Zpravodaj Československého sdružení uživatelů TeXu

Vít Novotný

Sazba textu české lidové písně „Když jsem já sloužil“ pomocí modulu l3seq jazyka expl3

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 33 (2023), No. 3-4, 153–164

Persistent URL: http://dml.cz/dmlcz/151997

Sazba textu české lidové písně „Když jsem já sloužil“ pomocí modulu l3seq jazyka expl3

Vít Starý Novotný

Jazyk plain TEX vznikl pro sazbu knih a turingovsky úplným programovacím jazykem se stal až na konci svého vývoje. Zatímco příprava textu dokumentů a úpravy vzhledu jsou v plain TEXu přímočaré, programování naráží na chybějící základní datové struktury a na odloženou expanzi maker, která neodpovídá běžnému vyhodnocování v moderních imperativních jazycích.

Ve stroji LuaTEX je možné programovat také v imperativním programovacím jazyce Lua. Jazyk Lua sice zmíněnými neduhy plain TEXu netrpí, ale komunikace mezi TEXem a Luou není přímočará a při předávání dat dochází ke ztrátě důležitých informací, jako jsou kategorie TEXových znaků.

Programovací jazyk expl3 nabízí zlatou střední cestu a umožňuje uživatelům programovat v TEXu způsobem, na který jsou zvyklí z moderních imperativních programovacích jazyků.

V tomto článku představuji modul l3seq jazyka expl3, který poskytuje datovou strukturu seznamu. Možnosti modulu demonstruji na sazbě textu české lidové písně *Když jsem já sloužil*. Implementaci v jazyce expl3 porovnávám s implementací v plain TEXu.

Klíčová slova: LATEX3, expl3, l3seq, xparse, plain TEX, OpTEX, texty písní

TEX je strojový kód světa digitální sazby, který programátorům poskytuje minimum vysokoúrovňových abstrakcí. V předchozím článku [1, sekce 2] jsem představil vysokoúrovňový programovací jazyk expl3 [2, 3], díky němuž mohou uživatelé programovat v TEXu způsobem, na který jsou zvyklí z moderních imperativních programovacích jazyků. V dalším článku jsem představil modul l3keys jazyka expl3 [4, sekce 3.2], jenž umožňuje přípravu datových sběrnic. V tomto článku představuji modul l3seq jazyka expl3, který poskytuje datovou strukturu seznamu. Možnosti modulu demonstruji na sazbě textu české lidové písně *Když jsem já sloužil* [5]. Implementaci v jazyce expl3 porovnávám s implementací v plain TEXu.

Nejprve v sekci 1 popisuji algoritmus pro generování textu písně. V sekci 2 na straně 155 algoritmus implementuji pomocí modulu l3seq jazyka expl3. Následně v sekci 3 na straně 157 vytvářím uživatelské rozhraní pomocí LATEXového balíčku xparse. Dále v sekci 4 na straně 158 ukazuji vysázený text písně. Nakonec v sekci 5 na straně 159 algoritmus implementuji v plain TEXu a porovnávám obě implementace. V sekci 6 na straně 163 shrnuji poznatky z článku a jejich přínos pro čtenáře.

DOI: 10.5300/2023-3-4/153

1 Algoritmus pro generování textu písně

V této sekci představuji českou lidovou píseň *Když jsem já sloužil* [5]. Nejprve píseň zasazuji do historického kontextu a popisuji strukturu textu písně. Následně popisuji algoritmus pro generování textu písně.

Píseň *Když jsem já sloužil* je autobiografie mladého muže, který sloužil devět let jako zemědělský dělník. Vypravěč zmiňuje robotu a omezování sňatků vrchností, což děj zasazuje do dob Rakouska-Uherska před zrušením nevolnictví roku 1781.

