

# Zpravodaj Československého sdružení uživatelů TeXu

---

Jiří Kosek

Použití parseru XML v TeXu

*Zpravodaj Československého sdružení uživatelů TeXu*, Vol. 13 (2003), No. 1, 6–14

Persistent URL: <http://dml.cz/dmlcz/149911>

## 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>

## Summary: Typesetting XML

This article summarizes methods suitable for processing XML documents by the  $\text{\TeX}$  system – direct typesetting (`xmltex`, `ConTeXt`), conversion to  $\text{\TeX}$  (`XSLT`) and  $\text{\TeX}$  based stylesheet language implementations (`XSL`, `DSSSL`). The article acts as an introduction for more detailed articles about processing XML with  $\text{\TeX}$ .

*Jiří Kosek*  
jirka@kosek.cz

---

## Použití parseru XML v $\text{\TeX}$ u

---

JIŘÍ KOSEK

Chceme-li XML dokumenty sázet čistě prostředky  $\text{\TeX}$ u bez nutnosti používání dalších pomocných programů, je asi nejvhodnějším nástrojem `xmltex`. Jedná se o makro nadstavbu na  $\text{\LaTeX}$ em, která implementuje parser XML. Ten umožňuje načítání dokumentů XML a umí podle pravidel definovaných v konfiguračním souboru převádět jednotlivé elementy na sekvence texového kódu, které pak řídí sazbu. Autorem `xmltex`u je David Carlisle, který je uznávaným odborníkem jak na oblast  $\text{\TeX}$ u, tak i XML a zejména pak stylového jazyka `XSL`.

Většina moderních distribucí  $\text{\TeX}$ u již `xmltex` obsahuje jako samostatný formát. Vysázení dokumentu XML je pak otázkou spuštění jediného příkazu:

```
xmltex dokument.xml
```

resp.

```
pdfxmltex dokument.xml
```

v případě, že chceme získat výstup v PDF pomocí `pdf $\text{\TeX}$` u. Nemáme-li `xmltex` připravený jako samostatný formát, jde využít jednoduchý pomocný dokument v  $\text{\TeX}$ u, který nadefinuje jméno dokumentu ke zpracování a poté aktivuje `xmltex`:

```
\def\xmlfile{dokument.xml}  
\input xmltex.tex  
\end{document}
```

Tento dokument se pak zpracuje pomocí klasického  $\text{\LaTeX}$ u.

## Instalace

Samotný xmltex je již nějakou dobu standardní součástí texových distribucí jako je  $\TeX$ Live nebo tetex. Pro první pokusy s ním tedy není potřeba nic složitě instalovat. Pokud xmltex není součástí vaší texové instalace, můžete si jej stáhnout z CTAN archivu nebo z adresy <http://www.dcarlisle.demon.co.uk/xmltex/> a nainstalovat podle přiložených instrukcí.

Jediným problémem standardně nainstalovaného xmltexu je chybějící podpora českého (a slovenského) dělení slov. Poměrně snadno ji však můžeme doplnit. Postup je popsán dále. Chcete-li si však jen vyzkoušet možnosti xmltexu a prozatím oželite správné dělení slov, můžete zbytek této části článku přeskočit.

Formát xmltexu je odvozen z formátu  $\LaTeX$ u, který používá pro lokalizaci babel (používá se tedy jiná metoda počestění než je  $\CS\LaTeX$ ). Pro správné dělení slov v naší mateřštině proto musíme do  $\LaTeX$ u zahrnout české vzory dělení slov, přegenerovat formát  $\LaTeX$ u a následně i formát xmltexu.

Nejprve musíme definovat, jaké vzory dělení slov ze zahrnou do formátu  $\LaTeX$ u. Definice je uložena v souboru `language.dat`, který se typicky nachází v adresáři `texmf/tex/generic/config`. V souboru musíme odkomentovat řádku, která se stará o načtení českých vzorů dělení slov:

```
czech                czhyph2e.tex
```

Nyní musíme přegenerovat formát  $\LaTeX$ u a xmltexu. Nejjednodušší je použít příkaz `fmtutil --all`, který se postará o přegenerování všech formátů. Ve windowové verzi  $\TeX$ Live je příkaz dostupný i v menu *TeXLive* → *Maintenance* → *Create all format files*.

