semesterplanner-lua — Semesterplanner package in lua with tikz only*

Lukas Heindl

: https://gitlab.com/AtticusSullivan/semesterplanner-lua

Released ?

TODO documentation TODO documentation

Abstract

This package provides a mean to easily print a timetable e.g. for a semesterplan. The reason for this package to exist is that I wanted to reimplement https://github.com/nlschn/semesterplanner/ with printing the timetable with `tikz` only (which is more easily to be modified) and with the ability to make entries spanning only a fraction of the column (for showing simultanious events).

Documents using this package need to be compiled with LuaLaTeX. The package requires `xcolor`, `fontawesome`, `tikz` (and `pgfkeys`).

Contents

1 Usage

1.1 timetable

timetable   `\begin{timetable}[opts]\ldots\end{timetable}`
opts are of course optional arguments:

days List of the names of the days that should be set as column names. Note that if you specify only 4 names only these 4 columns will be printed (with the first day being identified as Monday) *Default:* `Mon,Thue,Wend,Thur,Fri`

---

*This file describes version ?, last revised ?.

start time Explicit start-time of the timetable given in minutes (`HH*60 + MM`). Can be set as `start time/.evaluated={HH*60 + MM}`. If this is empty, the start time is derived from the given events. *Default: ""*

end time Equivalent to `start-time` *Default: ""*

width Give the width of the timetable. (can be given e.g. as `\textwidth` as this is directly given to tikz). *Default:* `\textwidth`

length Give the length of the timetable (measured in `cm`) (has to be a straight number since this is needed in calculation) *Default: 10*

This is the core environment of this package. Within it you can use `\lecture`, `\seminar`, `\tutorial`, `\officehour` and `\meeting`. All these commands are only defined inside the `timetable` environment, and have the same structure.

`\lecture`	`\lecture    {Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}`
`\tutorial`	`\tutorial   {Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}`
`\seminar`	`\seminar    {Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}`
`\officehour`	`\officehour{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}`
`\meeting`	`\meeting    {Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}`

Name Give the name of the lecture

Lecturer Give the name of the lecturer

Place Give the place of the event (most probably the room or an online plattform, see 1.2)

Day The weekday on which the event takes place. Has to be one of M, T, W, Th, F for **M**onday, **T**huesday, **W**ednesday, **Th**ursday, **F**riday. Might become customizable in a future version.

Time The timespan of the event formatted as `HH:MM-HH:MM` (24H clock)

Priority The priority of the event (see 1.2)

Event-code Free customizable event code. See the documentation at the end for keys that can be used here (all keys in `/event`). To simply pass arguments to the tikz-node that is being created for the event use `tikz/.append={your arguments}` (be careful with `text width`, `text height`, `text depth` as these keys are being used for the dimensions of the node as well as with `anchor`)

The entries `Day` and `Time` are mandatory since they are needed for the positioning of the node. All others are merely necessary for the content of the node and are therefore nor mandatory.

1.1.1 Special Notes

Note that the `length` argument does specify the length of the timetable without taking account of the column headers.

Same goes for the `width` parameter regarding the labels containing the time on the right. Since in this case any tex-lenght is allowed, you can simply try to subtract the length of the clock label using something like `\settowidth{\length}{12:30}` to set a length to the length of a clock label and then subtract this from the length you want to specify.

Hint: The content of the environment isn't processed by this package. Only the event commands (so to speak `\lecture,\tutorial,\seminar,\officehour,\meeting` are relevant. All other contents are set immediately before the timetable. Therefore, if you wan to add e.g. a `\hspace*{10cm}` to shift the timetable to the left, the last line of the env would be the place to do so (there musn't be an empty line below since otherwise a new paragraph is started).

