

Sun Microsystems, java.sun.com/xml, Java 1.4

Xerces, Java/C++/Perl

Apache Foundation, xml.apache.org

msxml, 3.0

Microsoft, msdn.microsoft.com/xml

XML dokument ako postupnosť SAX udalostí

```
<?xml version="1.0"?>
<doc name="first">
     <!-- text -->
     <autor>
        <meno>Roman</meno>
        <priez>F</priez>
     </autor>
     ...
</doc>
```

udalosti:

začiatok/ koniec:
- dokumentu
- elementu

text

komentár

…

- začiatok dokumentu
- začiatok elementu
  meno: doc
  atribúty: name = first
- text („\n\t“)
- komentár („ text “)
- text („\n\t“)
- začiatok elementu
  meno: autor
- text („\n\t\t“)
- začiatok elementu
  meno: meno
- text („Roman“)
- koniec elementu
  meno: meno
- text („\n\t\t“)
- začiatok elementu
  meno: priez
- text („F“)
- koniec elementu
  meno: priez
- text („\n\t“)
- koniec elementu
  meno: autor
- …
- koniec elementu
  meno: doc
- koniec dokumentu

SAX rozhrania

call-back rozhrania počas parsovania

Základná práca s udalosťami:

DocumentHandler
ErrorHandler
HandlerBase

DTDHandler, EntityResolver

Podporné:

InputSource
Locator
SAXException
SAXParseException

SAX, DocumentHandler

`void` **startDocument**()
Začiatok dokumentu.

`void` **endDocument**()
Koniec dokumentu.

`void` **startElement**(`java.lang.String name, AttributeList atts`)
Začiatok elementu - meno a atribúty.

`void` **endElement**(`java.lang.String name`)
Koniec elementu - meno.

`void` **characters**(`char[] ch, int start, int length`)
Reťazec znakov.

`void` **ignorableWhitespace**(`char[] ch, int start, int length`)
Ignorovateľný reťazec whitespace.

`void` **processingInstruction**(`java.lang.String target, java.lang.String data`)
Processing Instruction - meno a parametre.

`void` **setDocumentLocator**(`Locator locator`)
Nastavenie Locator-a parsovania.

SAX, ErrorHandler

`void` **error** `(SAXParseException exception)`
Nastala opraviteľná (recoverable) chyba.
`void` **fatalError** `(SAXParseException exception)`
Nastala neopraviteľná (non-recoverable) chyba.
`void` **warning** `(SAXParseException exception)`
Varovanie počas parsovania.

Každá z nich: `throws SAXException`

SAX, HanderBase

Trieda implementujúca všetky základné rozhrania
“prázdne” metódy, potrebné predefinovať

SAX, Locator

Poskytnuté na začiatku parsovania
DocumentHandler.setDocumentLocator ()

`int` **getColumnNumber** `()`
Číslo stĺpca, kde končí aktuálna "udalosť" v dokumente.
`int` **getLineNumber** `()`
Číslo riadku, kde končí aktuálna "udalosť" v dokumente.
`String` **getPublicId** `()`
Identifikátor (verejný) aktuálnej "udalosti" v dokumente.
`String` **getSystemId** `()`
Identifikátor (systémový) aktuálnej "udalosti" v dokumente.

SAX, InputSource

Zjednocujúce rozhranie pre rôzne vstupy

Konštruktory:

InputSource `(java.io.InputStream byteStream)`

InputSource `(java.io.Reader characterStream)`

InputSource `(java.lang.String systemId)`

SAX, Java implementácia

Import dát do DB - hierarchia “zmlúv” a “plánov”
dáta: priamo používateľ / výstup z inej DB
veľkosť dokumentu od 1KB - 1MB
logická chyba - zneplatňuje vstupné dáta

1. fáza - validácia dokumentu
syntaktická kontrola (štruktúra, typy dát…)
2. fáza - import dát
logická kontrola (existencia v DB, číselníky)
filtrovanie dát, import do databázy

SAX, Java implementácia

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Auftrag SYSTEM "http://...">
<?Import_Type Additional?><?Language DE?>

