

Zpravodaj Československého sdružení uživatelů TeXu

Jaroslav Hajtmar

ScanCSV – Lua knihovna pro zpracování CSV souborů ConTeXtem a
LuaLaTeXem

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 22 (2012), No. 2, 76–90

Persistent URL: <http://dml.cz/dmlcz/149951>

Terms of use:

© Československé sdružení uživatelů TeXu, 2012

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>

ScanCSV – Lua knihovna pro zpracování CSV souborů ConT_EXtem a LuaL_AT_EXem

JAROSLAV HAJTMAR

Tento článek popisuje možnosti použití jazyka Lua pro vytvoření knihovny Lua funkcí, které mohou zajímavým způsobem zpřístupnit ConT_EXtu, LuaT_EXu a LuaL_AT_EXu textové databázové údaje, uložené v CSV souborech. Prioritou při tvorbě popisované luaknihovny bylo, aby mohla být používána i uživateli bez sebemenších znalostí jazyka Lua. Kdo zvažuje, že si „něco začne“ s Lua, má příležitost zjistit, jak Lua funguje. Kdo chce zůstat „ryzím T_EXistou“, má možnost používat popisovanou knihovnu formou „blackboxu“, tj. do zdrojového textu ConT_EXtu (PlainT_EXu, L_AT_EXu) zapsat několik řádků Lua kódu, zkompilovat zdroják odpovídajícím formátem a koukat, jak to celé krásně funguje. Pro vážnější zájemce jsem připravil ke stažení řadu pokročilých ukázek, demonstrujících zajímavé možnosti praktického použití knihovny. Ačkoliv je Lua knihovna primárně určena pro použití v ConT_EXtu, přichystal jsem řadu ukázek použití v L_AT_EXu, který bude zřejmě čtenáři tohoto článku preferován.

1. Úvod

V tomto příspěvku se pokusím popsat svůj začátečnický pokus o vytvoření luaknihovny, s jejíž pomocí lze snadným způsobem zpřístupnit ConT_EXtu, LuaT_EXu (Plainu) a LuaL_AT_EXu textové databázové údaje, uložené v CSV souborech.

ConT_EXtem budu v celém příspěvku mínit verzi MKIV, která je postavena na jazyce Lua (narozdíl od původní verze MKII). Aplikace uvedených Lua funkcí umožňuje snadné vytváření dokumentů, obsahujících hromadná data z jednoduchých CSV databází, která jsou vhodným způsobem naformátována. Použití je velmi pestré, např. tisk hromadných dopisů, různých formulářů, vysvědčení, pozvánek, průkazek, vizitek, kartiček atd.

Při programování knihovny byl kladen velký důraz na to, aby funkce byly široce použitelné nejen v ConT_EXtu (pro který byly primárně napsány), ale i v LuaT_EXu a LuaL_AT_EXu. Hlavním cílem bylo také to, aby uživatel (Plainista, L_AT_EXista či ConT_EXtista) těchto Lua funkcí nemusel mít vůbec žádné znalosti jazyka Lua, a přitom mohl velmi dobře prakticky tyto funkce používat. Ve skutečnosti se ve zdrojovém T_EXovém kódu objevuje volání Lua kódu jen na začátku dokumentu, samotné zpřístupnění CSV dat je realizováno prostřednictvím T_EXových maker.

2. CSV formát

Většina čtenářů jistě zná formát CSV (comma-separated values, hodnoty oddělené čárkami). Jde se o velmi jednoduchý souborový formát určený pro výměnu tabulkových dat. Obecně CSV soubor sestává z řádků (záznamů), v nichž jsou jednotlivé položky (pole) odděleny oddělovačem (většinou čárkou nebo středníkem). Hodnoty takto oddělených položek v řádku mohou být navíc vymezeny dalšími znaky (standardně jsou uzavírány do uvozovek). To umožňuje, aby text položek obsahoval i oddělovač polí (tedy např. čárku či středník). Standardně CSV formát umožňuje vmezovačem (např. uvozovkami) vymežit pouze ty položky, které obsahují oddělovač položek, navíc umožňuje i to, aby jednotlivé položky obsahovaly i znak vmezovače (tj. uvozovky) atd. Jednoduchý algoritmus, který jsem navrhl pro rozparsování položek ovšem předpokládá, že pokud již začneme ve svých CSV souborech používat vmezovač, musí jím být vymezeny úplně všechny položky. Navíc nesmí být znak pro vmezovač obsažen uvnitř žádné ze sloupcových položek. Parsovací funkci `ParseCSVdata` lze samozřejmě zobecnit.

Za hojné používání CSV souborů se středníkem (byť pod názvem CSV) vděčí tento formát Microsoft Excelu v české verzi Microsoft Windows. Standardní oddělovač (středník) lze změnit (v místním a jazykovém nastavení).

