Zpravodaj Československého sdružení uživatelů TeXu

Jiří Kosek
JadeTeX

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 13 (2003), No. 1, 15–26

Persistent URL: http://dml.cz/dmlcz/149912

Terms of use:

© Československé sdružení uživatelů TeXu, 2003

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.

This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* http://dml.cz

JadeTEX

Jiří Kosek

JadeTEX je speciální formát pro TEX, který umožňuje formátování SGML a XML dokumentů pomocí stylů zapsaných v jazyce DSSSL. JadeTEX neumí interpretovat DSSSL styl přímo, dokumenty SGML/XML musíme nejprve zpracovat pomocí programu Jade, který vytvoří meziprodukt vhodný pro zpracování JadeTEXem.

DSSSL

Stylový jazyk DSSSL (Document Style Semantics and Specification Language) vznikl jako výsledek snahy o vytvoření jazyka, který umožní popsat formátování třídy dokumentů SGML bez ohledu na konkrétní výstupní formát. To umožní vytvářet předpis formátování nezávislý na konkrétní aplikaci použité pro formátování dokumentu. V roce 1996 byla tato snaha završena vydáním DSSSL jako ISO/IEC mezinárodní normy 10179.

DSSSL definuje v podstatě dva jazyky. První z nich – transformační – umožňuje popsat transformaci SGML dokumentu do jiného SGML dokumentu. To jde využít například při převodu SGML do HTML. Druhý jazyk umožňuje formátování dokumentů tím, že se jednotlivým částem vstupního dokumentu přiřadí prezentační charakteristiky jako zarovnání, písmo, barva apod. Z tohoto abstraktního předpisu pak formátovač vytvoří výsledný vysázený dokument. Řídí se přitom charakteristikami definovanými ve stylu s tím, že se rozhoduje o takových věcech jako jsou řádkové a stránkové zlomy.

Aby šlo v DSSSL vyjadřovat složité transformace a formátování, obsahuje ještě dotazovací jazyk a jazyk pro zápis výrazů. Dotazovací jazyk umožňuje pohyb po stromu dokumentu SGML a výběr jeho jednotlivých částí. Jazyk pro zápis výrazů vychází ze Scheme a je hodně podobný LISPu, který je v našich krajích známější než Scheme.

Jade/OpenJade

Abychom mohli na SGML aplikovat DSSSL styl, musíme mít k dispozici program, který umí pravidla ze stylu aplikovat na dokument. Takovému programu se obvykle říká procesor DSSSL. Jednoznačně nejznámějším je program Jade od Jamese Clarka. Protože na jeho vývoji již James několik let nepracuje, vzniklo na

jeho základě OpenJade, které velmi pozvolným tempem přidává nové vlastnosti a opravuje chyby. Jade/OpenJade[1] jsou open source programy napsané v C++ a jsou k dispozici pro všechny běžně používané platformy, Windows a Linux nevyjímaje.

Jade v sobě obsahuje několik koncových modulů, které jsou schopné výsledek formátování uložit do různých formátů. Mezi tyto formáty patří RTF, MIF (používá ho DTP systém FrameMaker od Adobe) a TEX. V případě TEXu se však nejedná o nějaký běžný formát jako plainTEX či LATEX, ale o dost specifický kód, který jde zpracovat jen JadeTEXem. JadeTEX je tedy jen poslední článek v řetězci, který umožňuje SGML dokumenty pomocí DSSSL převést do tištěné podoby.

Poznamenejme ještě, že Jade umí kromě SGML číst i XML dokumenty (XML je konec konců podmnožina SGML), takže jej lze použít i pro zpracování XML. Dále v textu proto budu mluvit o XML, i když popsaným způsobem jde zpracovat i dokumenty SGML.

Instalace

Jak již bylo řečeno, pro práci s JadeTEXem potřebujeme i program Jade. Můžeme si jej stáhnout z adresy `http://www.jclark.com/jade/`, nebo si můžeme pořídit jeho novějšího následníka OpenJade (`http://openjade.sourceforge.net`). Tuto možnost využijí zejména uživatelé Windows, protože většina linuxových distribucí již obsahuje Jade v podobě balíčku.

