INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

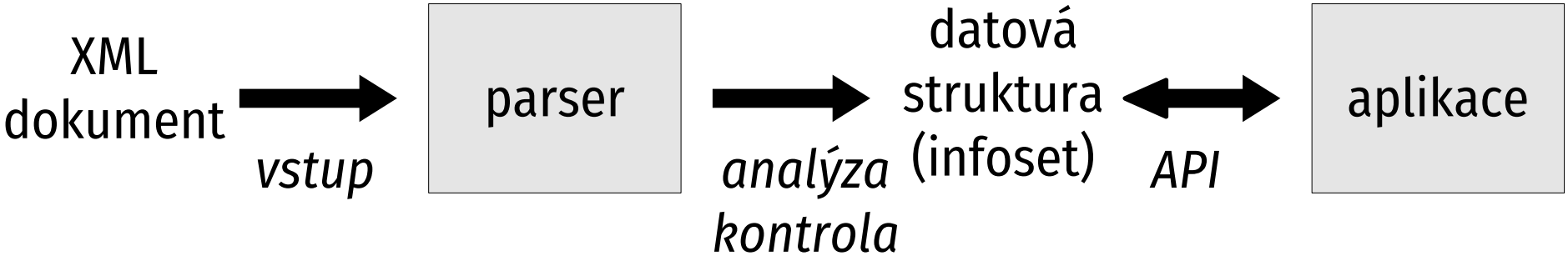
Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu *Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP* reg. č. CZ.1.07/2.2.00/29.0011

API pro XML

XML a programování

- **rukodělně**
  - XML dokument je textový soubor – lze zpracovávat běžnými textovými funkcemi, regulárními výrazy apod.
  - pracné, ale může být velmi rychlé
  - použitelné, pokud je struktura souboru jednoduchá
- **využití existující knihovny (API)**
  - méně práce (ale seznámení s knihovnou stojí čas)
  - sdílení zkušeností s ostatními programátory
  - doporučená cesta

Vstup XML dat – parser



- parser zajistí načtení, syntaktickou analýzu a kontrolu správnosti XML dokumentu
- vytvoří datovou strukturu odpovídající obsahu (typicky strom)

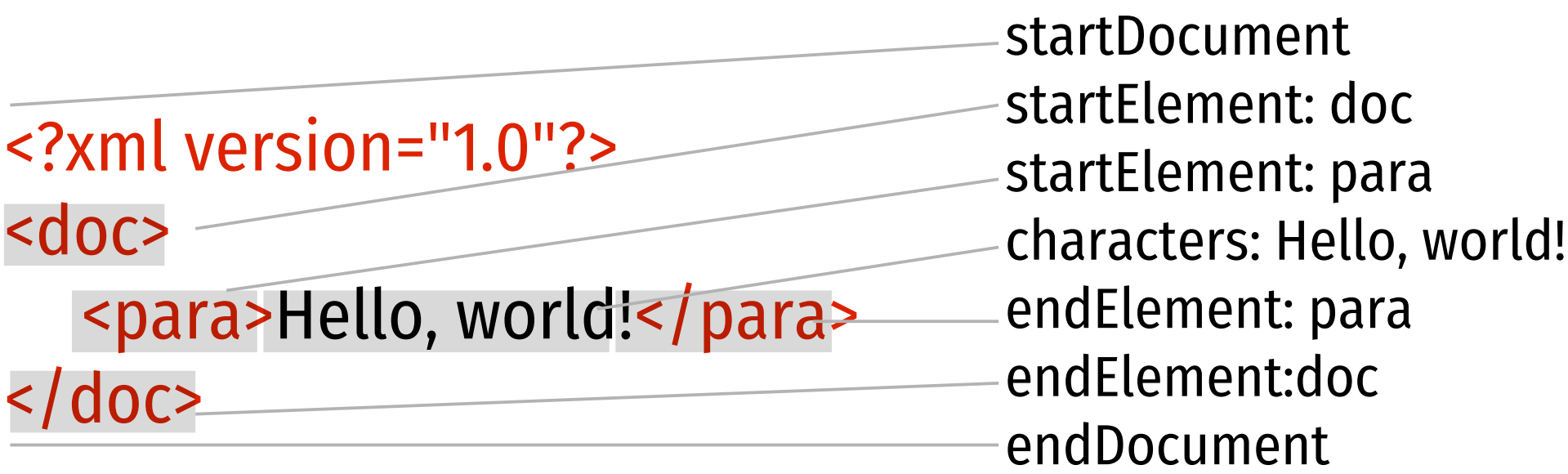
Základní typy API

- **řízené událostmi (SAX)**
  - kdykoli při čtení dokumentu dojde k důležité události (např. začátek prvku), parser volá obslužnou funkci
  - minimální režie (nevytváří strom v paměti)
  - veškeré souvislosti si musí udržovat aplikace
- **stromové (DOM)**
  - převede dokument na strom v paměti
  - volný přístup ke všem datům
  - režie úměrná velikosti dokumentu