Ukázky CSV tabulek na něž se budu následně v textu i zdrojovém kódu ukázkového dokumentu odvolávat:

Příklad 1 (`zaci.csv`) *CSV tabulka bez záhlaví. Položky oddělené středníkem, vmezovač není:*

```
1;Petr;Novák;19.5.1989;m;Nymburk;U Brány 7
2;Jan;Novotný;5.7.1991;m;Praha;Uhlířská 178
...
```

Příklad 2 (`studenti.csv`) *CSV tabulka se záhlavím. Položky oddělené středníkem, vmezovač není:*

```
Prijmeni;Jmeno;DatumNarozeni;Pohlavi;Mesto;PSC;Ulice
Novák;Jan;14.10.1997;m;Zbečno;27024;Farní 21
Pospíšilová;Hana;4.1.1996;ž;Zábřeh;78901;Studénky 420
...
```

Příklad 3 (`zamestnanci.csv`) *CSV tabulka se záhlavím. Oddělovačem položek je tentokrát čárka, vmezovačem jsou uvozovky, uvnitř nichž se může vyskytovat separátor polí (čárka):*

```
"PorCis","Prijmeni","Jmeno","Pohlavi","DatumNarozeni","Bydliste"
"1","Májková","Barbora","ž","2.4.1995","Mírov 96, 789 53 Mírov"
"2","Nová","Zuzana","ž","13.1.1995","U Dráhy 8, 789 01 Zábřeh"
...
```

3. Použití CSV tabulek v T_EXu

V r. 2005 jsem náhodou objevil makro `scanbase.tex` Petra Olšáka, které mne velmi zaujalo svou praktičností. Petr Olšák jej následně zobecnil a zmodifikoval tak, aby se dalo použít i pro zpracování CSV souborů nejen v Plainu, ale i v L^AT_EXu (modifikaci v r. 2008 navrhl Jaromir Kuben) a ConT_EXtu MKII. Pro svoji práci v Plainu jsem používal makro `scancsv.tex` hezkou řadu let, makro `scancsv-context.tex` občas použiju v ConT_EXtu MKII i dnes.

Původní makro Petra Olšáka mi bylo tak zásadní inspirací pro tvorbu mých Lua funkcí, že jsem ve své Lua aplikaci ponechal i stejné názvy hlavních T_EXových maker. Důležitým důvodem ovšem bylo také to, že jsem chtěl, aby se mi snáze předělávaly vlastní starší aplikace, založené na použití původního Olšákova makra. V původním zdrojovém textu pak stačilo udělat jen pár drobných úprav a zkompilovat jej ConT_EXtem MKIV. Jistou nepříjemností při používání CSV souborů v ConT_EXtu (LuaT_EXu, LuaL^AT_EXu) je, že pro správné fungování aplikací je požadováno UTF-8 kódování, zatímco CSV tabulky vytvořené v Microsoft Excelu jsou defaultně kódované CP 1250. Překódování naštěstí snadno zvládne každý solidní textový editor.

4. Knihovna `scancsv.lua`

4.1. Princip fungování luaknihovny

Lua knihovna `scancsv.lua` je tvořena sérií funkcí, které zpřístupňují data v CSV tabulkách (kódovaných UTF-8). Ačkoliv se budu neustále odvolávat na použití knihovny `scancsv.lua` v ConT_EXtu MKIV, musím poznamenat, že funkce byly navrženy vesměs tak, aby fungovaly i v LuaT_EXu či LuaL^AT_EXu (bylo testováno pouze na jednoduchých příkladech).

Princip fungování knihovny vychází z již zmiňovaného T_EXového algoritmu Petra Olšáka. Postupně jsou data obsažená v řádcích CSV tabulky prostřednictvím programových Lua cyklů rozparsována a zapracována T_EXových maker. Makra jsou v průběhu zpracování automaticky nadefinována (vygenerována) a následně mohou být použita ve zdrojovém textu ConT_EXtu (LuaL^AT_EXu či LuaT_EXu). První verze knihovny vyžadovala jistě znalosti Lua a určitý způsob umísťování Lua kódu do zdrojového textu. Vzhledem k tomu, že hlavní prioritou bylo, aby mohl knihovnu používat kterýkoliv T_EXista, aniž by měl sebemenší znalosti Lua, doznala knihovna nakonec takových změn, že ve zdrojovém dokumentu je umístěn Lua kód pouze na začátku dokumentu ¹. Kód slouží k načtení luaknihovny, což bude předvedeno v ukázkových příkladech pro ConT_EXt a LuaL^AT_EX.

¹Výjimkou může být volání knihovní funkce `filelineaction(csvfile,fromrow,torow)` v LuaL^AT_EXu, která zajistí, že se načte určitý rozsah řádků CSV tabulky (`fromrow – torow`). Pro zpracování celého CSV souboru je i v LuaL^AT_EXu u připraveno samostatné T_EXové makro.

Vzhled výsledného dokumentu (tiskové sestavy) je předurčen libovolně na-
definovaným \TeX ovým makrem `\lineaction`. V tomto makru mohou být sa-
mozřejmě použita všechna makra automaticky vygenerovaná funkcemi knihovny.
Sestava může mít libovolný vzhled, nejčastěji tabulkový nebo formulářový. Po
ukončení makra `\lineaction` se v cyklu automaticky načte další řádek CSV
tabulky, znovu se naplní všechna vygenerovaná makra texty sloupcových položek
z tohoto řádku a znovu se spustí `\lineaction`. To se opakuje tak dlouho, dokud
není zpracována celá tabulka nebo její část (určená dvojicí parametrů `fromrow`
a `torow`). Knihovna makro `\lineaction` implicitně nadefinuje, takže pokud už-
ivatel vlastní definicí toto makro nevytvoří, použije se implicitní nastavení, které
udělá to, že se vypíše všechny řádky CSV tabulky v jakémsi „zhuštěném“ formátu.