1.1.2 Example

```
\begin{timetable}[
        days={Mon,Thue,Wend,Thur,Fri}, start
        time/.evaluated={11*60}, end time/.evaluated={15*60}
    ]
    \lecture[title={TestingLectureLongOne},speaker={Heindl},location={RN1},day={W},time
13:30}]
    \lecture[title={TestingLectureLongOne},speaker={Heindl},location={RN1},day={Th},tim
13:30},offset=0.5,scale width=0.5]
    \lecture[title={TestingLectureLongOne},speaker={Heindl},location={\zoom},day={T},ti
13:30},prio={\phigh}]
\end{timetable}
```

🕑 Timetable

1.2 Icons

This package defines some modified fontawesome icons (they are being encircled with a white circle for better readability).

\zoom	📷	\teams	⊞
\BBB	**B**	\youtube	▶
\pmandatory	⚠	\phigh	🚩
\pmid	🚩	\plow	🚩
\pnone	⊗		
\tbd	**?**	\tba	📢

2 Implementation

This package uses `semesterplanner-lua` as prefix/directory where possible. Since this is not possible for latex macro names, in this occasions `semesterplannerLua@` is used as prefix.

2.1 semesterplanner-lua.sty

2.1.1 Global Stuff

1 ⟨∗package⟩

Define some colors for the course types (can be globally overwritten)

```
2 \definecolor{seminar}{rgb}{1.0, 0.8, 0.0}
3 \definecolor{lecture}{rgb}{0.2, 0.7, 1.0}
4 \definecolor{tutorial}{rgb}{0.0, 0.8, 0.0}
5 \definecolor{meeting}{rgb}{0.8, 0.0, 0.0}
6 \definecolor{officehour}{rgb}{0.0, 0.4, 0.6}
7 \definecolor{DodgerBlue}{HTML}{1E90FF}
```

`\semesterplannerLua@encircle` This macro puts a circle around its argument for better readability. In this package this is used for the fontawesome symbols.

```
 8     \newcommand*{\semesterplannerLua@encircle}[1]{
 9         \begin{minipage}[b][1em][c]{1.5em}
10             \begin{tikzpicture}
11                 \node[fill,circle,inner sep=1pt, color = white] {#1};
12             \end{tikzpicture}
13         \end{minipage}
14     }
```

Commands for exams

`\oral`

15 \protected\def\oral{\faComment}

`\written`

16 \protected\def\written{\faPencil}

Commands for symbols of priority

`\pmandatory`

17   \protected\def\pmandatory{\semesterplannerLua@encircle{\textcolor{red}{\faWarning}}}

`\phigh`

18   \protected\def\phigh{\semesterplannerLua@encircle{\textcolor{red}{\faFlag}}}

`\pmid`

19   \protected\def\pmid{\semesterplannerLua@encircle{\textcolor{yellow}{\faFlag}}}

`\plow`

20   \protected\def\plow{\semesterplannerLua@encircle{\textcolor{green}{\faFlag}}}

`\pnone`

21   \protected\def\pnone{\semesterplannerLua@encircle{\textcolor{gray}{\faTimesCircle}}}

Commands for online platforms.

`\teams`

22   \protected\def\teams{\semesterplannerLua@encircle{\textcolor{DodgerBlue}{\faWindows}}}

`\zoom`

23   \protected\def\zoom{\semesterplannerLua@encircle{\textcolor{DodgerBlue}{\faCamera}}}

<dl>
<dt>\youtube</dt>
<dd>

24 `\protected\def\youtube{\semesterplannerLua@encircle{\textcolor{red}{\faYoutubePlay}}}`
</dd>

<dt>\BBB</dt>
<dd>

25 `\protected\def\BBB{\semesterplannerLua@encircle{\textcolor{DodgerBlue}{\faBold}}}`
</dd>
</dl>

Command for "To be determined" and "To be Announced"

<dl>
<dt>\tbd</dt>
<dd>

26 `\protected\def\tbd{\faQuestion}`
</dd>

<dt>\tba</dt>
<dd>

27 `\protected\def\tba{\faBullhorn}`
</dd>
</dl>

Load the lua modules

```
28 \directlua{sp = require("semesterplanner-lua-timetable.lua")}
29 \directlua{cal = require("semesterplanner-lua-calendar.lua")}
```

