# The SCANCSV.LUA library

[1]

The more input the more fun ...

Jaroslav Hajtmar

ConTEXtmeeting 2010

18 September 2010

[1] http://jaroslav.hajtmar.com/storage/tex/ctm+te2010/presentation

# Apology English – speaking participants

Sorry, but this talk is Czech - English. Any parts of slideshow I will comment in Czech, others in English.
Due to my language skills I would probably did not know perfectly explain concrete things. The main things I try to say in English (for edification and also for laughing - thanks to my English of course). The guide slideshow (the only documentation) is only in English.
Sorry for my errors - I am eternal novice in English and in Lua too.
Thanks for your understanding.

This slideshow with examples you can find on address:

http://jaroslav.hajtmar.com/storage/tex/ctm+te2010/presentation

You can read it later at more convenient slideshow speed.

# Abstract

Data stored in CSV (Comma Separated Values) files are often used in data processing. This presentation describes the author's ScanCSV.lua library, its origin and demonstrates practical examples of its usage in ConTeXt MKIV. 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 ConTeXt MKIV (but LuaLATEX and LuaTEX as well) can easily use data from external CSV tables in their own documents via the library, using the TeX macros built on the library and make this data available in an attractive and very simple and natural way.

# Introduction

- SCANCSV.LUA library – easy way to use text database data stored in external CSV files in ConTeXt MkIV (in LuaLaTeX and LuaPlain is working too).
- Easily create LuaTeX documents which can handle multiple data sets (CSV simple database).
- Number of use cases: printing of various forms, collective letters, certificates, invitations, cards, business cards, double-sided cards, tables, animations etc.
- Main objective : easy to use without knowledge of Lua, use also in LuaLaTeX and LuaPlainTeX, access CSV data purely by TeX macros built on library functions (without Lua code), motivate other users to use LuaTeX.
- Very useful and very easy to using . . .

# CSV data format and SCANCSV.LUA

- Exchange data, export to CSV (e.g. the MySQL database), a simpler alternative to the XML, easy handling (sorting and editing), spreadsheets (Excel, Calc, Gnumeric, . . . )
- General description of CSV format.
- CSV format suitable for SCANCSV.LUA: file must be encoded in UTF-8! (Exported XLS files to be recoded – disadvantage).
- Fields separator (delimiter): basically anything, default value is ; semi–colon (MS Excel).
- Spacers fields (quoters?) (I dont know right designation): can be anything, left and right may be different (most often " - quotes), the default value is without spacers!
- The parsing algorithm in SCANCSV.LUA is trivial (although it can be freely adjusted) => limitation (if spacers are set, then must be used everywhere – it is not required generally).

# SCANCSV – history, inspiration

- 2005 – discovery of scanbase.tex macro of Petr Olšák. Macro process text files in a particular format.
- Petr Olšák modified and generalized the macro scanbase.tex to new macro scancsv.tex – it process text files in CSV format. I used it in plainTeX till 2008.
- 2008 - modification of macro for LaTeX (Jaromír Kuben) and for ConTeXt (Petr Olšák). I use it in ConTeXt MkII up to now.
- 2010 - I started to use ConTeXt MkIV. Original macro does not work there. ConTeXt is working with character set UTF8, but macro is unable to process this character set.
- March 2010 – my familiarization with LuaTeX and Lua

language, I started with creating the library scancsv.lua. First version was practically useless. . .

- July 2010 – first really usable version.
- today – daily usage, improvements, tuning and expansion of options.

# The operating principle of the library

1. Load library scancsv.lua (the only Lua code in the source text).
2. Optional settings of header flag, separator elements, and spacers (otherwise, the default value is used).
3. Open CSV file (different ways).
4. Load CSV table row (manually or in a cycle).
5. Parse row (column separation data).
6. Retrieve column data into TeX macros.
7. Repeat steps 4 to 6 for all lines of CSV tables.

Processing method of first table row depends on whether it's header or not. After loading the column data in the macro (ie when you open the CSV file) data are available ConTeXt (LuaTeX or LuaLaTeX). Rows can be browsed "manually“ or using the standard cycles or macros of library.