<Auftrag Auftrags_Nummer="XMLTEST2-1">
    <Teilauftrag Teilauftrags_Nummer="00010">
        <Kostenplan>
            <Kostenplan_pro_Monat Monat="2000.08"
                Rechner_Kosten="70000" Reise_Kosten="50000"
                Sonstige_Kosten="30000">
                <Kostenplan_pro_Herkunft_und_Monat
                    Herkunft="PSE" Stunden="12"/>
                <Kostenplan_pro_Herkunft_und_Monat
                    Herkunft="ANF" Stunden="8"/>
            </Kostenplan_pro_Monat>
        </Kostenplan>
    </Teilauftrag>
    ...
</Auftrag>
```


SAX, Java implementácia

```
import ...;

public class AuftragParser
{
    public ElementAuftrag parse (String strXml)
    {
        // validating parse
        // creating parse
    }

    class AuftragDocumentHandler extends HandlerBase
    {   }

    class AuftragValidateDocumentHandler extends HandlerBase
    {   }

    class AuftragErrorHandler implements ErrorHandler
    {   }
}
```


SAX, Java implementácia

```
import java.io.*;

import org.xml.sax.*;
import org.xml.sax.helpers.ParserFactory;
import org.apache.xerces.parsers.*;

import siemens.proweb...;

public class AuftragParser
{
    private Locator locator = null; // Locator
```

Xerces implementácia

SAX, Java implementácia

```
public ElementAuftrag parse (String strXml)
{
    try
    {
        Parser parser = ParserFactory.makeParser
                        ("org.apache.xerces.parsers.SAXParser");
        // validating parse
        parser.setDocumentHandler (new AuftragValidateDocumentHandler());
        parser.setErrorHandler (new AuftragErrorHandler());

        try { ((org.apache.xerces.parsers.SAXParser) parser).setFeature
                                ("http://xml.org/sax/features/validation", true);
              ((org.apache.xerces.parsers.SAXParser) parser).setFeature
                                ("http://xml.org/sax/features/namespaces", false);
        } catch (Exception e) {}

        parser.parse (new InputSource (new StringReader (strXml)));
```

SAX, Java implementácia

```
    // creating parse
    if (! wasError)
    {
        parser.setDocumentHandler (new AuftragDocumentHandler());

        try { ((org.apache.xerces.parsers.SAXParser) parser).setFeature
                        ("http://xml.org/sax/features/validation", false); }
        catch (Exception e) {}

        parser.parse (new InputSource (new StringReader (strXml)));
    }
    else {}
  }
  catch (Exception e) {}

  return ...;
}
```


SAX, Java implementácia

```
class AuftragValidateDocumentHandler extends HandlerBase
{
    public void setDocumentLocator (Locator loc)
    {
        locator = loc;
    }
}
class AuftragErrorHandler implements ErrorHandler
{
    public void warning (SAXParseException e) throws SAXException
    {
        System.out.println ("Warning: " + e);
    }

    public void error (SAXParseException e) throws SAXException
    { ... }

    public void fatalError (SAXParseException e) throws SAXException
    {
        System.out.println ("Fatal: " + e);
        throw e;
    }
}
```



```
class AuftragDocumentHandler extends HandlerBase
{
    private int intImpType;

    public void startElement (String strName, AttributeList alAttrs)
           throws SAXException
    {
        if (strName.equalsIgnoreCase ("Auftrag"))
        {
            putAuftrag (alAttrs, intImpType);
        }
        else if (strName.equalsIgnoreCase ("Teilauftrag"))
        {
            putTeilauftrag (alAttrs);
        }
        ...
    }

    public void endElement (String strName) throws SAXException
    { ... }
    public void processingInstruction (String strTarget, String strData)
           throws SAXException
    { ... }
    ...
}
```


SAX, Java implementácia

```
private void putAuftrag (AttributeList atts, int intImpType)
{
    ...
}

private void putTeilauftrag (AttributeList atts)
{
    ...
}

...
```

Logická kontrola dát oproti DB
Odpamätávanie relevantných dát

SAX 2.0

`HandlerBase` nahradený `DefaultHandler`

Rožšírené o podporu namespace (elementy a atribúty)

Upravené procesy získavania a parsovania dokumentu

SAX, zhrnutie

Simple API for XML

de-facto štandard

Udalosťami reprezentovaný XML dokument

startElement, endElement, …

Existujúce implementácie (Apache, …) API

Ďakujem za pozornosť

WWW Consortium (špecifikácie)
www.w3.org

OpenSource implementácia parsera (Xerces)
xml.apache.org

Tutoriály, materiály
www.zvon.org

iné
www.xml.org, …