<table>
<thead>
<tr>
<th>A01</th>
<th>B01</th>
<th>C01</th>
<th>D01</th>
<th>E01</th>
<th>F01</th>
<th>G01</th>
<th>H01</th>
<th>I01</th>
<th>J01</th>
<th>K01</th>
<th>L01</th>
</tr>
</thead>
<tbody>
<tr>
<td>A02</td>
<td>B02</td>
<td>C02</td>
<td>D02</td>
<td>E02</td>
<td>F02</td>
<td>G02</td>
<td>H02</td>
<td>I02</td>
<td>J02</td>
<td>K02</td>
<td>L02</td>
</tr>
<tr>
<td>A03</td>
<td>B03</td>
<td>C03</td>
<td>D03</td>
<td>E03</td>
<td>F03</td>
<td>G03</td>
<td>H03</td>
<td>I03</td>
<td>J03</td>
<td>K03</td>
<td>L03</td>
</tr>
<tr>
<td>A04</td>
<td>B04</td>
<td>C04</td>
<td>D04</td>
<td>E04</td>
<td>F04</td>
<td>G04</td>
<td>H04</td>
<td>I04</td>
<td>J04</td>
<td>K04</td>
<td>L04</td>
</tr>
<tr>
<td>A05</td>
<td>B05</td>
<td>C05</td>
<td>D05</td>
<td>E05</td>
<td>F05</td>
<td>G05</td>
<td>H05</td>
<td>I05</td>
<td>J05</td>
<td>K05</td>
<td>L05</td>
</tr>
<tr>
<td>A06</td>
<td>B06</td>
<td>C06</td>
<td>D06</td>
<td>E06</td>
<td>F06</td>
<td>G06</td>
<td>H06</td>
<td>I06</td>
<td>J06</td>
<td>K06</td>
<td>L06</td>
</tr>
<tr>
<td>A07</td>
<td>B07</td>
<td>C07</td>
<td>D07</td>
<td>E07</td>
<td>F07</td>
<td>G07</td>
<td>H07</td>
<td>I07</td>
<td>J07</td>
<td>K07</td>
<td>L07</td>
</tr>
<tr>
<td>A08</td>
<td>B08</td>
<td>C08</td>
<td>D08</td>
<td>E08</td>
<td>F08</td>
<td>G08</td>
<td>H08</td>
<td>I08</td>
<td>J08</td>
<td>K08</td>
<td>L08</td>
</tr>
<tr>
<td>A09</td>
<td>B09</td>
<td>C09</td>
<td>D09</td>
<td>E09</td>
<td>F09</td>
<td>G09</td>
<td>H09</td>
<td>I09</td>
<td>J09</td>
<td>K09</td>
<td>L09</td>
</tr>
<tr>
<td>A10</td>
<td>B10</td>
<td>C10</td>
<td>D10</td>
<td>E10</td>
<td>F10</td>
<td>G10</td>
<td>H10</td>
<td>I10</td>
<td>J10</td>
<td>K10</td>
<td>L10</td>
</tr>
<tr>
<td>A12</td>
<td>B12</td>
<td>C12</td>
<td>D12</td>
<td>E12</td>
<td>F12</td>
<td>G12</td>
<td>H12</td>
<td>I12</td>
<td>J12</td>
<td>K12</td>
<td>L12</td>
</tr>
<tr>
<td>A14</td>
<td>B14</td>
<td>C14</td>
<td>D14</td>
<td>E14</td>
<td>F14</td>
<td>G14</td>
<td>H14</td>
<td>I14</td>
<td>J14</td>
<td>K14</td>
<td>L14</td>
</tr>
<tr>
<td>A15</td>
<td>B15</td>
<td>C15</td>
<td>D15</td>
<td>E15</td>
<td>F15</td>
<td>G15</td>
<td>H15</td>
<td>I15</td>
<td>J15</td>
<td>K15</td>
<td>L15</td>
</tr>
<tr>
<td>A16</td>
<td>B16</td>
<td>C16</td>
<td>D16</td>
<td>E16</td>
<td>F16</td>
<td>G16</td>
<td>H16</td>
<td>I16</td>
<td>J16</td>
<td>K16</td>
<td>L16</td>
</tr>
</tbody>
</table>
1 Introduction