Pro správnou funkci Jade je po instalaci ještě potřeba upravit katalogové soubory pro SGML. Do proměnné prostředí `SGML_CATALOG_FILES` uložíme cestu ke katalogovému souboru pro Jade (obvykle `c:\jade\catalog` nebo `c:\openjade\dsssl\catalog`).

JadeTEX je formát TEXu a bývá běžnou součástí modernějších distribucí jako TEXLive a teTEX. Aktuální verzi si můžete stáhnout na adrese `http://jadetex.sourceforge.net`.

Pravidla

Základem DSSSL stylu jsou pravidla. Pravidlo definuje, jak se má určitá část vstupního dokumentu XML zpracovat. Při zpracování dokumentu pak Jade postupně prochází jeho jednotlivé elementy a hledá pravidla, která definují jejich zpracování. Pravidlo má tvar:

```
(element název elementu
        zpracování elementu)
```

[1] Dále v textu bude již používat jen název Jade, i když vše uvedené se týká i OpenJade.

Objekty toku dokumentu

DSSSL styl definuje pravidla, které popíší způsob převodu XML dokumentu do formátované podoby. Aby šlo výsledný vzhled dokumentu popsat neutrálně, nezávisle na nějakém konkrétním formátovacím systému, používá DSSSL abstraktní reprezentaci vzhledu dokumentu v podobě objektů toku (flow object). Dokument můžeme popsat jako sekvenci objektů několika různých tříd. K dispozici jsou například objekty definující rozměry stránky, vytvářející odstavec textu, tabulku, seznam či obrázek. Dokument se pak definuje jako sekvence takových objektů. V žargonu DSSSL se tomu říká sosofo – specifikace sekvence objektů toku.

Úkolem pravidel ve stylu je právě generovat pro jednotlivé elementy sekvence objektů toku. Kdybychom chtěli ve vstupním dokumentu převést element `para` na odstavec na výstupu, stačí v pravidle definovat, že pro element se má vytvořit objekt odstavce.

```
(element para
  (make paragraph
    (process-children)))
```

Pravidlo říká, že pro element `para` se vytvoří odstavec a jako jeho obsah se použije výsledek zpracování obsahu elementu dalšími pravidly (`process-children`).

Charakteristiky objektů

Při generování jednotlivých objektů samozřejmě chceme ovlivnit, jak budou vypadat. U odstavce asi budeme chtít nastavit použité písmo, řádkování, mezery před a za odstavcem, způsob zarovnání apod. U každého objektu proto můžeme nastavit několik charakteristik.

```
(element para
  (make paragraph
    font-size: 12pt
    line-spacing: 14pt
    quadding: 'justify
    font-posture: 'italic
    (process-children)))
```

Dotazovací jazyk

Použijeme-li ve všech pravidlech pouze `process-children`, zpracuje se dokument přesně v tom pořadí, jak byl vytvořen. V praxi však existují situace, kdy

potřebujeme některé části dokumentu vynechat nebo je použít na jiném místě. Pro tyto účely obsahuje DSSSL i dotazovací jazyk, který operuje nad stromovou reprezentací vstupního dokumentu XML.

Dotazovací jazyk umožňuje pohyb po vstupním dokumentu a výběr jeho částí. Například kdybychom chtěli vybrat mezi všemi potomky právě zpracovávaného elementu všechny elementy s názvem `obrazek`, zapíšeme to v DSSSL jako:

```
(select-elements (descendants (current-node)) "obrazek")
```

`current-node` zastupuje aktuální element, `descendants` vybere všechny potomky aktuálního elementu a mezi nimi pak pomocí funkce `select-elements` vybereme všechny s názvem `obrazek`.