Set all the pgfkeys required for the arguments. To achieve that the defaults are restored every time the environment is used, this is inside the environment definition. This of course disables all possibilities of setting a global default but enables setting local defaults for the events

```
30     \pgfkeys{
```

`/semesterplanner-lua` will be the pgf-path used for this package. Here all used keys are set (and initialized with defaults. `timetable/env/`:

days is a list of strings representing the header names for the day columns in the timetable (adding Sat and Sun (additional entries) will result in two more columns.

start time can be used to set a fixed time where the timetable starts (otherwise this is calculated from the entries) to enable this behaviour this key has to be set to `HH*60 + MM` (easy way is by using `start time/.evaluated={HH*60+MM}`)

end time equivalent to `start time`

width is the horizontal width of the timetable (not including the column headers on the top) this can be a latex length string or `\textwidth` as well.

length is the vertical length of the timetable (not including the clock labels on the side) measured in cm (in future versions this may become measured in pts for better interaction with the LaTeX lengths.

```
31         /semesterplanner-lua/timetable/env/.cd,
32         days/.initial={Mon,Thue,Wend,Thur,Fri}, days/.default={Mon,Thue,Wend,Thur,Fri},
33         dayse/.initial={M,T,W,Th,F}, dayse/.default={M,T,W,Th,F},
34         %
35         start time/.initial=, start time/.default=,
36         end time/.initial=, end time/.default=,
37         %
38         width/.initial=\textwidth, width/.default=\textwidth,
39         length/.initial=10, length/.default=10,
40         %
```

`timetable/event/`:

content is the content of the event (is passed on without any formatting). Since this is passed to lua without modification its value must be an unexpanded string (lua will simply print it so the eventually the string will be evaluated)

time is a `HH:MM-HH:MM` string representing start- and end-time of the event. Used in constructing the content as well

day is either `M,T,W,Th` or `F` specifying the day on which the event takes place

tikz this key allows the user to manually pass options to the node created for this event

5

scale width allows to scale the width of the event to be able to draw overlapping events besides each other. Will usually be a value between 0 and 1.

offset same goal like `scale width` but shifts the event node by the given value to the right. (Given as value between 0 and 1 indicating how many columns the event should be shifted)

textcolor foreground color of the content text

title title (set in bold by default)

speaker

location

prio

formatter this is special

```
41          /semesterplanner-lua/timetable/event/.cd,
42          % event arguments
43          content/.initial=, content/.default=,
44          %
45          time/.initial=, time/.default=,
46          day/.initial=, day/.default=,
47          %
48          tikz/.initial=, tikz/.default=,
49          scale width/.initial=1, scale width/.default=1,
50          offset/.initial=0, offset/.default=0,
51          %
52          textcolor/.initial=, textcolor/.default=,
53          title/.initial=, title/.default=,
54          speaker/.initial=, speaker/.default=,
55          location/.initial=, location/.default=,
56          password/.initial=, password/.default=,
57          prio/.initial=, prio/.default=,
58          type/.initial=, type/.default=,
59          formatter/.initial=timetableformatter, formatter/.default=timetableformatter,
60          %
```

calendar/:

draw

room

prio

course

desc

start

end

tikz

period

shift

print Only makes sence if the command is suffixed by a `%` otherwise somehow a space gets inserted (eventhough the `%` is inserted from lua as well

```
61          /semesterplanner-lua/calendar/.cd,
62          draw/.initial={true}, draw/.default={true},
63          room/.initial={}, room/.default={},
64          time/.initial={}, time/.default={},
65          prio/.initial={}, prio/.default={},
66          course/.initial={}, course/.default={},
67          desc/.initial={}, desc/.default={},
68          type/.initial={}, type/.default={},
69          date/.initial={}, date/.default={},
70          end/.initial={}, end/.default={},
71          tikz/.initial={}, tikz/.default={},
72          period/.initial={nil}, period/.default={nil},
73          shift/.initial={true}, shift/.default={true},
74          print/.initial={true}, print/.default={true},
75      }
```