Occasionally a question pops up on the ConTExt mailing list and answering it becomes a nice distraction from a boring task at hand. The spreadsheet module is the result of such a diversion. As with more support code in ConTExt, this is not a replacement for ‘the real thing’ but just a nice feature for simple cases. The module is loaded with

\usemodule[spreadsheet]

So this is (at least currently) not one of the core functionalities but an add-on. Of course some useful extensions might appear in the future.

2 Spreadsheet tables

We can use Lua in each cell, because under the hood it is all Lua. There is some basic parsing applied so that we can use the usual A..Z variables to access cells.

\startspreadsheettable[test]
  \startrow
    \startcell 1.1 \stopcell
    \startcell 2.1 \stopcell
  \stoprow
  \startrow
    \startcell 2.1 \stopcell
    \startcell 2.2 \stopcell
  \stoprow
  \startrow
  \stoprow
\stopspreadsheettable

The rendering is shown in figure 1. Keep in mind that in Lua all calculations are done using floats, at least in Lua versions with version numbers preceding 5.3.
The last cell can also look like this:

```
\startcell
function()
  local s = 0
  for i=1,2 do
    for j=1,2 do
      s = s + dat[i][j]
    end
  end
  return s
end
\stopcell
```

The content of a cell is either a number or a function. In this example we just loop over the (already set) cells and calculate their sum. The `dat` variable accesses the grid of cells.

```
\startcell
function()
  local s = 0
  for i=1,2 do
    for j=1,2 do
      s = s + dat[i][j]
    end
  end
  tmp.total = s
end
\stopcell
```

In this variant we store the sum in the table `tmp` which is local to the current sheet. Another table is `fnc` where we can store functions. This table is shared between all sheets. There are two predefined functions:

```
sum(columnname,firstrow,lastrow)
fmt(specification,n)
```

The `sum` function works top-down in columns, and roughly looks like this:

```
function sum(currentcolumn,firstrow,lastrow)
  local r = 0
  for i = firstrow, lastrow do
    r = r + cells[currentcolumn][i]
  end
  return r
end
```
The last two arguments are optional:

\begin{verbatim}
sum(columnname, lastrow)
\end{verbatim}

This is equivalent to:

\begin{verbatim}
function sum(currentcolumn, lastrow)
    local r = 0
    for i = 1, lastrow do
        r = r + cells[currentcolumn][i]
    end
    return r
end
\end{verbatim}

While:

\begin{verbatim}
sum(columnname)
\end{verbatim}

boils down to:

\begin{verbatim}
function sum(currentcolumn)
    local r = 0
    for i = 1, currentrow do
        r = r + cells[currentcolumn][i]
    end
    return r
end
\end{verbatim}

Empty cells or cells that have no numbers are skipped. Let's now see these functions in action:

\begin{verbatim}
\startspreadsheettable[test]
\startrow
    \startcell 1.1 \stopcell \startcell 2.1 \stopcell
\stoprow
\startrow
    \startcell 2.1 \stopcell \startcell 2.2 \stopcell
\stoprow
\startrow
    \startcell function()\end{verbatim}

\begin{verbatim}
    local s = 0
    for i=1,2 do
        for j=1,2 do
            s = s + dat[i][j]
        end
    end
    context.bold(s)
end
\stopcell
\startcell function()\end{verbatim}

\begin{verbatim}
    local s = 1
\stopcell
\end{verbatim}
for i=1,2 do
  for j=1,2 do
    s = s * dat[i][j]
  end
end
context.bold(fmt("@.1f",s))
end
\stopcell
\stoprow
\stopspreadsheettable

The result is shown in figure 2. Watch the fmt call: we use an at sign instead of a percent to please \TeX.