Text písně se dělí na sloky, které popisují jednotlivé roky vypravěčovy služby. První věta každé sloky udává rok služby a obdrženou odměnu, např.: „Když jsem já sloužil to *3. léto*, vysloužil jsem si *husičku* za to.“ Následuje dlouhé souvětí s popisem odměn za dosavadní roky služby počínaje současným rokem a konče prvním rokem, např.: „A ta husa chodí bosa a ta kačka bláto tlačká a to kuře krákoře, běhá po dvoře.“ Sloku zakončuje vypravěč domněnkou o aktivitách své partnerky, např.: „Má panenka pláče doma v komoře.“

Pokud si odměny ve čtvrtém pádu (např. „husičku“) a popisy odměn (např. „ta husa chodí bosa“) uložíme do dvou seznamů `čtvrté_pády` a `popisy` délky $N = 9$, můžeme text písně vygenerovat Algoritmem 1. V algoritmu je použita funkce **zip**, která postupně navrací prvky obou seznamů jako dvojice.

Algoritmus 1: Text písně *Když jsem já sloužil*

```
   Data: Seznamy čtvrté_pády a popisy délky N
 1 i ← 1; popisy_pozpátku ← []
 2 for čtvrtý_pád, popis in zip(čtvrté_pády, popisy) do
 3     if i < N then
 4         print "když jsem já sloužil to " + i + ". léto"
 5     else
 6         print "když jsem já sloužil poslední léto"
 7     print "vysloužil jsem si " + čtvrtý_pád + " za to"
 8     j ← i; popisy_pozpátku ← [popis] + popisy_pozpátku
 9     for popis_pozpátku in popisy_pozpátku do
10         if j < N then print "a " + popis_pozpátku
11         else print popis_pozpátku
12         j ← j − 1
13     if i < N then
14         print "má panenka pláče doma v komoře"
15     else
16         print "má panenka stele postel v komoře"
17     i ← i + 1
```

2 Implementace algoritmu pomocí modulu l3seq

V této sekci rozpracujeme soubor `kdyz-jsem-ja-slouzil.sty`, který bude implementovat Algoritmus 1 pomocí modulu l3seq jazyka expl3. Hotový soubor `kdyz-jsem-ja-slouzil.sty` můžeme stáhnout online [6].

```
\ProvidesExplPackage
  {kdyz-jsem-ja-slouzil}{2023-08-28}{1.0.0}%
  {Balíček pro sazbu textu lidové písně ,,Když jsem já sloužil``}

% Globální proměnné
\seq_new:N \g_kdyzjsemjaslouzil_ctvrte_pady_seq
\seq_new:N \g_kdyzjsemjaslouzil_popisy_seq

% Lokální proměnné
\int_new:N \l_kdyzjsemjaslouzil_aktualni_sloka_int
\int_new:N \l_kdyzjsemjaslouzil_posledni_sloka_int
\bool_new:N \l_kdyzjsemjaslouzil_posledni_sloka_bool
\seq_new:N \l_kdyzjsemjaslouzil_popisy_pozpatku_seq

% Funkce pro sazbu textu celé písně
\cs_new:Nn
  \kdyzjsemjaslouzil_vysazej_pisnicku:
  {
    % Nastav počáteční hodnoty proměnných i a N.
    \int_set:Nn \l_kdyzjsemjaslouzil_aktualni_sloka_int { 1 }
    \int_set:Nn \l_kdyzjsemjaslouzil_posledni_sloka_int {
      \seq_count:N \g_kdyzjsemjaslouzil_popisy_seq }
    % Současně procházej hodnoty polí čtvrté_pády a popisy a volej
    % nad nimi pomocnou funkci pro sazbu textu jedné sloky.
    \seq_map_pairwise_function:NNN
      \g_kdyzjsemjaslouzil_ctvrte_pady_seq
      \g_kdyzjsemjaslouzil_popisy_seq
      \kdyzjsemjaslouzil_vysazej_sloku:nn
    % Vymaž pomocné pole s popisy v opačném pořadí.
    \seq_clear:N \l_kdyzjsemjaslouzil_popisy_pozpatku_seq
  }

% Funkce pro sazbu textu jedné sloky
\cs_new:Nn
  \kdyzjsemjaslouzil_vysazej_sloku:nn
  { % Nastav pomocnou proměnnou na pravdivostní hodnotu i = N,
    % která udává, jestli sázíme poslední sloku.
```

```
\bool_set:Nn
  \l_kdyzjsemjaslouzil_posledni_sloka_bool
  {
    \int_compare_p:nNn
      { \l_kdyzjsemjaslouzil_aktualni_sloka_int }
      =
      { \l_kdyzjsemjaslouzil_posledni_sloka_int }
  }
% Vysázej rok služby a obdrženou odměnu.
Když~jsem~já~sloužil~
\bool_if:NTF
  \l_kdyzjsemjaslouzil_posledni_sloka_bool
  { poslední~léto }
  {
    to~
    \int_use:N
      \l_kdyzjsemjaslouzil_aktualni_sloka_int
    .~léto
  }
, \\ vysloužil~jsem~si~#1~za~to. \\

% Vysázej popisy odměn za dosavadní roky služby.
\seq_put_left:Nn
  \l_kdyzjsemjaslouzil_popisy_pozpatku_seq
  { #2 }
\seq_map_indexed_inline:Nn
  \l_kdyzjsemjaslouzil_popisy_pozpatku_seq
  {
    % Při popisu odměny pro současný rok začni verzálkou.
    \int_compare:nTF
      { ##1 = 1 }
      {
        % V poslední sloce vynech počáteční spojku „a".
        \bool_if:NTF
          \l_kdyzjsemjaslouzil_posledni_sloka_bool
          {
            \str_uppercase:f { \tl_head:n { ##2 } }
            \tl_tail:n { ##2 }
          }
          { A~##2 }
      }
      { a~##2 }
```

```
      % Při popisu odměny pro první rok přidej čárku.
      \int_compare:nNnT
        { ##1 }
        =
        { \l_kdyzjsemjaslouzil_aktualni_sloka_int }
        { , } \\
    }
  % Vysázej vypravěčovu domněnku o aktivitách partnerky.
  má~panenka~
  \bool_if:NTF
    \l_kdyzjsemjaslouzil_posledni_sloka_bool
    { stele~postel~ }
    { pláče~doma~ }
  v~komoře. \par
  \int_incr:N
    \l_kdyzjsemjaslouzil_aktualni_sloka_int
}
```