$\TeX$ Live tímto způsobem rovnou přegeneruje i formát pro xmltex, distribuci tetex o to musíme požádat ručně:

```
fmtutil --cnffile /usr/share/texmf/tex/xmltex/xmltexfmtutil.cnf --all
```

Od této chvíle by měly být programy xmltex a pdfxmltex schopné používat české dělení slov.

## Vazba mezi XML a $\TeX$ em

V okamžiku, kdy xmltexu předáme dokument XML ke zpracování, snaží se xmltex najít definiční soubor, popisující mapování XML na texové sekvence. Tyto soubory mají obvykle příponu `.xmt`. Vazba jednotlivých druhů dokumentů XML na xmt-soubory je definována v katalogu. Vždy se prohledává globální katalog, který musí mít jméno `xmltex.cfg`, a lokální, který se jmenuje stejně jako dokument XML, ale má příponu `.cfg`.

Asi nejspolehlivější způsob svázání XML dokumentů s xmt-souborem je pomocí jmenného prostoru, který lze využít v případě, že zpracováváný dokument

používá jmenné prostory. Dejme tomu, že máme v XML uložený článek a používáme i jmenné prostory.

```
<clanek xmlns="urn:x-kosek:schemas:clanek:v1.0">
...
</clanek>
```

Uložíme-li pak definiční soubor popisující převod elementů článku na texové sekvence například do souboru `clanek.xmt`, musíme do katalogu přidat následující řádku:

```
\NAMESPACE{urn:x-kosek:schemas:clanek:v1.0}{clanek.xmt}
```

Bohužel v mnoha jednodušších případech se jmenné prostory nepoužívají a jednotlivé typy dokumentů XML nelze jednoznačně identifikovat. V těchto případech pak můžeme vazbu na xmt-soubor vytvořit na základě jména kořenového elementu:

```
\NAME{clanek}{clanek.xmt}
```

Musíme být však opatrní, zvláště při použití této deklarace v globálním katalogu, protože se může stát, že budeme mít dva různé druhy dokumentů se stejným kořenovým elementem. `xmltex` pak nebude mít možnost, jak je správně rozlišit.

## Definiční xmt-soubory

Nyní se dostáváme k nejdůležitější části `xmltexu`, a to je vazba mezi elementy XML a `TeXem`. V xmt-souboru můžeme pro každý element definovat kód, kterým se nahradí výskyt počátečního a koncového tagu. Základní kostra definice je velmi jednoduchá:

```
\XMLelement{název elementu}
{zachycení atributů}
{kód pro začátek elementu}
{kód pro konec elementu}
```

Ukažme si konkrétní použití na jednoduchém dokumentu XML s článkem. Předpokládejme, že obsah souboru `dokument.xml` je

```
<?xml version="1.0" encoding="iso-8859-2"?>
<clanek>
  <zahlaví>
    <rubrika>Aktuality</rubrika>
    <nazev>Státní správa bude používat TeX místo Wordu</nazev>
    <autor email="wild@duck.cz">Jan Novák</autor>
  </zahlaví>
  <perex>... text perexu ...</perex>
```

```

<para>... text odstavec ...</para>
<para>... text odstavec ...
    ... <em>zvýrazněný text</em> ...
    ... </para>
<para>... text odstavec ...</para>
</clanek>

```

Vysázet tento článek v L<sup>A</sup>T<sub>E</sub>Xu je velmi jednoduché:

```

\documentclass{article}
\usepackage[czech]{babel}
\gdef\email#1{\{\small\ttfamily#1\}\par}
\begin{document}

\title{Státní správa bude používat TeX místo Wordu}
\author{Jan Novák \email{wild@duck.cz}}
\date{}
\maketitle

\noindent{\bf ... text perexu ...}\par

... text odstavec ...

... text odstavec ...
... {\it zvýrazněný text\}/} ...
...

... text odstavec ...

\end{document}

```

V našem případě si dokument XML s latexovým kódem téměř odpovídá, proto je popsání definice mapování elementů XML velmi jednoduché.