\begin{figure}[h]
\centering
\begin{tabular}{|c|c|}
\hline
1.1 & 2.1 \\
2.1 & 2.2 \\
7.5 & 10.7 \\
\hline
\end{tabular}
\caption{Cells can be (complex) functions.}
\end{figure}

Keep in mind that we're typesetting and that doing complex calculations is not our main objective. A typical application of this module is in making bills, for which you can combine it with the correspondence modules. We leave that as an exercise for the reader and stick to a simple example.

\begin{verbatim}
\startspreadsheettable[test]
\startrow
  \startcell[align=flushleft,width=8cm] "item one" \stopcell
  \startcell[align=flushright,width=3cm] @ "0.2f EUR" 3.50 \stopcell
\stoprow
\startrow
  \startcell[align=flushleft] "item two" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" 8.45 \stopcell
\stoprow
\startrow
  \startcell[align=flushleft] "tax 19\%" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" 0.19 * (B[1]+B[2]) \stopcell
\stoprow
\startrow
  \startcell[align=flushleft] "total 1" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" sum(B,1,3) \stopcell
\stoprow
\startrow
  \startcell[align=flushleft] "total 2" \stopcell
\stoprow
\startrow
  \startcell[align=flushleft] "total 3" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" sum(B) \stopcell
\stoprow
\stopspreadsheettable
\end{verbatim}
Here (and in figure 3) you see a quick and more readable way to format cell content. The @ in the template is optional, but needed in cases like this:

@ "(@0.2f) EUR" 8.45

A @ is only prepended when no @ is given in the template.

<table>
<thead>
<tr>
<th>item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>item one</td>
<td>3.50 EUR</td>
</tr>
<tr>
<td>item two</td>
<td>8.45 EUR</td>
</tr>
<tr>
<td>tax 19%</td>
<td>2.27 EUR</td>
</tr>
<tr>
<td>total 1</td>
<td>14.22 EUR</td>
</tr>
<tr>
<td>total 2</td>
<td>14.22 EUR</td>
</tr>
<tr>
<td>total 3</td>
<td>42.66 EUR</td>
</tr>
</tbody>
</table>

**Figure 3** Cells can be formatted by using @ directives.

In practice this table we can be less specific and let \texttt{\sum} behave more automatical. That way the coding can be simplified (see figure 4) and also look nicer.

\texttt{\startspreadsheettable[test][frame=off]}
  \startrow
  \startcell[align=flushleft,width=8cm] "The first item" \stopcell
  \startcell[align=flushright,width=3cm] @ "0.2f EUR" 3.50 \stopcell
  \stoprow
  \startrow
  \startcell[align=flushleft] "The second item" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" 8.45 \stopcell
  \stoprow
  \startrow
  \startcell[align=flushleft] "The third item" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" 5.90 \stopcell
  \stoprow
  \startrow[topframe=on]
  \startcell[align=flushleft] "VAT 19\ percent" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" 0.19 * sum(B) \stopcell
  \stoprow
  \startrow[topframe=on]
  \startcell[align=flushleft] "\bf Grand total" \stopcell
  \startcell[align=flushright] @ "0.2f EUR" sum(B) \stopcell
  \stoprow
\stopspreadsheettable

| The first item | 3.50 EUR |
| The second item| 8.45 EUR |
| The third item | 5.90 EUR |
| VAT 19%        | 3.39 EUR |
| **Grand total**| 21.24 EUR|

**Figure 4** The \texttt{\sum} function accumulates stepwise.
There are a few more special start characters. This is demonstrated in figure 5. An \texttt{=} character is ignored.\footnote{Taco suggested to support this because some spreadsheet programs use that character to flush a value.} When we start with an \texttt{!}, the content is not typeset. Strings can be surrounded by single or double quotes and are not really processed.