3 Uživatelské rozhraní pomocí LaTeXového balíčku xparse

Programovací jazyk expl3 má pro uživatele TeXu poměrně neobvyklou syntax. V této sekci ukončíme soubor `kdyz-jsem-ja-slouzil.sty` a pomocí LaTeXového balíčku vytvoříme uživatelské rozhraní, které bude pro uživatele přirozenější.

```
\RequirePackage { xparse }
\NewDocumentEnvironment
  { kdyz-jsem-ja-slouzil }
  { }
  {
    \NewDocumentCommand
      \sloka
      { m m }
      {
        \seq_put_right:Nn
          \g_kdyzjsemjaslouzil_ctvrte_pady_seq
          { ##1 }
        \seq_put_right:Nn
          \g_kdyzjsemjaslouzil_popisy_seq
          { ##2 }
      }
  }
  { \kdyzjsemjaslouzil_vysazej_pisnicku: }
```

4 Sazba textu písně

V této sekci vytvoříme a vysázíme ukázkový soubor `priklad-latex.tex`. Soubor nejprve načte soubor `kdyz-jsem-ja-slouzil.sty`, který jsme připravili v sekcích 2 a 3, a následně vysází text písně *Když jsem já sloužil*.

```
\documentclass{article}
\usepackage[czech]{babel}
\usepackage[T1]{fontenc}
\usepackage{lmodern,multicol,parskip}
\usepackage{kdyz-jsem-ja-slouzil}
\begin{document}
\begin{multicols}{2}
\begin{kdyz-jsem-ja-slouzil}
\sloka {kuřátko}   {to kuře krákoře, běhá po dvoře}
\sloka {kachničku} {ta kačka bláto tlačká}
\sloka {husičku}   {ta husa chodí bosa}
\sloka {vepříka}   {ten vepř jako pepř}
\sloka {telátko}   {to tele hubou mele}
\sloka {kravičku}  {ta kráva mléko dává}
\sloka {volečka}   {ten vůl jako kůl}
\sloka {botičky}   {ty boty do roboty}
\sloka {děvčátko}  {to děvčátko jak poupátko}
\end{kdyz-jsem-ja-slouzil}
\end{multicols}
\end{document}
```