SAX

- **Simple API for XML**
- jeden průchod dokumentem, významné jevy při čtení jsou přenášeny jako **události**, typicky:
  - startDocument, endDocument
  - startElement, endElement
  - characters
- návod, odkazy na konkrétní parsery apod.: www.saxproject.org

Příklad SAX událostí



Nevýhody SAX

- SAX je „tlačné“ (push) rozhraní – aplikace je de facto řízena parserem
  - „tažná“ (pull) rozhraní používají podobný způsob práce s dokumentem, ale o události si říká aplikace, např. opakovaným voláním funkce next()
- je jednosměrný – pouze pro čtení dokumentů
  - žádná podpora pro XML výstup
  - výstup je realizován jako běžný text

StAX

- **Streaming API for XML**
- **kurzor** označuje aktuální pozici v XML dokumentu
  - ukazuje na logické prvky (textový uzel, značku prvku,...)
  - pohybuje se jen vpřed, řídí aplikace – metodou next()
  - získávání informací metodami getName(), getText(),...
- základní rozhraní: třídy XMLStreamReader a XMLStreamWriter
- https://github.com/codehaus/stax/

Příklad – jména prvků

```
while (true) {
    int event = parser.next();

    if (event == XMLStreamConstants.END_DOCUMENT) {
       parser.close();
       break;
    }

    if (event == XMLStreamConstants.START_ELEMENT) {
        System.out.println(parser.getLocalName());
    }
}
```

viz http://www.xml.com/pub/a/2003/09/17/stax.html

DOM

- **Document Object Model**
- stromová reprezentace dokumentu (hierarchie objektů odpovídajících částem XML – prvkům, atributům atd.)
- všechny informace přístupné naráz – lze využívat libovolně a opakovaně (ale vyžaduje paměť a čas)
- vytvořilo W3C jako univerzální nástroj
  - používá i JavaScript pro manipulaci s WWW stránkou

XOM

- **XML Object Model**
- demonstrace převoditelnosti přístupů – postaví strom dokumentu, sám ale využívá SAX
- snaha o jednoduché, snadno zvládnutelné a efektivní rozhraní
- umožňuje zpracovávat dokument po částech
- xom.nu

API vázané na data (data binding)

- podobný DOMu – vytváří v paměti hierarchii odpovídající dokumentu
- objekty odpovídají **„na míru“ použitému XML jazyku**, nikoli obecným součástem XML
  - objekty Cenik, Zbozi atd. nikoli Element, jehož atribut Name má hodnotu Cenik či Zbozi
- typicky staví na **kompilaci schématu** – připraví třídy
- např. **XMLBeans** (xmlbeans.apache.org)

Transformační/dotazovací API

- aplikační interface pro XSLT, XPath a podobně
- hlavní kód většinou leží mimo API, knihovna jen zajišťuje tlumočení mezi aplikací a kódem provádějícím XML operace
- např.
  TrAX (http://xml.apache.org/xalan-j/trax.html)
  Jaxen (https://github.com/jaxen-xpath/jaxen)

Příklady XML jazyků

Konkrétní XML jazyky (formáty)

- XML je nástroj pro definici jazyků pro konkrétní aplikace a služby
- zveřejněných jazyků s ambicí na neutralitu a pozici de facto či de iure standardu jsou stovky – viz http://en.wikipedia.org/wiki/Category:XML-based_standards
- následují vybrané příklady

XHTML

- **eXtensible HyperText Markup Language**
- jazyk webových stránek, následník HTML
- svého času nejpoužívanější jazyk založený na XML
- W3C ohlásilo, že bude dále rozvíjet jen XHTML
- XHTML 2.0 neuspělo
- vznik HTML5 mimo W3C, následně přijato W3C, existuje i XHTML5
- validator.w3.org

RSS

- **Rich Site Summary (RDF Site Summary, Really Simple Syndication)**
- pro oznamování novinek na WWW serverech
- houfně používají zpravodajské servery a blogy
- sledovat lze specializovaným programem, WWW klientem i na integrujících stránkách (Feedly)
- velmi jednoduchý jazyk, přesto existuje několik nekompatibilních verzí