# Using in the "manual" mode

- Load library: `\directlua{dofile("scancsv.lua")}`.
- Setting a header flag (when the header is present): `\setheader` (or unset - `\resetheader`).
- Open CSV file `\opencsvfile{file.csv}`.
- Then, in source text, we can use the macros `\cA`, `\cB` ... (or `\Firstname`, `\Lastname`, ... if first line contains header information ie values Firstname, Lastname, ...). These macros contains the column values of the current CSV row.
- `\nextrow` - go to the next table row (macro `\cA`, `\cB` ... or `\Firstname`, `\Lastname`, ... are filled with new values).

# Main TeX macros for using the library

- `\setfiletoscan{CSVFile}` – setting of name of CSV file
- `\setheader` – set a header flag
- `\resetheader` – unset a header flag
- `\setsep{,}`, `\setld{*}`, `\setrd{!}` – set separator of columns, spacers of columns to user value – left and right (nondafault value)
- `\resetsep`, `\resetld`, `\resetrd` – unset to default values
- `\opencsvfile{CSVFile}` - open CSV table, `\openheadercsvfile{CSVFile}` - open CSV table (and set header flag)
- `\nextrow` – goto to next row of CSV file
- `\printline`, `\printall` – print all of line / all of CSV table
- `\filelineaction` `\filelineaction{CSVfile}`

`\filelineaction{CSVfile}{to}`
`\filelineaction{CSVfile}{from}{to}`
– macros for processing of user–defined macro `\lineaction` (name is required) in a cycle. For LaTeX users is only applicable `\filelineaction{CSVfile}` option (one parameter).
`ScanCSV.lua` library distinguish different environments (eg ConTeXt or LuaLaTeX), and hence to define any the appropriate macros (because of macro compatibility).

# Simplest example 1

```
% Processing file - example1.csv:
% 1;Petr;Novák;19.5.1989;m;Nymburk;U Brány 7
% 2;Jan;Novotný;5.7.1991;m;Praha;Uhlířská 178
% ...

\directlua{dofile("scancsv.lua")} % Load library

% pattern for cycle processing
\def\lineaction{\item \cB\ {\bf \cC}\  (\cD)}

\starttext
List of participants:
 \startitemize[n]
  \filelineaction{example1.csv} % Cycle generating
 \stopitemize
\stoptext
```

# Result of example 1

List of participants:

1. Petr **Novák** (19.5.1989)
2. Jan **Novotný** (5.7.1991)
3. Zuzana **Vašíčková** (13.9.1984)
4. Pavel **Brožkan** (14.8.1992)
5. Lenka **Rábelová** (8.11.1992)

# Simplest example 2

```
%
% example2.csv:
% Surname,Firstname,Birthdate,Sex,City,Zipcode,Street
% Novák,Jan,14.10.1997,m,Zbečno,27024,Farní 21
% Pospíšilová,Hana,4.1.1996,ž,Zábřeh,78901,Studénky 420
% ...


\directlua{dofile("scancsv.lua")}

% pattern for cycle processing
\def\lineaction{\item \Birthdate: {\bf \Surname}\ \Firstname}

\starttext
\setheader % set header flag
\setsep{,} % set separator (delimiter)

List of participants:
```

```
\startitemize[n]
 %\filelineaction{example2.csv} % Cycle generating for all rows
 \filelineaction{example2.csv}{3} % for first 3 rows
 % \filelineaction{example2.csv}{3}{5} % from 3 to 5 row
\stopitemize

\stoptext
```

## Result of example 2

List of participants:

1. 14.10.1997: **Novák** Jan
2. 4.1.1996: **Pospíšilová** Hana
3. 15.8.1997: **Žáková** Eliška