Já většinou nadefinuji makro `\printaction` (ve kterém se používají auto-
maticky vygenerovaná makra), které určuje vzhled tiskové sestavy a toto makro
následně vhodným způsobem zakomponuji do makra `\lineaction`. Volitelně si
může uživatel nadefinovat také makra `\blinehook` (háčkové makro, které se vždy
provede automaticky před provedením makra `\lineaction` tj. před zpracováním
každého řádku CSV tabulky) a `\elinehook` (provede se vždy po `\lineaction`)
a dále makra `\bfilehook` a `\efilehook`, což jsou makra, která se provedou před
začátkem zpracování a na konci zpracování celého CSV souboru. Pomocí těchto
„háčků“ (které jsou implicitně knihovnou nastaveny na `\relax`) lze nastavovat
různá záhlaví a zápatí skupin, odstránkování atd. Rád bych upozornil na to, že
nic nebrání zpracování i několika různých CSV souborů v jednom dokumentu
atd.

4.2. \TeX ová makra pro zpřístupnění sloupcových dat CSV tabulky

Nejdůležitějšími makry pro použití jsou makra, která zpřístupňují sloupcová data
v CSV tabulce. Názvy těchto maker jsou odvozeny od názvů sloupců v excelovské
tabulce. Data ve sloupcích A, B, C, atd. jsou k dispozici v makerch s názvy `\cA`,
`\cB`, `\cC`, atd.

K vytváření názvů sloupců je potřebná funkce `ar2xls(number)` (Arabic to
XLS). Ta převádí pořadové číslo sloupce na název excelovského sloupce. Nepřed-
pokládá se, že by někdo potřeboval více než 703 sloupců (tj. sloupce nad rozsah
A – ZZ). Funkci `ar2xls` by bylo nutné v tom případě upravit.

Pokud by někomu místo excelovského formátu názvů sloupců spíše vyhovovaly
názvy, založené na římských číslech, tj. `\cI`, `\cII`, `\cIII`, `\cIV`, atd., lze to snadno
zařídit nastavením Lua proměnné `UserColumnNumbering` na jinou hodnotu než
XLS (defaultní hodnota). V tomto případě se použije pro vytváření názvů sloupců
funkce `ar2rom(number)`. Počet sloupců v CSV tabulce není nijak limitován (něk-
teré verze Excelu však mohou např. obsahovat maximálně 256 sloupců). Pro
CSV tabulky s větším počtem sloupců může být ovšem používání excelovského
či římského označování sloupců poněkud nepřehledné. Knihovna proto vygene-

ruje i \TeX ová makra se stejnými názvy, jako jsou názvy polí v prvním řádku CSV tabulky (záhlaví). Uživatel tak nemusí pracně „odpočítávat“ a identifikovat jednotlivé sloupce, postačí mu znát název sloupce a makro tohoto jména mu zpřístupní potřebná data. Je na místě dodat, že tato makra se vygenerují dokonce i v případě, že první řádek je „datový“ (tj. není tzv. záhlavím), přičemž tato volba je nastavena defaultně. Názvy maker v tomto případě budou nejspíš nesrozumitelné, neboť „nevhodné“ názvy sloupců jsou (z hlediska striktních požadavků na názvy \TeX ových maker) vždy ošetřovány funkcí `TMN(string)` (\TeX Macro Name). Vstupním parametrem této funkce je textový řetězec ze záhlaví daného sloupce CSV tabulky. V tomto řetězci se nahradí všechny diakritické znaky české abecedy (pro jiné jazyky je nutná modifikace makra) znaky bez diakritiky, arabské číslice se nahradí římskými, nealfabetické znaky se nahradí zástupným znakem "x". Nakonec se zkrátí délka makra na 20 znaků (lze zvýšit). Výstupem funkce je název makra, který může být pro uživatele nesrozumitelný, ale \TeX u by neměl vadit. Pokud tedy zamýšlíte využívat toho, že knihovna vygeneruje „pěkné“ názvy maker ze záhlaví CSV tabulky, měli byste se při vytváření záhlaví vyvarovat používání všech znaků, které se nemohou vyskytovat v názvech \TeX ových maker. Nutno dodat, že nastavením Lua proměnné `CSVHeader` na hodnotu `true` docílíme toho, že knihovna první řádek CSV tabulky nepovažuje za datový řádek. Data z tohoto řádku pak knihovna nezpracovává (nevypisuje je). Tento řádek se pak ani nezapočítává do počtu zpracovávaných řádků tabulky. Defaultní hodnota Lua proměnné `CSVHeader` je ovšem `false`, což nebrání knihovně vytvořit z prvního řádku názvy sloupcových maker. Všechny řádky jsou ovšem brány automaticky jako datové a zpracovávají se.

Pokud použijeme ke zpracování CSV soubor, uvedený v příkladu 2, budou sloupcová data zpřístupněna jednak v makrech `\cA`, `\cB`, `\cC`, ..., `\cH`, ale také v makrech `\Prijmeni`, `\Jmeno`, `\DatumNarozeni` atd., zatímco pokud použijeme ke zpracování CSV soubor, uvedený v příkladu 1, budou sloupcová data zpřístupněna v makrech `\I=\cA` (číslice 1 je nahrazena římskou číslicí), `\Petr=\cB`, `\Novak=\cC` (písmeno „á“ nahrazeno písmenem „a“), podivně vyhlížející makro `\IIIXVxIIIXVIIIIIX=\cD` vzniklo nahrazením arabských cifer a teček.²

Všechna výše uvedená \TeX ová makra vznikají v průběhu cyklu zcela automaticky bez jakéhokoliv přispění uživatele, uživatel je dokonce nemusí ani v $\text{Con}\text{\TeX}$ tu definovat. Tím nastává paradoxní situace, neboť se ve zdrojovém $\text{Con}\text{\TeX}$ tovém textu se mohou objevovat např. makra `\Prijmeni`, `\Jmeno`, `\DatumNarozeni` atd., která ale nejsou nikde výše ve zdrojáku nadefinovaná.