\begin{verbatim}
\startspreadsheettable[test][offset=1ex]
 \startrow
 \startcell[align=flushleft] "first" \stopcell
 \startcell[align=flushleft] \\texttt{@ "[@i]" 1}' \stopcell
 \startcell[align=flushright,width=3cm] @ "[@i]" 1 \stopcell
 \stoprow
 \startrow
 \startcell[align=flushleft] "second" \stopcell
 \startcell[align=flushleft] \\texttt{= 2}' \stopcell
 \startcell[align=flushright] = 2 \stopcell
 \stoprow
 \startrow
 \startcell[align=flushleft] "third" \stopcell
 \startcell[align=flushleft] \\texttt{! 3}' \stopcell
 \startcell[align=flushright] ! 3 \stopcell
 \stoprow
 \startrow
 \startcell[align=flushleft] "fourth" \stopcell
 \startcell[align=flushleft] \\texttt{4}' \stopcell
 \startcell[align=flushright] 4 \stopcell
 \stoprow
 \startrow
 \startcell[align=flushleft] \textbf{total one} \stopcell
 \startcell[align=flushleft] \texttt{\texttt{\texttt{\texttt{\texttt{sum(C)}}}}' \stopcell
 \startcell[align=flushright] \texttt{sum(C)} \stopcell
 \stoprow
 \startrow
 \startcell[align=flushleft] \textbf{total two} \stopcell
 \startcell[align=flushleft] \texttt{= \texttt{\texttt{\texttt{\texttt{sum(C)}}}}}' \stopcell
 \startcell[align=flushright] = \texttt{sum(C)} \stopcell
 \stoprow
\end{verbatim}

The \texttt{sum} function is clever enough not to include itself in the summation. Only preceding cells are taken into account, given that they represent a number.

\section{Normal tables}

In the previous examples we used \LaTeX\ commands for structuring the sheet but the content of cells is Lua code. It is also possible to stick to a regular table and use specific commands to set and get cell data.
Figure 5 Cells can be hidden by ! and can contain strings only.

The method to use depends on the complexity of the table. If there is more text than data then this method is probably more comfortable.

<table>
<thead>
<tr>
<th>first</th>
<th>@ &quot;[@i]&quot; 1</th>
<th>[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>second</td>
<td>= 2</td>
<td>2</td>
</tr>
<tr>
<td>third</td>
<td>! 3</td>
<td></td>
</tr>
<tr>
<td>fourth</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>total one</td>
<td>sum(C)</td>
<td>10</td>
</tr>
<tr>
<td>total two</td>
<td>= sum(C)</td>
<td>20</td>
</tr>
</tbody>
</table>

Figure 6 A sheet can be filled and accessed from regular tables.

4 A few settings

It’s possible to influence the rendering. The following example demonstrates this. We don’t use any
formatting directives.

\startspreadsheettable[test]
  \startrow
  \startcell 123456.78 \stopcell
  \stoprow
  \startrow
  \startcell 1234567.89 \stopcell
  \stoprow
  \startrow
  \stoprow
\stopspreadsheettable

\begin{tabular}{|c|}
  \hline
  123456.78  \\
  1234567.89  \\
  1358024.67  \\
  \hline
\end{tabular}

\textbf{Figure 7} Formatting (large) numbers.

Figure 7 demonstrates how this gets rendered by default. However, often you want numbers to be split in parts separated by periods and commas. This can be done as follows:

#definehighlight[BoldAndRed] [style=bold,color=darkred]
#definehighlight[BoldAndGreen][style=bold,color=darkgreen]

\setupspreadsheet
  [test]
  [period={\BoldAndRed{.}},
   comma={\BoldAndGreen{,}},
   split=yes]

\begin{tabular}{|c|}
  \hline
  123,456.78  \\
  1,234,567.89  \\
  1,358,024.67  \\
  \hline
\end{tabular}

\textbf{Figure 8} Formatting (large) numbers with style and color.