Soubor zpracujeme příkazem `lualatex priklad-latex` a obdržíme tento výstup:

Když jsem já sloužil to 1. léto,
vysloužil jsem si kuřátko za to.
A to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 2. léto,
vysloužil jsem si kachničku za to.
A ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 3. léto,
vysloužil jsem si husičku za to.
A ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 4. léto,
vysloužil jsem si vepříka za to.
A ten vepř jako pepř
a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 5. léto,
vysloužil jsem si telátko za to.
A to tele hubou mele
a ten vepř jako pepř

a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 6. léto,
vysloužil jsem si kravičku za to.
A ta kráva mléko dává
a to tele hubou mele
a ten vepř jako pepř
a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 7. léto,
vysloužil jsem si volečka za to.
A ten vůl jako kůl
a ta kráva mléko dává
a to tele hubou mele
a ten vepř jako pepř
a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil to 8. léto,
vysloužil jsem si botičky za to.
A ty boty do roboty
a ten vůl jako kůl
a ta kráva mléko dává
a to tele hubou mele
a ten vepř jako pepř
a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka pláče doma v komoře.

Když jsem já sloužil poslední léto,
vysloužil jsem si děvčátko za to.
To děvčátko jak poupátko
a ty boty do roboty
a ten vůl jako kůl
a ta kráva mléko dává
a to tele hubou mele
a ten vepř jako pepř
a ta husa chodí bosa
a ta kačka bláto tlačká
a to kuře krákoře, běhá po dvoře,
má panenka stele postel v komoře.

5 Implementace algoritmu pomocí plain TEXu

V této sekci vytvoříme soubor `kdyz-jsem-ja-slouzil.opm`, který bude implementovat Algoritmus 1 v jazyce plain TEX. Pro usnadnění použijeme formát OpTEX, který přidává nad rámec plain TEXu podporu pro jmenné prostory [7, sekce 2.2.3], jež využijeme při definici lokálních proměnných, a podporu pro vícesloupcovou sazbu [7, sekce 1.5.2], kterou využijeme v ukázkovém souboru.

```
\_codedecl
\sequence {Balíček pro sazbu textu lidové písně ,,Když jsem já
  sloužil`` <1.0.0>}
\_namespace {kdyzjsemjaslouzil}

% Lokální proměnné
\newcount \.pocetslok
\newcount \.cislosloky
\newcount \.cisloradku
```

```
% Pomocné funkce pro práci se seznamy, které jsou zde
% implementovány jako hašová tabulka.
\def \.sdef #1{\expandafter \def \csname #1\endcsname}
\def \.suse #1{\csname #1\endcsname}
% Pomocná funkce pro změnu prvního písmene na verzálku.
\def \.prvnivelke #1{\uppercase \expandafter {#1}}

% Funkce pro uložení jedné sloky
\def \sloka #1#2{%
  \advance \.pocetslok 1
  \.sdef {čtvrtý pád:\the \.pocetslok}{#1}%
  \.sdef {popis:\the \.pocetslok}{#2}%
}

% Funkce pro sazbu textu celé písně
\def \vysazej_pisnicku {%
  \loop
  \ifnum \.cislosloky < \.pocetslok
    % Vysázej rok služby a obdrženou odměnu.
    \advance \.cislosloky 1
    Když jsem já sloužil
    \ifnum \.cislosloky = \.pocetslok
      poslední
    \else
      to \the \.cislosloky.~%
    \fi
    léto,\nl
    vysloužil jsem si
    \.suse {čtvrtý pád:\the \.cislosloky}
    za to.\nl
    % Vysázej popisy odměn za dosavadní roky služby.
    {%
      \.cisloradku = \.cislosloky
      \loop
      % Při popisu odměny pro současný rok začni verzálkou.
      \ifnum \.cisloradku = \.cislosloky
        % V poslední sloce vynech počáteční spojku „A".
        \ifnum \.cislosloky < \.pocetslok A \else
          \expandafter \expandafter \expandafter
          \expandafter \expandafter
          \.prvnivelke
        \fi
```

```
      \else a \fi
      \.suse {popis:\the \.cisloradku}%
      % Při popisu odměny pro první rok přidej čárku.
      \ifnum \.cisloradku = 1 , \fi \nl
      \advance \.cisloradku -1
      \ifnum \.cisloradku > 0 \repeat
    }%
    % Vysázej vypravěčovu domněnku o aktivitách partnerky.
    má panenka
    \ifnum \.cislosloky = \.pocetslok
      stele postel
    \else
      pláče doma
    \fi
    v~komoře.
    \par \medskip
  \repeat
}%
\_endnamespace
\_endcode
```