# Report information produced by `\csvreport` macro

**Current CSV file report**
Input CSV file: : **example2.csv**
Separator (delimiter) and "quoters" see Lua variables `Sep`, `Ld` a `Rd`
Current settings of delimiters and quoters: `pole1,pole2,pole3, ...`
Number of columns in a table: **7**
Number of rows in the table: **5**
Macros supplying columns data in each row of table:
**\cA=\Surname**, **\cB=\Firstname**, **\cC=\Birthdate**, **\cD=\Sex**, **\cE=\City**, **\cF=\Zipcode**, **\cG=\Street**,
Additional predefined macros:
**\csvfilename** – name of open CSV file (**example2.csv**)
**\numcols** – number of table columns (**7**)
**\numrows** – number of currently processed lines ()
**\numline** – number of the currently loaded row (for use in print reports)
**\csvreport** – prints the report on file open
**\printline** – lists the current CSV row table in a condensed form
**\printall** – CSV output table in a condensed form
**\setfiletoscan**{*filename*} – setting of name of CSV file
**\opencsvfile**{*filename*} – open CSV table
**\openheadercsvfile**{*filename*} – open CSV table (and set header flag)
**\setheader** – set a header flag
**\resetheader** – unset a header flag
**\readrow** – next row of CSV table
**\nextrow** – next row of CSV table (with test of EOF)
**\setsep**{*separator*} – set separator of columns
**\resetsep** – unset to default values
**\setld**{*delimiter*} – set left quoter
**\resetld** – unset left quoter to default values
**\setrd**{*delimiter*} – set right quoter
**\resetrd** – unset right quoter to default values
**\blinehook** – begin line hook macro (process before first column value of each row)
**\elinehook** – end line hook macro (process after last column value of each row)
**\bfilehook** – begin file hook macro (process before whole file processing)
**\efilehook** – end file hook macro (process after whole file processing)

# Simplest example 3 – table generator

```
%
% example2.csv:
% Surname,Firstname,Birthdate,Sex,City,Zipcode,Street
% Novák,Jan,14.10.1997,m,Zbečno,27024,Farní 21
% Pospíšilová,Hana,4.1.1996,ž,Zábřeh,78901,Studénky 420
% ...


\directlua{dofile("scancsv.lua")}

% pattern for cycle processing
\def\tableaction{\expanded{
 \bTR
  \bTD \numline. \eTD
  \bTD \Firstname\ \Surname \eTD
  \bTD \Street, \Zipcode\ \City \eTD
 \eTR}
}
```

```
\starttext
\setheader % set header flag
\setsep{,} % set separator (delimiter)
\opencsvfile{example2.csv} % open CSV file

List of participants:
\blank[big]

% Example of use standard ConTeXt cycles
\bTABLE
 %\dorecurse{4}{\tableaction\nextrow} % for first 4 rows
 \doloop{\ifEOF\exitloop\else\tableaction\nextrow\fi} % All rows
\eTABLE

\stoptext
```

# Result of example 3

List of participants:

| | | |
|---|---|---|
| 1. | Jan Novák | Farní 21, 27024 Zbečno |
| 2. | Hana Pospíšilová | Studénky 420, 78901 Zábřeh |
| 3. | Eliška Žáková | Radniční 38, 78901 Zábřeh |
| 4. | Jan Novotný | Uhlířská 178, 11150 Praha |
| 5. | Přemysl Oráč | Rybniční 25, 75421 Mošnov |

# Simplest example 4 – using hooks for table typing

```
%
% example2.csv:
% Surname,Firstname,Birthdate,Sex,City,Zipcode,Street
% Novák,Jan,14.10.1997,m,Zbečno,27024,Farní 21
% Pospíšilová,Hana,4.1.1996,ž,Zábřeh,78901,Studénky 420
% ...

\directlua{dofile("scancsv.lua")}

% hooks defining:
\def\bfilehook{\bTABLE}
\def\efilehook{\eTABLE}
\def\blinehook{\bTR}
\def\elinehook{\eTR}
```

```
% pattern for cycle processing
\def\lineaction{\expanded{
 \bTD \numline. \eTD
 \bTD \Firstname\ \Surname \eTD
 \bTD \Street, \Zipcode\ \City \eTD}
}

\starttext
\setheader % set header flag
\setsep{,} % set separator (delimiter)

List of participants:\blank[big]

% only from 3rd to 5th row of example2.csv file
\filelineaction{example2.csv}{3}{5}

\stoptext
```

# Result of example 4

List of participants:

| | | |
|---|---|---|
| 1. | Eliška Žáková | Radniční 38, 78901 Zábřeh |
| 2. | Jan Novotný | Uhlířská 178, 11150 Praha |
| 3. | Přemysl Oráč | Rybniční 25, 75421 Mošnov |