Pokud má někdo strach, že si automaticky generované názvy \TeX ových maker nezapamatuje, mohu jej uklidnit. Pomocí Lua funkce `CSVReport()` resp. pomocí

²Sloupcové údaje jsou v tomto režimu k dispozici i prostřednictvím globálních Lua proměnných `CSV.Prijmeni`, `CSV.Jmeno`, `CSV.DatumNarozeni` atd. Jejich použití však již vyžaduje jisté znalosti Lua.

TeXového makra `\csvreport` si může uživatel kdykoliv ve zdrojovém textu ConTeXtu či Lua⁴TeXu nechat vypsat nejen řadu užitečných informací o otevřené CSV tabulce, ale mj. i názvy všech vygenerovaných maker.

Vzhledem k tomu, že jsem předpokládal, že většina CSV tabulek nebude obsahovat záhlaví s názvy sloupců, knihovna „předpokládá“, že záhlaví u tabulek nejsou. Implicitní hodnota Lua proměnné `CSVHeader` je v knihovně určena hodnotou proměnné `UserCSVHeader` (defaultně `false`), což lze v knihovně vyeditovat. Pokud by měl někdo zájem používat snadno zapamatovatelná makra (označující sloupcové položky) a přitom nemít záhlaví CSV tabulek, může to zařídit tak, že si na začátku definice makra `\lineaction` (resp. `\printaction`) pomocí `\let` vytvoří potřebná jména, a jim přiřadí hodnoty standartních excelovských (či římských) názvů maker (např. `\let\PorCislo\cA`). Způsob nastavení vlastních názvů bude později předveden v ukázkovém příkladu.

4.3. TeXová makra pro získání „systémových“ informací

Knihovna vygeneruje automaticky i další TeXová makra, která může uživatel využít pro tvorbu tiskových sestav nebo pro vypsání důležitých informací o zpracovávané CSV tabulce:

- `\csvfilename` : Jméno aktuálně otevřeného CSV souboru.
- `\numcols` : Počet sloupců CSV tabulky.
- `\numrows` : Počet aktuálně zpracovaných (vypsáných) řádků. Může se lišit od celkového počtu řádků tabulky (pokud uživatel vypisuje pouze určitý rozsah řádků).
- `\numline` : Pořadové číslo aktuálně načteného řádku. Vhodné pro použití v tiskových sestavách.
- `\csvreport` : Reportní informace o otevřeném CSV souboru (mj. názvy všech upotřebitelných TeXových maker atd.).
- `\printline` : Aktuální řádek CSV tabulky ve „zhuštěném“ tvaru.
- `\printall` : CSV tabulka ve „zhuštěném“ tvaru.

4.4. Modifikace funkčnosti knihovny

Uživatel si může knihovnu do značné míry přizpůsobit svým potřebám. Prostřednictvím několika globálních proměnných lze nastavit řadu defaultních vlastností. Proměnné mohou být přenastaveny buď přímo ve vlastním souboru knihovny, což umožní uživateli „jednou provždy“ provést nastavení vyhovující jeho standartním potřebám (např. oddělovačem položek v používaných CSV tabulkách nebude středník ale čárka). Nastavení hodnot příslušných proměnných lze provést po načtení knihovny i ve zdrojovém TeXovém souboru.

Přímo v knihovně lze nastavit defaultní hodnoty těchto proměnných:
`UserCSVSeparator` : Používaný separátor polí. Defaultní je středník.

UserCSVLeftDelimiter : Pole mohou být vymezena vymezovačem. To umožní používat znak pro separátor polí i uvnitř textu pole. Defaultní je prázdný řetězec `UserCSVLeftDelimiter=''`.

UserCSVRightDelimiter : Lze nastavit i pravý vymezovač. Defaultní hodnota je stejná jako v předchozím případě.

UserCSVHeader : Načtený CSV soubor je defaultně posuzován jako bez hlavičky (údaje v hlavičce jsou brány jako názvy sloupcových maker). Defaultně je `UserCSVHeader=false`.

UserColumnNumbering : Něco jiného než XLS nebo nedefinovaná hodnota (např. zakomentováním řádku) nastaví římské číslování. Defaultní je `XLS`.

Ve zdrojovém textu lze „přebít“ defaultní hodnoty výše uvedených proměnných (v případě, že mimořádně zpracováváme jiný typ CSV souboru než je běžné) a změnit tak chování knihovny prostřednictvím nastavení těchto proměnných:

File2Scan : Jméno používaného CSV souboru. Proměnná se může nastavit manuálně nebo se nastaví automaticky při volání funkce, jejímž parametrem je název souboru. Její nastavení umožňuje volat funkce `OpenCSVFile()`, `CSVReport()`, `lineaction()` a `filelineaction()` bez parametrů. Nastavit proměnnou lze i makrem (např. `\setfiletoscan{zaci.csv}`).

Sep : Oddělovač sloupců v CSV souboru. Defaultní hodnota separátoru je v proměnné `UserCSVSeparator`. Některé znaky bohužel nelze použít jako separátor. Použít lze rovněž makro (např. `\setsep{,}`).

Ld : Left delimiter. Sloupcové položky mohou být ohraničeny jinak zleva a jinak zprava. Defaultní hodnota je v proměnné `UserCSVLeftDelimiter`. Některé znaky nelze použít. K dispozici je makro (např. `\setld{"}`).