Praktická ukázka

Ukažme si teď na jednoduchém příkladě kompletní funkční styl v DSSSL. Předpokládejme, že chceme definovat formátování pro články zapisované v XML. Každý článek má přitom následující strukturu:

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!DOCTYPE clanek SYSTEM "clanek.dtd">
<clanek>
  <zahlavi>
    <rubrika>Aktuality</rubrika>
    <nazev>Státní správa bude používat TeX místo Wordu</nazev>
    <autor email="wild@duck.cz">Jan Novák</autor>
  </zahlavi>
  <perex>... text perexu ...</perex>
  <para>... text odstavce ...</para>
  <para>... text odstavce ...
        ... <em>zvýrazněný text</em> ...
        ... </para>
  <para>... text odstavce ...</para>
</clanek>
```

Chceme přitom nadpis článku vysázet větším písmem a vycentrovat, rubrika článku by se měla objevit v záhlaví a jméno a případný e-mail autora až na konci článku. Může nám k tomu posloužit následující styl:

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!DOCTYPE style-sheet
          PUBLIC "-//James Clark//DTD DSSSL Style Sheet//EN">
<style-sheet>
<style-specification>
<style-specification-body>
```

```
;; základní nastavení charakteristik
(declare-initial-value writing-mode 'left-to-right)
(declare-initial-value font-size 12pt)
(declare-initial-value line-spacing 14pt)
(declare-initial-value font-family-name "Arial")
(declare-initial-value language 'CS)
(declare-initial-value quadding 'justify)
(declare-initial-value hyphenate? #t)

;; obsluha kořene stromu XML dokumentu
(element clanek
    (make simple-page-sequence
      ;; rozměry stránky a velikost okrajů
      page-width: 210mm
      page-height: 297mm
      left-margin: 1in
      right-margin: 1in
      top-margin: 1in
      bottom-margin: 1in
      footer-margin: 0.5in
      header-margin: 0.5in
      ;; obsah záhlaví (název rubriky + čára)
      right-header: (make sequence
                      (literal "Rubrika: "
                               (data (select-elements
                                      (descendants (current-node))
                                      "rubrika")))
                      (make rule
                        orientation: 'horizontal
                        line-thickness: 0.4pt
                        ))
      ;; obsah zápatí (číslo strany)
      center-footer: (make sequence
                       font-posture: 'italic
                       (literal "- ")
                       (page-number-sosofo)
                       (literal " -"))
      ;; obsluha vnořených elementů
      (process-children)
      ;; za textem článku se zobrazí jméno autora
      (make paragraph
        space-before: 12pt
        quadding: 'end
```

```
        font-posture: 'italic
        (literal
         "\U-2013; "     ;; Unicode kód pomlčky
         (data (select-elements
                (descendants (current-node))
                "autor"))
         " \U-2013;"
        ))
       ;; jestliže má autor e-mail, zobrazí se
       (if (attribute-string "email" (select-elements
                (descendants (current-node))
                "autor"))
          (make paragraph
            quadding: 'end
            font-size: 10pt
            font-posture: 'italic
            (make sequence
              font-posture: 'normal
              (literal "e-mail: "))
            (literal
             (attribute-string "email" (select-elements
                                        (descendants (current-node))
                                        "autor"))
             ))
          (empty-sosofo))))

;; rubriku si pro zobrazení vybíráme sami,
;; proto ji standardně ignorujeme
(element rubrika
  (empty-sosofo))

;; autora si pro zobrazení vybíráme sami,
;; proto jej standardně ignorujeme
(element autor
  (empty-sosofo))

;; zpracování názvu článku
(element nazev
  (make paragraph
    quadding: 'center
    font-size: 24pt
    line-spacing: 28pt
    font-weight: 'bold
    space-after: 12pt
```

```
    (process-children)))

;; zpracování perexu
(element perex
  (make paragraph
    font-weight: 'bold
    font-posture: 'italic
    (process-children)))

;; zpracování odstavců
(element para
  (make paragraph
    first-line-start-indent: 18pt
    (process-children)))

;; zvýrazněný text bude kurzívou
(element em
  (make sequence
    font-posture: 'italic
    (process-children)))

</style-specification-body>
</style-specification>
</style-sheet>
```

První zajímavostí stylu, o které jsme se ještě nezmínili, je jeho zápis. I když jádro stylu se vyjadřuje pomocí pravidel zapsaných v jazyce odvozeném od Scheme, styl jako celek má podobu SGML dokumentu. Proto jsou pravidla „obalena“ několika elementy.