Následně vytvoříme ukázkový soubor `priklad-optex.tex`. Soubor nejprve načte soubor `kdyz-jsem-ja-slouzil.opm`, který jsme připravili v této sekci, a následně vysází text písně *Když jsem já sloužil*.

```
\input kdyz-jsem-ja-slouzil.opm
\chyph
\fontfam [lm]
\parindent 0pt
\begmulti 2
\sloka {kuřátko}   {to kuře krákoře, běhá po dvoře}
\sloka {kachničku} {ta kačka bláto tlačká}
\sloka {husičku}   {ta husa chodí bosa}
\sloka {vepříka}   {ten vepř jako pepř}
\sloka {telátko}   {to tele hubou mele}
\sloka {kravičku}  {ta kráva mléko dává}
\sloka {volečka}   {ten vůl jako kůl}
\sloka {botičky}   {ty boty do roboty}
\sloka {děvčátko}  {to děvčátko jak poupátko}
\vysazej_pisnicku
\endmulti
\bye
```

Soubor zpracujeme příkazem `optex priklad-optex` a obdržíme výstup ze sekce 4.

Nyní porovnáme implementaci v jazyce expl3 ze sekcí 2 a 3 s implementací v jazyce plain TEX z této sekce. Implementace v jazyce expl3 má takřka dvojnásobnou délku. Délkový rozdíl je způsoben především rozvláčností jazyka expl3, ne složitostí kódu, a dopad na čitelnost kódu je tedy sporný.

Implementace seznamů v plain TEXu není přímočará a vyžaduje opatrnou práci s expanzí. Implementace z této sekce se proto seznamům vyhýbá a nahrazuje je hašovou tabulkou. Díky tomu je implementace v plain TEXu stručná a čitelná, ale není obecná: programátor se musí přizpůsobit omezením plain TEXu a připravit kód na míru řešenému problému, zatímco jazyk expl3 disponuje bohatou knihovnou datových struktur, které je možné použít pro řešení různých problémů.

Recenzent článku navrhl ještě jednodušší řešení v idiomatickém plain TEXu, které implementaci algoritmu nevyčleňuje do samostatného balíčku a které jednotlivé sloky okamžitě sází, aniž by si je předtím ukládalo do datové struktury. Navržené řešení s drobnými úpravami uvádím se svolením recenzenta:

```
\fontfam [lm]
\parskip = 6pt plus 2pt
\parindent = 0pt
\def \sloka #1#2#3{\par
  \ifx^#3^\else
    \edef \popisy {%
      #3\ifx \popisy \empty \else \nl a \fi \popisy
    }%
  \fi
  Když jsem já sloužil #1 léto,\nl
  vysloužil jsem si #2 za to.\nl
  \A \popisy, běhá po dvoře,\nl
  má panenka \cosidela v komoře.\par
}
\def \popisy {}
\def \cosidela {pláče doma }   \def \A {A }
\sloka {to první}  {kuřátko}   {to kuře krákoře}
\sloka {to druhé}  {kachničku} {ta kačka bláto tlačká}
\sloka {to třetí}  {husičku}   {ta husa chodí bosa}
\sloka {to čtvrté} {vepříka}   {ten vepř jako pepř}
\sloka {to páté}   {telátko}   {to tele hubou mele}
\sloka {to šesté}  {kravičku}  {ta kráva mléko dává}
\sloka {to sedmé}  {volečka}   {ten vůl jako kůl}
\sloka {to osmé}   {botičky}   {ty boty do roboty}
\def \cosidela {stele postel } \def \A {}
\sloka {poslední}  {děvčátko}  {To děvčátko jak poupátko}
\bye
```

6 Závěr

Díky programovacímu jazyku expl3 mohou uživatelé TEXu programovat způsobem, na který jsou zvyklí z moderních imperativních programovacích jazyků.