Rd : Right delimiter. Defaultní hodnota je v proměnné `UserCSVRightDelimiter`. Nejčastěji bývají jednotlivé položky ohraničeny znakem " (tj. vloženy do uvozovek). Makrem např. `\setrd{"}`.

CSVHeader : Hodnota `true`, znamená, že 1. řádek obsahuje záhlaví se jmény sloupců (tj. není datový). Defaultní hodnota `false` – všechny řádky jsou uvažovány jako datové. Lze užít `TeX`ové makro `\setheader`.

Další možnosti nastavení zvědavý čtenář vyčte přímo z knihovního souboru `scansv.lua`. Ten obsahuje vysvětlující komentáře, z nichž lze vyčíst princip fungování.

4.5. Praktické ukázky užití knihovny

Po dohodě s redakcí jsme ustoupili od zveřejnění zdrojového textu knihovny. Zdrojový text obsahující množství komentářů si bude moci čtenář spolu s řadou funkčních příkladů stáhnout z úložiště a následně prostudovat, vyzkoušet a otestovat. V tuto chvíli pojďme ukázat pouze výpis krátkého programu. Z něj snad bude patrné, jak knihovna funguje.

V následujícím zdrojovém textu budeme předpokládat, že používáme při práci UTF-8 kódované CSV soubory `zaci.csv`, `studenti.csv` a `zamestnanci.csv` z příkladů 1, 2 a 3. Zdrojový text příkladu je pro pochopení fungování dostatečně okomentován. Ukázka výstupu následujícího příkladu je na obr. 1 a obr. 2. Na nich jsou předvedeny poslední dvě stránky výsledného PDF souboru.

```
% Ukázka použití. Kompilace: lualatex examples-latex.tex
\documentclass{article}
\usepackage{luatextra}
\usepackage[utf8]{luainputenc}
\parindent0pt

\begin{document}
\pagestyle{empty}

% Načtení knihovny
\directlua{dofile("../scancsv-general/scancsv.lua")}

% Volitelné definice "háčků" (defaultně jsou háčky "zarelazovány")
% Hooks před a po zpracování celé tabulky:
\def\bf{filehook}{Seznam účastníků zájezdu {\bf\csvfilename}:
  \par\bigskip}
\def\ef{filehook}{\par Sestavou bylo vypsáno {\bf\numrows} účastníků.
\bigskip}

% Hooks před a po zpracování řádkové informace:
\def\bl{linehook}{Účastník č. \numline :\par} % akce před výpisem ř.
\def\el{linehook}{\par\smallskip\hrulefill\medskip\par} % po výpisu ř.

% Definice pomocných vyhodnocujících maker použitých v sestavách
\def\priponaa{\if m\pohl\else a\fi}
\def\priponai{\if m\pohl\else í\fi}

% Definice tiskové sestavy:
\def\printaction{
  pan\priponai\ \Jmeno\ \Prijmeni\par
  narozen\priponaa\ \DatumNarození\par
  bytem \bydlste
}

% Pro zpracování nyní stačí definice makra \lineaction např. takto:
% \def\lineaction{\printaction}
```

```

% Vzhledem k ukázkovému zpracování více CSV souborů příklad
% poněkud zobecníme (byť za cenu menší přehlednosti)

% Vlastní tiskové makro pro soubor zaci.csv si připravíme např. takto:
% Uvědomme si, že tento CSV soubor je "bez hlavičky" tj. není známa
% informace o názvech sloupců

\def\lineaction{
  % Kvůli přehlednosti si můžeme nastavit potřebná makra
  \let\Jmeno\cB % ve 2. sloupci tj. excelovsky sloupec B
  \let\Prijmeni\cC
  \let\DatumNarozeni\cD
  \let\pohl\cE% pro fungování pomocných maker \priponaa a \priponai
  \def\bydliste{\cG, \cF} % můžeme i definovat nová makra
  % Cílem těchto "tanečků" je docílit toho, aby byla všechna makra
  % používaná v tiskové sestavě \printaction nadefinována dříve, než
  \printaction % zavoláme.
}

% Zpracování sestavy pro tabulku zaci.csv zařídí Lua tento kód:
%\directlua{filelineaction("zaci.csv")}

\filelineaction{zaci.csv} % nebo příslušné TeXové makro

% Závěrečná ukázka použití některých "systémových" maker:

Poslední zpracovaný řádek tabulky \csvfilename:\par\smallskip

\printline\bigskip

CSV tabulka \csvfilename\ ve "zhuštěném tvaru":\par\smallskip

\printall\pagebreak

Výpis reportních informací z CSV souboru \csvfilename:\par\bigskip

\csvreport

% Ukázka toho, jak naráz zpracovat více různých CSV souborů:

% Tiskové makro pro soubor studenti.csv si připravíme jinak.