\section{5 The LUA end}

You can also use spreadsheets from within Lua. The following example is rather straightforward:

\startluacode
context.startspreadsheettable { "test" }
  context.startrow()
    context.startcell() context("123456.78") context.stopcell()
  context.stoprow()
context.startrow()
However, even more Lua-ish is the next variant:

\startluacode
local set = moduledata.spreadsheets.set
local get = moduledata.spreadsheets.get

moduledata.spreadsheets.start("test")
set("test",1,1,"123456.78")
set("test",2,1,"1234567.89")
moduledata.spreadsheets.stop()

context.bTABLE()
context.bTR()
  context.bTD() context(get("test",1,1)) context.eTD()
context.eTR()
context.bTR()
  context.bTD() context(get("test",2,1)) context.eTD()
context.eTR()
context.bTR()
  context.bTD() context(get("test",3,1)) context.eTD()
context.eTR()
context.eTABLE()
\stopluacode

Of course the second variant does not make much sense as we can do this way more efficient by not using a spreadsheet at all:

\startluacode
local A1, A2 = 123456.78, 1234567.89
context.bTABLE()
context.bTR()
  context.bTD() context(A1) context.eTD()
context.eTR()
context.bTR()
  context.bTD() context(A2) context.eTD()
context.eTR()
context.bTR()
  context.bTD() context(A1+A2) context.eTD()
context.eTR()
context.eTABLE()
\stopluacode
You can of course use format explicitly. Here we use the normal percent directives because we’re in Lua, and not in \TeX, where percentage signs are a bit of an issue.

\startluacode
    local A1, A2 = 123456.78, 1234567.89
    local options = { align = "flushright" }
    context.bTABLE()
    context.bTR()
    context.bTD(options)
    context("%.2f",A1)
    context.eTD()
    context.eTR()
    context.bTR()
    context.bTD(options)
    context("%.2f",A2)
    context.eTD()
    context.eTR()
    context.bTR()
    context.bTD(options)
    context("%.2f",A1+A2)
    context.eTD()
    context.eTR()
    context.eTABLE()
\stopluacode

As expected and shown in figure 9, only the first and last variant gets the numbers typeset nicely.

\begin{table}
\centering
\begin{tabular}{llll}
1\textasciitilde & 123\textasciitilde & 456\textasciitilde & 78\textasciitilde \\
1\textasciitilde & 234\textasciitilde & 567\textasciitilde & 89\textasciitilde \\
1\textasciitilde & 358\textasciitilde & 024\textasciitilde & 67\textasciitilde \\
\end{tabular}
\caption{Spreadsheets purely done as Con\TeX{} Lua Document.}
\end{table}

6 Helper macros

There are two helper macros that you can use to see what is stored in a spreadsheet:

\inspectspreadsheet\[test\]
\showspreadsheet\[test\]

The first command reports the content of test to the console, and the second one typesets it in the running text:

\begin{verbatim}
t={
    { 123456.78, 1234567.89, 1358024.67 },
}
\end{verbatim}

Another helper function is \doifelsespreadsheetcell, You can use this one to check if a cell is set.

(1,1): \doifelsespreadsheetcell\[test\]{1}{1}{set}{unset}
\( \text{(2,2): } \\textbackslash \text{doifelsespreadsheetcell}[\text{test}][2][2]\{\text{set}\}\{\text{unset}\} \)

\( \text{(9,9): } \\textbackslash \text{doifelsespreadsheetcell}[\text{test}][9][9]\{\text{set}\}\{\text{unset}\} \)

This gives:

\( \text{(1,1): set} \)

\( \text{(2,2): unset} \)

\( \text{(9,9): unset} \)

There is not much more to say about this module, apart from that it is a nice example of a \LaTeX\ and Lua mix. Maybe some more (basic) functionality will be added in the future but it all depends on usage.

**Colofon**

**author** Hans Hagen, PRAGMA ADE, Hasselt NL  
**version** May 12, 2016  
**website** [www.pragma-ade.nl](http://www.pragma-ade.nl) - [www.contextgarden.net](http://www.contextgarden.net)  
**copyright** cc-by-sa-nc