# Simplest example 5 – use in LuaLaTeX

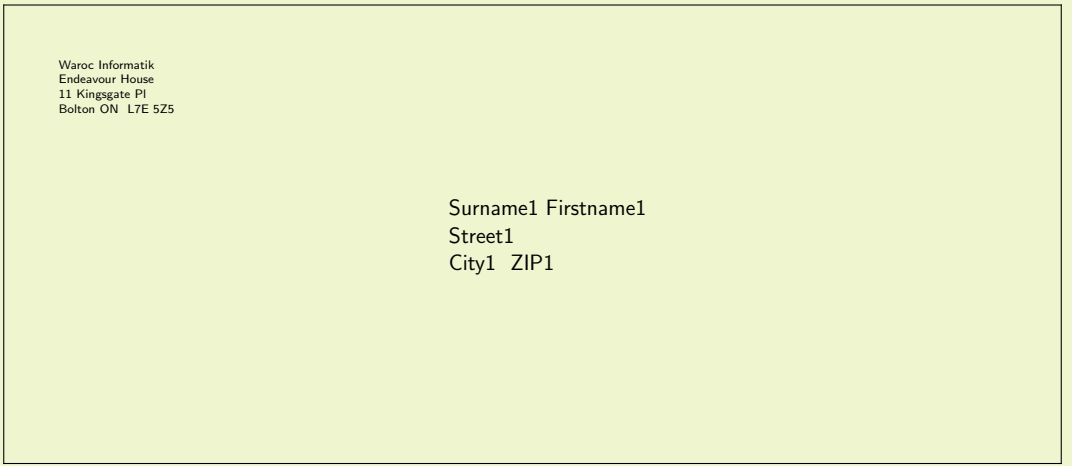
```
% database.csv:
% Id;Surname;Firstname;City;ZIP;Street;Country
% 1;Surname1;Firstname1;City1;ZIP1;Street1;Country1
% 2;Surname2;Firstname2;City2;ZIP2;Street2;Country2
%
% Compiling: lualatex scancsv-envelope.tex

\documentclass[business]{envelope}
\usepackage[utf8]{luainputenc}
\usepackage{luatextra}

\def\lineaction{
       \Addressee{
       \Surname\ \Firstname \\
       \Street \\
       \City\ \ \ZIP \\
  }
       \makeEnvelope\newpage
}
```

```
\begin{document}
 \directlua{dofile("scancsv.lua")}
 \setheader
 \filelineaction{database.csv}
\end{document}
```

## Result of example 5



# SCANCSV.LUA process "Crazy CSV tables" too!

## Example of crazy CSV table:

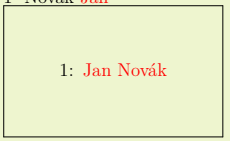
```
Id;Align;Color;LineMacro;lineaction;Prijmeni;Jmeno;DatumNarozeni;Pohlavi;Mesto;PSC;Ulice
1;left;\red;{\framed[width=5cm,height=3cm]{\numline: {\Color\Jmeno\ \Prijmeni}} };{\numline -
\Prijmeni\ {\Color\Jmeno}\par\LineMacro};Novák;Jan;14.10.1997;m;Zbečno;27024;Farní 21
2;right;\blue;{\framed[width=5cm,height=3cm]{\numline: {\Color\Prijmeni\ \Jmeno}} };{\numline
- \Jmeno\ {\Color\Prijmeni}\par\LineMacro};Pospíšilová;Hana;4.1.1996;ž;Zábřeh;78901;Studénky
420
3;middle;\orange;{\framed[width=3cm,height=5cm]{\numline: {\Color\Jmeno\ \Prijmeni}} };{\numline
- Jméno = {\Color\Jmeno}\par\LineMacro};Žáková;Eliška;15.8.1997;ž;Zábřeh;78901;Radniční 38
4;right;\magenta;{\framed[width=2cm,height=4cm]{\numline: {\Color\Jmeno\ \Prijmeni}} };{\numline
- Příjmení= {\Color\Prijmeni\LineMacro}\par};Novotný;Jan;5.7.1991;m;Praha;11150;Uhlířská 178
5;left;\green;{\framed[width=6cm,height=3cm]{\numline: {\Color\Jmeno\ \Prijmeni}} };{\numline
- XXL \LineMacro};Oráč;Přemysl;11.5.1989;m;Mošnov;75421;Rybniční 25
...
```