```

% Tabulka má záhlaví s názvy sloupců. Lze se tak odvolávat na názvy
% sloupcových maker, případně si vytvořit další potřebná makra

```
\def\lineaction{
\let\pohl\Pohlavi% fungování pomocných maker \priponaa a \priponai
\def\bydliste{\Ulice, \PSC\ \Mesto} % bydliště trochu jinak
\printaction % standartní tisková sestava (lze užít i jinou)
}
```

% Vlastní zpracování sestavy pro tabulku studenti.csv
\setheader % nastaví CSVHeader=true - příznak existence záhlaví
% \filelineaction{studenti.csv} % V LaTeXu zpracuje jen celý soubor
% nebo pomocí Lua i souvislé skupiny řádků
% filelineaction("studenti.csv",3,5) -- vypíše jen od ř.3 do ř.5

```
\directlua{filelineaction("studenti.csv",3,5)}
% \directlua{filelineaction("studenti.csv",8,17)} % další řádky
```

Poslední zpracovaný řádek tabulky \csvfilename:\par\smallskip

```
\printline\bigskip
```

CSV tabulka \csvfilename\ ve "zhuštěném tvaru":\par\smallskip

```
\printall\pagebreak
```

Výpis reportních informací z CSV souboru \csvfilename:\par\bigskip

```
\csvreport\pagebreak
```

% Zveřejněná ukázka výstupu programu bude provedena od tohoto místa:

% Ukázka toho, že lze také zpracovat i CSV soubor s jiným formátem:

% Příprava tiskového makra pro soubor zamestnanci.csv

% Tabulka má také záhlaví s názvy sloupců, takže postačí jen:

```
\def\lineaction{
\let\pohl\Pohlavi% Pro pomocná makra
\def\bydliste{\Bydliste} % jinak - dle CSV tabulky zamestnanci.csv
\printaction % standartní tisková sestava (lze ale použít i jinou)
}
```

% Vlastní zpracování sestavy pro tabulku zamestnanci.csv

```
% Předem nastavíme potřebné proměnné modifikující chování knihovny
\setheader % tj. soubor má hlavičku s názvy polí
\setsep{,} % oddělovač polí
\setld{"} % levý vymezořač
\setrd{"} % pravý vymezořač
\filelineaction{zamestnanci.csv}

Poslední zpracovaný řádek tabulky \csvfilename:\par\smallskip

\printline\bigskip

CSV tabulka \csvfilename\ ve "zhuřtěném tvaru":\par\smallskip

\printall\pagebreak

Výpis reportních informací z CSV souboru \csvfilename:\par\bigskip

\csvreport

\end{document}
```

5. Netriviální použití luaknihovny

Jednoduchá ukázka nemůže plně vystihnout možnosti popisované luaknihovny, proto jsem pro zájemce připravil i netriviální ukázky jejího praktického použití. Na několika příkladech si můžete systém vyzkoušet a nechat se inspirovat možnostmi. Na mém datovém úložišti jsou uloženy dokumenty, které byly vytvořeny pro účely administrativní agendy ConT_EXtmeetingu 2010 a T_EXperience 2010. Jedná se o seznamy účastníků, identifikačních kartiček, časových rozvrhů a programů konferencí (viz ukázka na obr. 3 a obr. 4). Připravena je i ukázka současného zpracování tabulek `ucitele.csv` a `zaci.csv` (s fiktivními údaji), které jsou během zpracování propojeny přes název třídy pomocí relace 1:N. Výsledkem je dokument se seznamy tříd s jejich třídními učiteli. Všechny ukázky jsou připraveny v mém oblíbeném ConT_EXtu, stejně tak jako ukázka tisku maturitních protokolů (s fiktivními údaji), které byly na našem gymnáziu několik let prakticky využívány. Aby nepřišli zkrátka ani zájemci z řad L^AT_EXistů, přiložil jsem dvě ukázky pro tisk obálek a faktur. Modifikací zdrojových souborů a jejich následnou kompilací můžete proniknout do možností luaknihovny. Jednotlivé zdroje lze kompilovat aktuálními verzemi LuaL^AT_EXu a ConT_EXtu.

Všechny dokumenty potřebné pro vyzkoušení fungování knihovny naleznete na adrese <http://cstugbull2012.hajtmar.com>.

Seznam účastníků zájezdu `zamestnanci.csv`:

Účastník č. 1:

pamí Barbora Májková
narozena 2.4.1995
bytem Mírov 96, 789 53 Mírov

Účastník č. 2:

pamí Zuzana Nová
narozena 13.1.1995
bytem U Dráhy 8, 789 01 Zábřeh

Účastník č. 3:

pamí Petr Dyk
narozen 10.4.1995
bytem Tímova 207, 789 01 Zábřeh

Účastník č. 4:

pamí Anželika Ferová
narozena 1.1.1995
bytem Revoluční 56, 787 01 Šumperk

Účastník č. 5:

pamí Jan Netěla
narozen 2.4.1995
bytem Bratrušovská 31, 787 01 Šumperk

Sestavou bylo vypsané 5 účastníků.

Poslední zpracovaný řádek tabulky `zamestnanci.csv`:

6,Netěla,Jan,m,2.4.1995,Bratrušovská 31, 787 01 Šumperk,

CSV tabulka `zamestnanci.csv` ve "zhuštěním tvaru":

- 1, Májková, Barbora, ž, 2.4.1995, Mírov 96, 789 53 Mírov,
- 2, Nová, Zuzana, ž, 13.1.1995, U Dráhy 8, 789 01 Zábřeh,
- 3, Dyk, Petr, m, 10.4.1995, Tímova 207, 789 01 Zábřeh,
- 4, Ferová, Anželika, ž, 1.1.1995, Revoluční 56, 787 01 Šumperk,
- 6, Netěla, Jan, m, 2.4.1995, Bratrušovská 31, 787 01 Šumperk,

Obř. 1 Ukázka výstupu vygenerovaného triviálním ukázkovým příkladem. Zdrojový text příkladu je na předchozích stránkách.