Začneme povinnou kostrou dokumentu s latexovou preambulí. Ta se musí objevit v každém dokumentu. Nejvhodnější ji vygenerujeme v definici pro kořenový element `clanek`:

```

\XMLElement{clanek}{
    {\documentclass{article}
    \usepackage[czech]{babel}
    \gdef\email##1{\{\small\ttfamily##1\}\par}
    \begin{document}}
    {\end{document}}
}

```

Při nalezení počátečního tagu se vygeneruje preambule, po nalezení koncového se naopak ukončí dokument. Protože je `\XMLElement` makro, musíme



```

\else
  \email{\gEmailAutora}
\fi
}}

```

Zároveň ještě testujeme, zda je atribut vůbec v dokumentu nastaven, aby-  
chom zbytečně nesázeli prázdnou e-mailovou adresu.

Chceme-li si vše vyzkoušet, můžeme definice obsluhy elementů uložit do sou-  
boru `clanek.xmt` a vytvořit si jednoduchý katalog `dokument.cfg`:

```
\NAME{clanek}{clanek.xmt}
```

Příkazem

```
pdfxmtex dokument.xml
```

tak získáme vysázenou podobu našeho článku v XML. Anebo ne? Na první  
pohled je vše v pořádku, ale při bližším ohledání zjistíme, že vypadla některá pís-  
mena s diakritikou – třeba ‘ž’, ‘ř’ a ‘č’. Problém je v tom, že XML jako znakovou  
sadu používá Unicode. `xmlltex` však umí standardně na výstup korektně zapsat  
jen znaky, které spadají do kódování ISO Latin 1 (tedy znaky západoevropských  
jazyků).

Naštěstí není nijak složité přidat podporu pro chybějící znaky. Do katalogu  
(soubor s příponou `.cfg`) můžeme dodefinovat chybějící znaky pomocí makra  
`\UnicodeCharacter`. Makro má dva parametry – prvním je unicodový kód  
znaku a druhým sekvence texových příkazů, kterými se znak nahradí. Chybějící  
písmeno ‘ž’ (s kódem 382 decimálně) tak můžeme dodefinovat pomocí:

```
\UnicodeCharacter{382}{\v{z}}
```

Pokud nám hrozí, že písmeno použijeme i v matematickém režimu, je bez-  
pečnější definice ve tvaru:

```
\UnicodeCharacter{382}{\ifmmode \check{z}\else \v{z}\fi}
```

Pro lenochy jako já je tu ještě druhá možnost. Součástí `PassiveTeXu` jsou  
již připravené definice pokrývající slušnou podmnožinu celého Unicode. Načí-  
tají se však až přímo do dokumentu, nekládají se do katalogu. Do preamble  
dokumentu stačí přidat:

```
\RequirePackage{unicode}
```

```
\RequirePackage{characters}
```

Pro ilustraci se nyní podívejme na celý definiční soubor pro náš článek.

```
\XMLElement{clanek}{}
```

```

{\documentclass{article}
 \usepackage[czech]{babel}
 \RequirePackage{unicode}
 \RequirePackage{characters}
 \usepackage{palatino}
 \gdef\email##1{\{\small\ttfamily##1\}\par}

```

```

\begin{document}}
{\end{document}}

\XMLElement{zahlaví}{-}{\date}{\maketitle}

\XMLElement{název}{-}{\xmlgrab}{\title{#1}}

\XMLElement{autor}
  {\XMLAttribute{email}{\emailAutora}{\relax}}
  {\xmlgrab}
  {\xdef\gEmailAutora{\emailAutora}
   \author{#1
    \ifx\gEmailAutora\relax
    \else
    \email{\gEmailAutora}
    \fi
   }}

\XMLElement{rubrika}{-}{\xmlgrab}{-}

\XMLElement{para}{-}{-}{\par}

\XMLElement{perex}{-}{-}{\noindent\bf}{\par}

\XMLElement{em}{-}{\it}{\}/}

```

## Další možnosti

Již toho známe dost, abychom mohli pomocí xmltexu formátovat jednoduché dokumenty XML. V následujících odstavcích se podíváme ještě na pár drobností, které xmltex umí a které se nám mohou v některých případech hodit.