## or CSV data using in animation example:

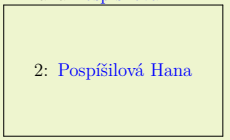
```
Number;Pointcolor;Linecolor;Text;Radius
1;\red;\blue;Zkušební text \the\numexpr\Number+\numcols\relax;30
2;\blue;\green;Lorem ipsum \the\numexpr\Number+\numline+55\relax;25
3;\cyan;\blue;Lorem \the\numexpr\Number-\numline+107\relax;20
4;\yellow;\green;Lorem sit \the\numexpr\Number+26+\numline\relax;18
5;\green;\blue;Dim = \the\dimexpr(\vsize+\numline\hsize-2730pt);16
...
```

# Example of result of crazy data

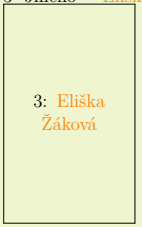
1- Novák Jan




2- Hana Pospíšilová

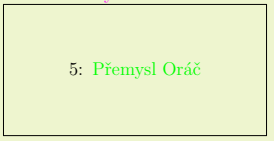



3- Jméno = Eliška





4- Příjmení= Novotný




5- XXL

# TeX macros for accessing of columns data

**CSV file without Header ie default option** (**\resetheader** is not necessary**)**

\cA \cB \cC \cD …  \resetheader

1;Petr;Novák;19.5.1989;m;Nymburk;U Brány 7
2;Jan;Novotný;5.7.1991;m;Praha;Uhlířská 178
3;Zuzana;Vašíčková;13.9.1984;ž;Ostrava;Jánská 14
...

no header (default)

data lines

**CSV file with Header (switch with \setheader)** (both types of macros are defined)

\cA = **\Surname** \cB = **\Firstname** \cC = **\Birthdate** … \setheader

**Surname;Firstname;Birthdate;Sex;City;Zipcode;Street** ← Header (no data)
Novák,Jan,14.10.1997,m,Zbečno,27024,Farní 21
Pospíšilová,Hana,4.1.1996,ž,Zábřeh,78901,Studénky 420
…

data lines

There are possibility set of Roman numbers of columns:

\cI, \cII, \cIII, \cIV, ... (default UserColumnNumbering='XLS')

# TeX macros to obtain „system“ information

- `\csvfilename` – name of actual open CSV file
- `\numcols` – number of columns of the CSV table
- `\numrows` – number of processed (offered) lines
- `\numline` – the serial number of the currently loaded row
- `\csvreport` – Report information on open CSV file

# Hooks for data processing

- `\blinehook`, `\elinehook` – begin line hook, end line hook - this macros are executed before and after processing row macro `\lineaction` (ie CSV table row).
- `\bfilehook`, `\efilehook` – performed before and after processing the entire CSV table.
- `\bch`, `\ech` – begin column hook, end c.h. - can be manually set in lua code, because of the impossibility of testing the macro, this option is disabled.

Default values of all hooks are `\relax`.

# TeX IF for testing EOF CSV file

- `\ifEOF` – TRUE, if we get to the end of processing a CSV file.
- `\ifnotEOF` – opposite `\ifEOF`.

# Modification of functions of library

- Default settings can be changed by editing the file `scancsv.lua` - in the introductory section of code.
- During the processing of ConTeXt MKIV (LuaLaTeX) it is possible to continuously change settings separator (delimiter), spacers, headers, using TeX macros . . . .
- Possibility of processing many different CSV files in one document (with different separators and spacers columns).
- Use hooks – default are `\relax`.

# Main Lua library functions

- `ParseCSVdata()` – function for parsing of individual records (rows) of CSV table.
- `lineaction()` – processing of user macro `\lineaction` according to the specified range of lines at the open CSV file.
- `CreatePageFiles()` – create two CSV files from one open CSV file. It will by used to print double-sided cards, printed on the page in block R x C (it reorder the CSV file with the 2nd page so that the front and back of the tiles match).
- `Filelineactioncards()` – printing 1st and 2nd sides of list of cards from the files created by the previous function.
- `CSVReport()` – get report information about the open CSV file.
- `csvfilename()` – name of actually open CSV file
- `TMN(s)` – (TeX Macro Name). Macro name must not contain

prohibited characters.