Výpis reportních informací z CSV souboru `zamestnanci.csv`:

Informace o používaném CSV souboru

Vstupní CSV soubor: `zamestnanci.csv`

Vymezovace a oddělovací sloupce viz. Lua proměnné `Sep`, `Ld` a `Rd`

Nastavení vymezovací a oddělovací sloupce: "pole1", "pole2", "pole3", ...

Počet sloupců v tabulce: 6

Počet řádků v tabulce: 5

Makra zpřístupňující řádková data v tabulce:

`\cA=`{PorCis}, `\cB=`{Prijmeni}, `\cC=`{Jmeno}, `\cD=`{Pohlavi}, `\cE=`{DatumNarozeni}, `\cF=`{Bydliste},

Další předdefinovaná makra:

`\csvfilename` – otevřený soubor s CSV tabulkou (`zamestnanci.csv`)

`\numcols` – počet sloupců tabulky (6)

`\numrows` – počet aktuálně zpracovaných (vypsaných) řádků (5)

`\numline` – číslo aktuálně načteného řádku (pro použití v tiskových sestavách)

`\csvreport` – výpis se report o otevřeném souboru

`\printline` – výpis aktuální řádek CSV tabulky ve zhuštěném tvaru

`\printall` – výpis CSV tabulky ve zhuštěném tvaru

`\setfiletoscan{filename}` – nastaví jméno zpracovávaného CSV souboru

`\opencsvfile{filename}` – otevření CSV souboru

`\openheadersvfile{filename}` – otevření CSV souboru s hlavičkou

`\setheader` – nastavení příznaku existence hlavičky

`\resetheader` – nastavení příznaku neexistence hlavičky

`\readrow` – načtení řádku (+ zhuštění na něm)

`\nextrow` – načtení řádku (+ skok na další)

`\setsep{separator}` – nastavení oddělovací pole na příslušný znak

`\resetsep` – nastavení oddělovací pole na defaultní hodnotu

`\setld{delimiter}` – nastavení levého vymezovacího znaku

`\resetld` – nastavení levého vymezovace na defaultní hodnotu

`\setrd{delimiter}` – nastavení pravého vymezovacího znaku

`\resetrd` – nastavení pravého vymezovace na defaultní hodnotu

`\blinenoek` – nastavení řádku (vykmaného před načtením řádku)

`\elineenoek` – nastavení řádku (vykmaného po zpracování řádku)

`\bfileenoek` – nastavení řádku (vykmaného před načtením CSV souboru)

`\efileenoek` – nastavení řádku (vykmaného po zpracování CSV souboru)

Obř. 2 Výpis reportních informací spolu se seznamem mak-
ker, které má uživatel knihovny `scanscv.lua` k dispozici.

(to date 6. 9. 2010)

Monday 13.9.

2325	Review	ConTeXt installation doc.	working	90	Notes
2025	200 Homework / Hans Hagen / Mages Magesse		120	Notes	
Tuesday 14.					
820	Reading	Colloquium on Graph		15	Notes
830	Review	Review: Document		15	Notes
915	200 Homework	Review: Lab	145	15	Notes
930	200 Homework	Reference manual update	145	15	Notes
945	200 Homework	What comes next reference	145	15	Notes
1000	200 Homework / Hans Hagen	What comes next reference	145	15	Notes
1015	200 Homework (presented by Teo)	Reference works in lua	145	15	Notes
1030	Coffee			30	Notes
1100	Hans Hagen	Why structure matters	145	30	Notes
1130	Wolfgang Schwaier	Why structure matters	145	30	Notes
1200	Luigi Scano	Why structure matters	145	30	Notes
1230	Lunch	module „moby“	145	30	Notes
1400	Meat Magesse	A Different Philosophy - Thoughts on Writing ConTeXt	discussion	60	Notes
1430	Meat Magesse	Two tables: dependencies to handling ConTeXt and writing documentation to match	discussion	60	Notes
1530	Coffee	not interfere time and layout modules (TeX, LaTeX etc.)	145	30	Notes
1600	Wolfgang Schwaier	Requirements for Documentation	145	30	Notes
1630	Hans Hagen	Req. Group: Grouping Two Sites	145	30	Notes
1700	200 Homework	Documentation discussion	discussion	60	Notes
1730	200 Homework	Documentation discussion	145	30	Notes
1800	200 Homework	Review the first ConTeXt book	145	30	Notes
1830	200 Homework	Review the first ConTeXt book	145	30	Notes
1900	Hans Hagen	Wolfgang: You Have to Know	145	30	Notes
Wednesday 15.					
820	Reading	How to Use ConTeXt in a standard office environment	145	30	Notes
830	Meat Magesse	The default module „Magesse“ comes	145	30	Notes
930	Hans Hagen / Mages Magesse	The default module „Magesse“ comes	145	30	Notes
1000	Hans Hagen	SQL: defaulting ones	145	60	Notes
1100	Coffee			30	Notes
1130	Hans Hagen	Priority class with WinEdt/Winlog	145	30	Notes
1215	200 Homework	Priority class with WinEdt/Winlog	145	30	Notes
1300	Lunch	Priority class with the „at“ module	145	30	Notes
1430	Hans Hagen	Print: diagrams using the „at“ module	145	30	Notes
1450	Mages Magesse	Print: diagrams using the „at“ module	145	30	Notes
1500	200 Homework	Print: diagrams using the „at“ module	145	30	Notes
1600	Coffee	Print: diagrams using the „at“ module		30	Notes
1630	Aditya Majeed (presented by Luigi)	Print: diagrams using the „at“ module	145	30	Notes
1700	Hans Hagen	Print: diagrams using the „at“ module	145	30	Notes
1730	Wolfgang Schwaier	The new module	145	30	Notes
1900	Review	How to Use ConTeXt in a standard office environment	145	30	Notes
2030	200 Homework	How to Use ConTeXt in a standard office environment	145	30	Notes
2100	Mages Magesse	How to Use ConTeXt in a standard office environment	145	30	Notes
2130	200 Homework	How to Use ConTeXt in a standard office environment	145	30	Notes

Obř. 4 Pro vygenerování výstupu byl použit zkonvertovaný náslleý Google document s oficiálním programem drojkonference (ConTeXt meeting a TjXpertence 2010) v Břelově.