Příklad RSS

- tematický kanál = soubor

```
<rss version="0.91">
<channel>
    <title>Satrapovy novinky</title>
    <link>http://www.kai.tul.cz/~satrapa/rss.xml</link>
    <description>Mé fiktivní zpravodajství pro studenty.</description>
    <language>cs</language>
<item>
    <title>Vypsány termíny z Počítačových sítí</title>
    <link>http://stag.tul.cz/</link>
    <description>Termíny ze sítí jsou ve STAGu, zapisujte se.</description>
</item>...
</channel>
</rss>
```

ATOM

- **Atom Syndication Format**
- nástupce RSS
- nese více informací (různé povinné položky)
- větší možnosti pro obsah – různé formáty, včetně binárních

DocBook

- pro **technickou dokumentaci** (postupně se protlačuje do pozice de facto standardu)
- původně v SGML, více se používá XML verze
- spousta prvků – definována podmnožina **Simplified DocBook** (o něco menší spousta prvků)
- k dispozici konverze (na bázi XSLT) do řady formátů – (X)HTML, PDF, RTF,...
- www.docbook.org

Příklad DocBook

```
<book id="kniha-pokus">
 <title>Minimalistická kniha</title>
 <chapter id="kap-uvod">
  <title>Úvod</title>
  <para>No nazdar!</para>
  <para>Začal jsem psát knihu, snad to dobře dopadne...</para>
 </chapter>
 <chapter id="kap-xml">
  <title>XML</title>
  <para>V první kapitole se seznámíme s XML.</para>
 </chapter>
</book>
```

OpenDocument Format (ODF)

- připravila OASIS (stejně jako DocBook), nyní ISO standard
- původně jej vytvořil a používá **OpenOffice.org**, používají i další balíky
- formát pro kancelářské aplikace
- může být jednoduchý XML soubor, častěji ale ZIP archiv obsahující řadu adresářů a souborů obsahujících jednotlivé prvky dokumentu
- ISO standard

Office Open XML

- vytvořil Microsoft pro nové verze svého balíku **Microsoft Office** (od 2007)
- přímý konkurent ODF
- po řadě kontroverzí přijat jako ISO standard

MathML

- **Mathematical Markup Language**, W3C doporučení
- původní cíl: matematické vzorce do webu
- typický představitel masivního značkování
- y=|x|

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
<ci>y</ci><mo>=</mo>
<apply>
    <abs/>
    <ci>x</ci>
</apply>
</math>
```

SVG

- **Scalable Vector Graphics**
- W3C doporučení
- dvojrozměrná vektorová grafika (ale může obsahovat i rastrová data)
- podporuje řada prohlížečů (nativně či pomocí plug-inů) – vektorová grafika pro web
- implementace: www.svgi.org
  doporučuji **Inkscape** (www.inkscape.org)

Příklad

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
      "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg   width="100%" height="100%"
       version="1.1" xmlns="http://www.w3.org/2000/svg">

<rect  width="300"
       height="100"
       style="fill:rgb(0,0,255);
              stroke-width:1;stroke:rgb(0,0,0)"/>

</svg>
```

XML-RPC

- **Remote Procedure Call**
- jednoduchý mechanismus pro síťovou spolupráci aplikací – klient volá podprogram na serveru
- přenosovým protokolem HTTP
- data formátována v minimalistickém XML jazyce (méně než 10 datových typů)

Příklad XML-RPC

```
<?xml version="1.0"?>
<methodCall>
    <methodName>
        priklad.getPSCMesto
    </methodName>
    <params>
        <param>
            <value>
                <i4>46000</i4>
            </value>
        </param>
    </params>
</methodCall>
```

```
<?xml version="1.0"?>
<methodResponse>
    <params>
        <param>
         <value>
          <string>Liberec</string>
         </value>
        </param>
    </params>
</methodResponse>
```

SOAP

- **Simple Object Access Protocol**
- následník XML-RPC, univerzální vrstva pro výměnu zpráv ve webových aplikacích
- různé přenosové modely a mechanismy, nejběžnější je RPC – požadavek a odpověď kódovány do SOAP zpráv