- `ar2rom()` – converts Arabic numbers to Roman. Used for "numbering" the column in the macro.
- `ar2xls()` – converts numbers to the column name (Excel format).
- `ar2colnum()` – converts TeX macro column name based on the global variable (Excel, Roman).
- `printline()` – prints actual row of the CSV table.
- `printall()` – prints the whole CSV table.
- `printallcontext()` – prints the whole CSV table in ConTeX syntax.

# Testing and cycles

**Conditions with AND and OR (see Olšák TBN).**

```
% Condition A AND B
\doloop{
\ifnum\Id>2
       \ifnum\Id<10\lineaction
       \fi
\fi
\ifEOF\exitloop\else\nextrow\ifEOF\exitloop\fi\fi
}
----------------------------------------
% Condition A OR B
\def\AorB{\lineaction}
\doloop{
\ifnum\Id=1\AorB%
\else\ifnum\Id>3\AorB\fi
\fi
\ifEOF\exitloop\else\nextrow\ifEOF\exitloop\fi\fi
}
```

# SCANCSV.LUA and cycles

Examples of ConTeXt cycles:

- `\dorecurse{5}{\lineaction\nextrow}` - `\lineaction` macro for next 5 rows.
- `\doloop{\lineaction\nextrow\ifnum\numline>7\exitloop\fi}` - clear.
- `\doloop{\ifEOF\exitloop\else\lineaction\nextrow\fi}` - clear.
- `\doloop{\lineaction\nextrow \if\Id3 \exitloop \fi}` - clear.

**Examples of library cycles (only in test version SCANCSV.LUA):**
The macros are based on `\doloop` macro to easier use in source code.

- `\doloopwhile{\Class}{3.A}{\tableaction}` - List all meeting the criterion
- `\doloopuntil{\Class}{3.A}{\tableaction}` - list until it is not satisfied
- `\doloopforall{\lineaction}` – for all lines will `\lineaction` macro
- `\doloopfromto{3}{7}{\lineaction}`

- `\doloopaction` – without parameter done for all rows macro `\lineaction`.
- `\doloopaction{\useraction}` – done for all rows user macro `\useraction`.
- `\doloopaction{\useraction}{5}` – for the first 5 rows will doing `\useraction` macro.
- `\doloopaction{\useraction}{5}{7}` - for rows 5-7 will doing `\useraction` macro.

# Constraints, compatibility, flaws

- SCANCSV.LUA does not process general CSV files. Reason: The parsing algorithm is very simple. If the item contains a column separator (delimiter) "," the CSV output is the following:
  `1, Jan, Novotny, "The Gate 4, 111 50 Prague", ...`

  Solution: an better (general) algorithm. Only requires to change `ParseCSVdata()` function.
- Occasional problems with the expansion. E.g. I failed to get SCANCSV.LUA running in the module database (`\usemodule [database]`) by Mojca Miklavec.
- Some things work only in ConTeXt.
- Mind that the CSV files may contain characters %, &, _, ... etc. Change catcodes is needed first.

# Possibilities of improvement...

- Cycles implementation.
- Purification and English code comment.
- Improvements and generalizations of parsing algorithm (current algorithm is fully sufficient for (my) normal use!!!).
- Create a separate module ONLY FOR MKIV (ie removing number of limitations in LuaLaTeX)???
- Implementation of any improvements from users.

# Thanks...


- To members of mail conference `ntg-context@ntg.nl` for advices about ConTeXt and Lua. The library would not have been created without their kind assistance. Special thanks to Taco Hoekwater, Hans Hagen, Wolfgang Schuster.
- To members of mail conference `cstex@cs.felk.cvut.cz` for advices about TeX and LaTeX. Especially to Mr. Zdenek Wagner, Vit Zýka, Pavel Stříž, Petr Olšák, ...
- To Pavel Stříž for inspiration, testing, advices and for convincing me to finish the library and present it at this conference.

# Links to examples

http://jaroslav.hajtmar.com/storage/tex/ctm+te2010/presentation

And here is direct link to file browser