2.2   Tikz Calendar add weekday labels

```
76 \tikzoption{day headings}{\tikzstyle{day heading}=[#1]}
77 \tikzstyle{day heading}=[]
78 \tikzstyle{day letter headings}=[
79    execute before day scope={ \ifdate{day of month=1}{%
80      \pgfmathsetlength{\pgf@ya}{\tikz@lib@cal@yshift}%
81      \pgfmathsetlength\pgf@xa{\tikz@lib@cal@xshift}%
82      \pgftransformyshift{-\pgf@ya}
83      \foreach \d/\l in {0/M,1/T,2/W,3/T,4/F,5/S,6/S} {
84        \pgf@xa=\d\pgf@xa%
85        \pgftransformxshift{\pgf@xa}%
86        \pgftransformyshift{\pgf@ya}%
87        \node[every day,day heading]{\l};%
88      }
89    }{}%
90  }%
91 ]
```

2.2.1   Local Stuff (timetable-env local)

timetable   This is the environment doing all the stuff. To gate the positions where the corresponding macros can be used (and in terms of pgfkeys for reasons of default values) all the macros used are put into the environment.

```
92 \newenvironment{timetable}[1][]{
93    \section*{\faClockO~Timetable}
```

Read the argumens given by the user after restoring the defaults (Restoring currently makes no sense, since they are created a few lines above anyways, but creation might be moved outside the environment some day.
Afterwards the lua module is beeing initialized (erase data from possible previous runs.

```
94    \pgfkeys{/semesterplanner-lua/timetable/env/.cd, days,dayse, start time,end time, width,l
95    \directlua{sp.init{
96      days=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/days}]],
97      min=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/start time}]],
98      max=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/end time}]],
99      dayse=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/dayse}]]}}}
```

\semesterplanner@event   Is used to pass the event to the lua engine which in turn will collect the event to draw it in the end. For that the arguments given are parsed after restoring the pgf keys to their default values. The optional argument herby is a sequence of pgf keys, the second argument is a string representing the content (this MUST be unexpanded since this is passed to lua which in turn will pass it unmodified back)

```
100    \newcommand{\semesterplannerLua@event}[1][]{
101      \pgfkeys{/semesterplanner-lua/timetable/event/.cd,content,time,day,tikz,scale
102      width,offset,textcolor,title,speaker,type,location,password,prio,formatter, ##1}
103      \directlua{
104        sp.addEvent{
```

7

```
105            time="\pgfkeysvalueof{/semesterplanner-lua/timetable/event/time}",
106            day="\pgfkeysvalueof{/semesterplanner-lua/timetable/event/day}",
107            tikz=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/tikz}]],
108            offset=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/offset},
109            scale_width=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/scale width}
110            formatter=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/formatter},
111            textcolor=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/textcolor}]]
112            title=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/title}]],
113            speaker=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/speaker}]],
114            location=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/location}]],
115            password=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/password}]]],
116            prio=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/prio}]],
117            type=[[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/type}]],
118        }
119     }
120  }
```

Short-hand macros for different events using the corresponding background color

`\lecture`

```
121  \newcommand{\lecture}[1][]{
122      \semesterplannerLua@event[tikz={fill=lecture,}, textcolor=white, type=lect, ##1]
123      \ignorespaces
124  }
```

`\seminar`

```
125  \newcommand{\seminar}[1][]{
126      \semesterplannerLua@event[tikz={fill=seminar,}, textcolor=white, type=sem, ##1]
127      \ignorespaces
128  }
```