Stručně se zmíníme o významu některých klíčových míst stylu. Pomocí `declare-initial-value` můžeme pro mnoho charakteristik nastavit výchozí hodnoty. To je vhodné zejména pro věci, které budou společné pro celý dokument nebo alespoň většinu jeho částí.

Poněkud nezvykle nazvaná charakteristika `quadding` určuje způsob zarovnání. Pro zarovnání do bloku se použije hodnota `'justify`. Zarovnání doleva a doprava se vyjadřuje pomocí `'start` a `'end`. Toto pojmenování je voleno, aby bylo nezávislé na směru sazby textu.

Kořenový element dokumentu generuje objekt `simple-page-sequence`, který umí na výstupu generovat sekvence stránek. Jako charakteristiky objektu lze mimo jiné uvést velikost okrajů nebo definovat obsah záhlaví a zápatí (charakteristiky `*-header` a `*-footer`).

Předpokládejme nyní, že ukázkový dokument XML máme uložen v souboru `dokument.xml` a styl v souboru `clanek.dsl`. Chceme získat formátovaný výsledek pomocí JadeTEXu. Nejprve musíme převést dokument XML do podoby vhodné pro JadeTEX:

```
jade -d clanek.dsl -t tex /cesta/k/xml.dcl dokument.xml
```

Parametr `-d` určuje styl s předpisem formátování, `-t` požadovaný výstupní formát. Soubor `xml.dcl` je deklarace SGML pro XML. Tento soubor je nutný, aby se Jade navržený pro práci s SGML vypořádal s dokumenty XML. Soubor je součástí Jade, v Linuxu ho nejčastěji najdete v `/usr/share/sgml/xml.dcl`, ve Windows pak `c:\jade\xml.dcl`, případně `c:\openjade\pubtext\xml.dcl`.

Dostaneme tak texový soubor, který však není určen pro lidské oči:

```
\FOT{2}\Seq%
{\def\WritingMode%
{lefttoright}\def\fSize%
{12\p@}\def\LineSpacing%
{14\p@}\def\LineSpacingFactor%
{0}\def\fFamName{Arial}\def\Language%
{CS}\def\Quadding%
{justify}\def\Hyphenate%
{1}}\Node%
{}\Node%
{\def\Element%
{0}}\SpS%
{\def\PageWidth%
{595.275\p@}\def\PageHeight%
{841.889\p@}\def\LeftMargin%
{72\p@}\def\RightMargin%
{72\p@}\def\TopMargin%
{72\p@}\def\BottomMargin%
{72\p@}\def\FooterMargin%
{36\p@}\def\HeaderMargin%
{36\p@}}
\SpSOtherBackLeftFooter%
... zkráceno ...
```

Můžeme jej však pomocí JadeTEXu přeložit do PDF nebo DVI:

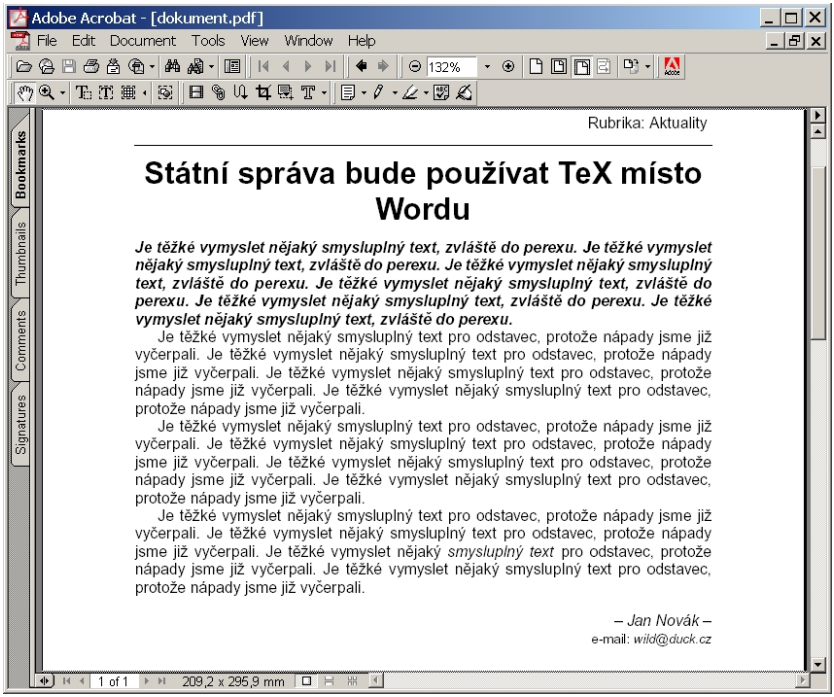
```
pdfjadetex dokument
```

Z našeho dokumentu XML tak dostaneme PDF výstup (viz obrázek 1). Obrázek 2 ukazuje stejný dokument zpracovaný stejným stylem, ale převedený do formátu RTF pomocí příkazu:

```
jade -d clanek.dsl -t rtf /cesta/k/xml.dcl dokument.xml
```

Nezobrazí-li se nám na výstupu správně české znaky, stačí nastavit proměnnou prostředí `SP_ENCODING` na identifikátor použitého kódování (`iso-8859-2` nebo `windows`). Další možností je nastavit proměnnou na hodnotu `XML`. Kódování se pak určí podle XML deklarace na začátku každého dokumentu:

```
<?xml version="1.0" encoding="..."?>
```

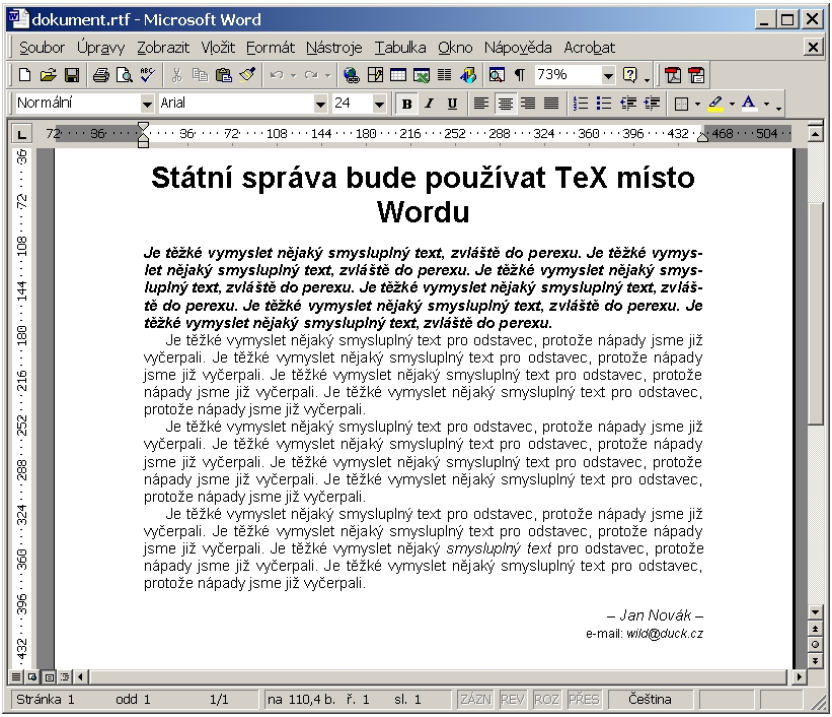


Obrázek 1: Výsledek převodu do PDF pomocí JadeTEXu

Sazba matematiky

DSSSL obsahuje i základní objekty pro vytváření matematické sazby. Můžeme vytvářet zlomky, odmocniny, závorky, operátory (jako suma nebo limita) a matice. Předpokladem samozřejmě je, že ve vstupním dokumentu máme matematiku označkovánu. Pro ilustraci jsem si vymyslel jednoduchou (a v praxi dost nepoužitelnou) notaci pro zápis vzorců obsahujících zlomky a odmocniny:

```
<vzorec>
  <odmocnina>
   <zlomek>
     <citatel>1</citatel>
     <jmenovatel>2</jmenovatel>
   </zlomek>
  </odmocnina>
</vzorec>
```