V článku jsem představil modul l3seq jazyka expl3, který poskytuje datovou strukturu seznamu. Možnosti modulu jsem demonstroval na sazbě textu české lidové písně *Když jsem já sloužil*. Dále jsem ukázal, že text písně je možné stručně a čitelně vysázet také v jazyce plain TEX, pokud se programátor přizpůsobí omezenému počtu základních datových struktur v plain TEXu.

Odkazy

1. NOVOTNÝ, Vít. Vysokoúrovňové jazyky pro TEX. *Zpravodaj CSTUGu*. 2022, **32**(1–4), 35–48. Dostupné z DOI: 10.5300/2022-1-4/35.
2. THE LATEX PROJECT. *The expl3 package and LATEX3 programming* [online]. CTAN, 2023-08-29 [cit. 2023-10-05]. Dostupné z: https://tug.ctan.org/macros/latex/contrib/l3kernel/expl3.pdf.
3. THE LATEX PROJECT. *The LATEX3 interfaces* [online]. CTAN, 2023-08-29 [cit. 2023-10-05]. Dostupné z: https://tug.ctan.org/macros/latex/contrib/l3kernel/interface3.pdf.
4. STARÝ N., Vít. Nápadovník jmen pro tvůrčí psaní v LuaTEXu. *Zpravodaj CSTUGu*. 2023, **33**(1–2), 3–38. Dostupné z DOI: 10.5300/2023-1-2/3.
5. HROUDOVÁ, Eva. *Když jsem já sloužil* [online]. ProMaminky.cz, 2015-06-20 [cit. 2023-08-13]. Dostupné z: https://www.promaminky.cz/pisnicky/lidove-36/kdyz-jsem-ja-slouzil-287.
6. STARÝ N., Vít. *Sazba textu české lidové písně „Když jsem já sloužil“ pomocí modulu l3seq jazyka expl3: Release The latest version* [online]. GitHub, 2023-09-10 [cit. 2023-09-10]. Dostupné z: https://github.com/Witiko/typesetting-czech-folksong-with-l3seq/releases/tag/latest.
7. OLŠÁK, Petr. *OpTEX: Format Based on Plain TEX and OPmac* [online]. CTAN, 2023-05-25 [cit. 2023-09-18]. Dostupné z: https://mirrors.ctan.org/macros/optex/doc/optex-doc.pdf. Verze 1.12.

Summary: Typesetting Lyrics of Czech Folksong “Když jsem já sloužil” using the l3seq Module of Expl3 Language

The language of plain TEX was developed for typesetting books and only became a Turing-complete programming language at the end of its development. Whereas writing and designing documents is straightforward in plain TEX, programming is difficult due to a lack of basic data structures and the delayed macro expansion, which is different from modern imperative programming languages.

In the LuaTEX engine, authors can also program in the imperative programming language of Lua. Although Lua does not share the limitations of plain TEX, passing data between TEX and Lua is not straightforward and important information such as token category codes are lost in transit.

The expl3 programming language combines the best of both worlds and allows authors to program in TEX in a way that is similar to modern imperative programming languages.

In this article, I introduce the l3seq module of the expl3 language that provides the list data structure. Using l3seq, I typeset the lyrics of the Czech folksong *Když jsem já sloužil.* I also compare the l3seq implementation with plain TEX.

Keywords: LATEX3, expl3, l3seq, xparse, plain TEX, OpTEX, song lyrics

Vít Starý Novotný, witiko@mail.muni.cz