Příklad – SOAP dotaz

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/
envelope/">
    <soap:Body>
        <getPSCMesto xmlns="http://kdesi.cz/psc">
            <psc>46000</psc>
        </getPSCMesto>
    </soap:Body>
</soap:Envelope>
```

Příklad – SOAP odpověď

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/
envelope/">
    <soap:Body>
        <getPSCMestoResponse xmlns="http://kdesi.cz/psc">
            <getPSCMestoResult>
                <jmeno>Liberec</jmeno>
                <psc>46000</psc>
                <kraj>Liberecky</kraj>
            </getPSCMestoResult>
        </getPSCMestoResponse>
    </soap:Body>
</soap:Envelope>
```

XUL

- **XML User Interface Language**
- definuje uživatelské rozhraní Mozilly
- obsahuje prvky pro popis běžných součástí GUI (okna, tlačítka, menu)
- využívá existující technologie (CSS, JavaScript,...)
- MS Windows mají **XAML** (Extensible Application Markup Language)

Alternativy XML

JSON

- **JavaScript Object Notation**
- využíván především pro on-line aplikace
- syntax vychází z JavaScriptu
- existují parsery pro řadu jazyků – viz json.org
- **podobné:** textový zápis, hierarchické uspořádání
- **odlišné:** kratší (nemá koncové značky), obsahuje pole, nemá atributy, chybí nadstavbový ekosystém
- ECMA-404, RFC 7159

Příklad

```
{ "cenik" : [
    {   "nazev" : "Houska",
        "cena" : 1.70                },
    {   "nazev" : "Voda",
        "cena" : 7.50                }
    ]
}
```

JSON typy

- **řetězec znaků** – uzavřen do " ... "
- **číslo** – neřeší varianty
- **pravdivostní hodnota** – true, false
- **objekt** – uzavřen do { ... }, nezáleží na pořadí
- **pole** – uzavřeno do [...], indexováno od 0
- **null** – prázdná hodnota

JSON Schema

- cíl: umožnit strojovou kontrolu korektnosti dat (obvyklý vývoj – JSON je populární pro svou jednoduchost, ale uživatelé chtějí další schopnosti)
- standardizace teprve probíhá: draft-bhutton-json-schema
- vychází z JSON syntaxe, definuje názvy položek a typy jejich hodnot
- json-schema.org
- několik implementací

Příklad

```
{    "$schema" : "http://json-schema.org/draft-04/schema#",
     "title" : "Ceník",
     "descripiton" : "Informace o nabídce a cenách zboží",
     "type" : "object",
     "properties" : {
          "cenik" : {
               "type" : "array",
               "items" : {
                    "type" : "object",
                    "properties" : {
                         "nazev" : { "type" : "string" },
                         "cena" : { "type" : "number", "minimum" : 0 } }
                    "required" : [ "nazev", "cena" ]
}}}}
```

YAML

- **YAML Ain‘t Markup Language**
  (původně Yet Another Markup Language)
- inspirováno programovacími jazyky, XML a formátem elektronické pošty
- založeno na odsazování
  - sourozenci odsazeni stejně od levého okraje
  - potomci odsazeni více

Příklad

cenik:

- nazev: Houska
  cena: 1.70

- nazev: Voda
  cena: 7.50

YAML typy (1)

- **řetězec znaků** – nevyžaduje uvozovky
- **číslo** – rozlišuje celá a s plovoucí desetinnou čárkou
- **pravdivostní hodnota** – true, false
- **čas** – podle ISO 8601, příp. s mezerami
  2015-01-14t16:23:37.15-02:00
- **datum** – v pořadí rok-měsíc-den podle ISO 8601
  2015-01-14

YAML typy (2)

- **pole** – položky zahájeny pomlčkou a mezerou:
  - první
  - druhá
  - třetí

  lze i kompaktně [první, druhá, třetí]
- **asociativní pole (mapy)** – zápis *klíč*: *hodnota*

  nazev: Houska
  cena: 1.70

  lze i kompaktně { nazev: Houska, cena: 1.70 }

YAML typy (3)

- lze i definovat vlastní
- obvykle se určují automaticky, ale lze i explicitně převést na daný typ !!float 10
- nemá prostředky pro definici schématu, ale vznikají doprovodné prostředky
  - Rx
  - Kwalify

Srovnání

- vlastní XML lze nahradit snadno
  - moc toho neobsahuje, totéž lze i efektivněji
- kontrolu dat už obtížněji
  - nabídka nástrojů pro popis a validaci datových struktur je poměrně omezená
- transformace/dotazování velmi obtížně
  - univerzální nástroje XSLT či XQuery nemají alternativu
  - pouze vlastní aplikace na míru pro konkrétní použití

Dotazy ve Stack Overflow