Obrázek 2: Stejný dokument a styl, tentokráte však ve formátu RTF

```
<vzorec>
  <zlomek>
    <citatel>
      <odmocnina>2</odmocnina>
    </citatel>
    <jmenovatel>2</jmenovatel>
  </zlomek>
</vzorec>
```

Obsluha těchto elementů v DSSSL je velmi přímočará, protože je lze snadno mapovat na jednotlivé matematické objekty toku dokumentu.

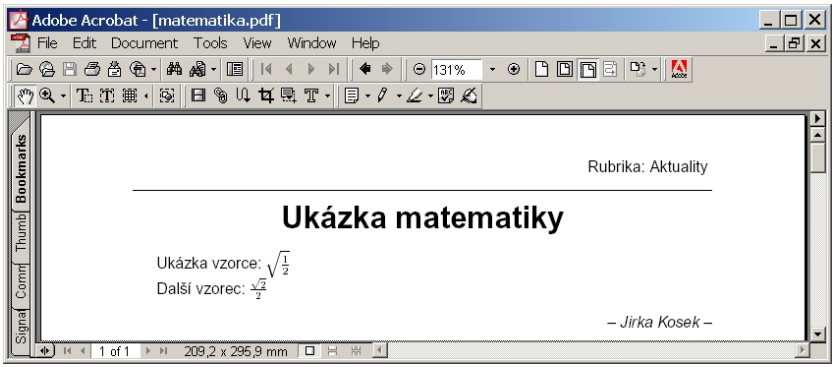
```
(element vzorec
  (make math-sequence
    math-display-mode: 'inline
    (process-children)))
```

```
(element odmocnina
  (make radical
    (process-children)))

(element zlomek
  (make fraction
    (process-children-trim)))

(element citatel
  (make math-sequence
    label: 'numerator
    (process-children)))

(element jmenovatel
  (make math-sequence
    label: 'denominator
    (process-children)))
```



Obrázek 3: Ukázka formátování matematických vzorců

Pro zápis matematiky v XML existuje standardizovaný jazyk MathML[2]. Na adrese `http://www.nag.co.uk/projects/OpenMath/mml-files/` jsou k dispozici styly DSSSL, které umějí zpracovat podmnožinu MathML.

[2] `http://www.w3.org/Math/`

Závěr

Jazyk DSSSL je zajímavou možností pro formátování XML a SGML dokumentů, která umožňuje zcela oddělit obsah dokumentu od jeho vzhledu. Již dnes je však jasné, že mnohem lépe byl přijat novější stylový jazyk XSL a ve většině aplikací se jím DSSSL postupně nahrazuje.

Odkazy

[1] *ISO/IEC 10179. Information technology – Processing languages – Document Style Semantics and Specification Language (DSSSL).* ISO/IEC, 1996. URL: `ftp://ftp.ornl.gov/pub/sgml/WG8/DSSSL/`

Summary: JadeTEX

JadeTeX is a TeX macro package which is able to process SGML and XML documents according to a DSSSL stylesheet in conjunction with (Open)Jade DSSSL processor. This article briefly describes basic principles of the DSSSL language and its usage for formatting XML documents. Complete working example of a DSSSL stylesheet is shown in the article.

Jiří Kosek
`jirka@kosek.cz`

PassiveTEX

Jiří Kosek

PassiveTEX je implementace jazyka XSL FO založená na TEXu. PassiveTEX umí vysázet dokument, jehož obsah a vzhled je popsán speciálním dokumentem XML, který jako značky používá tzv. formátovací objekty. Pojďme se však nejprve stručně seznámit se samotným XSL, abychom lépe pochopili, jaká je role PassiveTEXu při zpracování dokumentů XML.

Princip XSL

Na jazyku pro formátování XML dokumentů se začalo pracovat v podstatě ve stejné době jako na samotném jazyce XML. Poměrně brzy se přišlo na to, že