Během formátování dokumentu vypisuje xmltex do logu informace o struktuře zpracovávaného dokumentu. Zejména u delších dokumentů to může být docela rušivé.

```

<0:clanek (./clanek.xmt) >
<0:zahlaví >
<0:rubrika >
</0:rubrika>
Grabbed content
End Grabbed content
<0:název >

```



```

</0:nazev>
Grabbed content
End Grabbed content
<0:autor
  0:email = "wild@duck.cz" >
</0:autor>
Grabbed content
End Grabbed content
</0:zahlati>
<0:perex >
</0:perex>
<0:para >
  <0:em >
  </0:em>
</0:para>
<0:para >
</0:para>
<0:para >
</0:para>
</0:clanek>

```

Chceme-li se hlášení zbavit, stačí do katalogu přidat `\xmltraceoff`.

Ve výpisech je také vidět, že před názvy všech elementů je ještě uvedeno 0:. Tím nám xmltex dává najevo, že element nepatří do žádného jmenného prostoru. Pokud však jmenné prostory používáme, musíme si pro ně v definičním souboru definovat zástupný prefix, kterým se pak dále identifikují elementy a atributy patřící do daného jmenného prostoru. Např.:

```

\DeclareNamespace{m}{http://www.w3.org/1998/Math/MathML}
\XMLelement{m:math}{...}
\XMLelement{m:mi}{...}

```

Poněkud v rozporu s XML specifikací, za to velmi užitečně, umožňuje xmltex předefinování entit uvedených v lokálních deklaracích XML dokumentu. Ukažme si vše na malém příkladě. Dejme tomu, že se v textu píše často o  $\TeX$ u a my si proto pro něj deklarujeme interní textovou entitu:

```

<!DOCTYPE claneek [
<!ENTITY TeX "TeX">
]>
<claneek>
  <zahlati>
    <nazev>Státní správa bude používat &TeX; místo Wordu</nazev>
    ...
</claneek>

```

Při zpracování dokumentu nahradí parser odkaz na entitu `&TeX`; řetězcem „TeX“. To je většině případů žádoucí, protože kromě `TEX`u neumí jiné systémy logo správně zobrazit (například webový prohlížeč). Pro účely zpracování `xmltex` si však můžeme do definičního souboru přidat předdefinování entity. Ta nyní bude volat makro pro správné vysázení názvu našeho oblíbeného typografického systému:

```
\XMLentity{TeX}{\TeX{}}
```

Odkazy na entitu se nyní nahradí za „TeX“ a dostaneme tak požadovaný výstup.

Sám autor `xmltex`u o něm tvrdí, že to není procesor XML, který by 100% odpovídal standardu. Jedním z problémů je chybějící podpora načítání dokumentů kódovaných v UTF-16. V praxi to naštěstí příliš limitující není, protože toto kódování není moc používané. Standardně jsou podporována kódování UTF-8 a ISO Latin 1. Novější verze jsou distribuovány i s podporou středoevropských kódování ISO Latin 2 a windows-1250. Není problém přidat podporu dalších osmibitových kódování. Stačí vytvořit soubor `jméno_kódování.xml` a definovat v něm překódovací tabulku ve tvaru:

```
\InputCharacter{kód}{kód Unicode}
```

## Závěr

V článku jsem se pokusil shrnout a popsat základní principy přímé sazby dokumentů XML pomocí parseru napsaného čistě v `TEX`u. Je to přístup vhodný zejména pro jednodušší dokumenty, které již obsahují text víceméně v tom pořadí, jak se má objevit vysázený na výstupu. V případech, kdy je potřeba před zpracováním dokumentu provést jeho složitější transformace, je vhodnější ke konverzi XML do `TEX`u použít specializované nástroje, například XSLT transformaci.

## Summary: Use of an XML parser in T<sub>E</sub>X

This article shows how to use `xmltex`—an XML parser written in a pure `TEX`—to directly typeset XML documents. Special interest is devoted to correct processing of localized Czech/Slovak documents.

*Jiří Kosek*  
jirka@kosek.cz