This document was created (generated) using **scancsv**.lua library by Jurek Hajtmański.
Scancsv.lua presentation to be held on Saturday, 18. 9. 2010 on 4CH+STE in Białystok.

6. Scansv.lua a ConT_EXtový modul t-scansv.lua

Hlavní funkce luaknihovny `ParseCSVdata()` obsahuje jen jednoduchý parsovací algoritmus vycházející z formátu CSV tabulek, které používám. Algoritmus není schopen zpracovávat obecné CSV soubory, mající některé položky vymezeny vymezovačem a jiné nikoliv (když položky obsahují oddělovač polí). Stejně tak není schopen správně zpracovat CSV soubory, obsahující uvnitř vymezovačů samotné vymezovače (text pole vymezený uvozovkami obsahuje uvozovky). V ConT_EXtovém modulu `t-scansv.lua`³, který vznikl kompletním přepracováním luaknihovny na základě inspirujících připomínek a oprav Hanse Hageny, je řada rozšíření a vylepšený parsovací algoritmus.

Skalní T_EXisty jistě nepřekvapí, že zdrojový kód luaknihovny je oproti zmíněnému T_EXovému zdrojáku makra p. Olšáka značně obsáhlý (a to nejen díky mým rozsáhlým komentářům). Funkčnost obou systémů je v podstatě stejná. Zmíněný ConT_EXtový modul má navíc již několik zajímavých vylepšení mj. i možnost použití nepodmíněných i podmíněných cyklů, maker s proměnlivým počtem parametrů, možností relačního spojování tabulek (1:N) atd. Podstatným způsobem to rozšiřuje možnosti praktického použití (filtrování rozsáhlých CSV tabulek atd). Svoji původní luaknihovnu `scansv.lua` (z r. 2010) tak mohu doporučit pro jednodušší praktické použití nebo alespoň jako studijní či inspirační materiál zájemcům o jazyk Lua a jeho používání ve vlastních T_EXových dokumentech a makrech. S ConT_EXtovým modulem `t-scansv.lua` lze řešit i náročnější úlohy (viz ukázky na úložišti).

7. Závěr

Popisovaná luaknihovna i ConT_EXtový modul vznikly díky laskavé pomoci členů mailové konference `ntg-context@ntg.nl`. Speciálně chci poděkovat Hansovi Hagenovi, Wolfgangu Schusterovi a Taco Hoekwaterovi. Členům mailové konference `csTeX@cs.felk.cvut.cz` chci poděkovat za rady a pomoc týkající se obecně T_EXu. Speciální poděkování patří Zdeňku Wagnerovi a Petru Olšákovi. Pavlu Střížovi děkuji za testování knihovny, cenné rady a za to, že mne inspiroval luaknihovnu dopracovat do fáze, kdy ji snad může využít i někdo jiný.

Popisovaná luaknihovna by měla být považována za moji „luaprvtinu“. Nedokonalosti a neoptimálnosti kódu nechtě jsou tomu přičítány. Lua jazykem jsem se začal zabývat především proto, aby se mohl oprostít zejména od častého používání jazyka Perl spolu s ConT_EXtem (MKII). Pro možnosti jazyka Lua

³V červnu 2011 jsem kompletně původní Lua knihovnu přepracoval a vytvořil regulérní lua-modul pro ConT_EXt MKIV. Tento modul (`t-scansv.lua`) obsahuje řadu zajímavých vylepšení, nové funkce a je optimalizován pro ConT_EXt, v němž byl také již mnohokrát v ostrém provozu otestován.

a jeho integraci do T_EXu jsem se nadchnul a všem T_EXistům doporučuji Lua jazyk minimálně vzít na milost :-). Doufám, že se mezi čtenáři najdou i ti, kdo se mým příspěvkem nechají inspirovat a rozšíří řady příznivců jazyka Lua a LuaT_EXu, LuaL^AT_EXu či ConT_EXtu.

Summary: ScanCSV – Lua Library for CSV file ConT_EXt and LuaL^AT_EX processing

Data stored in CSV (Comma Separated Values) files are often used in data processing. This article describes the author's `scancsv.lua` library, its origin and demonstrates practical examples of its usage in ConT_EXt MKIV and LuaL^AT_EX. Author shows how easily and quickly create print reports, letters, forms, certificates, invitations, cards, business cards, double-sided cards, tables, animations etc. using external CSV text databases.

Users of ConT_EXt MKIV (but LuaL^AT_EX and LuaT_EX as well) can easily use data from external CSV tables in their own documents via the library, using the T_EX macros built on the library and make this data available in an attractive and very simple and natural way.

*Jaroslav Hajtmar, hajtmar@gyza.cz
Gymnázium Zábřeh
Nám. Osvobození 20
789 01 Zábřeh*