`\tutorial`

```
129  \newcommand{\tutorial}[1][]{
130      \semesterplannerLua@event[tikz={fill=tutorial,}, textcolor=white, type=tut, ##1]
131      \ignorespaces
132  }
```

`\meeting`

```
133  \newcommand{\meeting}[1][]{
134      \semesterplannerLua@event[tikz={fill=meeting,}, textcolor=white, type=meet, ##1]
135      \ignorespaces
136  }
```

`\officehour`

```
137  \newcommand{\officehour}[1][]{
138      \semesterplannerLua@event[tikz={fill=officehour,}, textcolor=white, type=office, ##1]
139      \ignorespaces
140  }
```

```
141 }{
```

At the end of the environment after all events have been collected, generate and output the tikz code needed to draw the timetable.

```
142     \directlua{sp.draw(
143         [[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/length}]],
144         [[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/width}]])}
145 }
```

```
146
```

`printSpCalendar` Print a calendar from startDate to endDate (encoded as YYYY-MM-DD) as one calendar per month in a matrix with the given amount of columns

```
147 \newcommand{\printSpCalendar}[3][3]{\directlua{cal.drawCalendar("#2", "#3", #1)}}
```

```latex
148
149 \newenvironment{appointments}[2][Room]{
150     \directlua{cal.init(#2)}
151     \newcommand{\appointment}[1][]{%
152         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,t:
153         \directlua{
154             cal.addAppointment{draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},
155             room=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/room}]],
156             time=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/time}]],
157             prio=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/prio}]],
158             course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
159             desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
160             date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
161             endDate=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/end}]],
162             tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
163             period=\pgfkeysvalueof{/semesterplanner-lua/calendar/period},
164             shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
165             print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
166         \ignorespaces
167         }
168     \section*{\faCalendar~Appointments}
169     \begin{tabular}{rlllll}
170         \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Description}&\textbf{#1}&\textbf{]
171 }{
172     \end{tabular}
173 }
174
175 \newenvironment{exams}[1]{
176     \directlua{cal.init(#1)}
177     \newcommand{\exam}[1][]{%
178         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,t:
179         \directlua{
180             cal.addExam{
181                 draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},
182                 room=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/room}]],
183                 time=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/time}]],
184                 course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
185                 desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
186                 date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
187                 tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
188                 type=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/type}]],
189                 shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
190                 print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
191         \ignorespaces
192         }
193     \section*{\faStickyNoteO~Exams}
194     \begin{tabular}{rllll}
195         \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Type}&\textbf{Note}\\
196 }{
197     \end{tabular}
198 }
199
200 \newenvironment{deadlines}[1]{
201     \directlua{cal.init(#1)}
202     \newcommand{\deadline}[1][]{%
203         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,t:
204         \directlua{
205             cal.addDeadline{
206                 draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},
207                 course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
208                 desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
209                 date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
210                 tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
```

```
211                    prio=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/prio}]],
212                    shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
213                    print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
214            \ignorespaces
215            }
216    \section*{\faStickyNoteO~Deadlines}
217    \begin{tabular}{rlll}
218        \textbf{Date}&\textbf{Course}&\textbf{Description}&\textbf{Prio}\\
219 }{
220    \end{tabular}
221 }

222 ⟨/package⟩
```

2.3   semesterplanner-lua-timetable.lua

223 ⟨∗luaTimetable⟩

init Initialize global variables to remove previous values (e.g. events from the previous timetable)

days A string with the names of the weekdays for the header

min Time where the timetable should start. If empty this is calculated from the events.

max Time where the timetable should end. If empty this is calculated from the events.

```
224 function init(opts)
225     tex.print([[\newwrite\timetableATdataOutput \immediate\openout\timetableATdataOutput=\job
    data.dat]])
226     if(not checkKeys(opts, {"days", "min", "max", "dayse"})) then
227         error("missing argument")
228     end
229     -- clean up first
230     -- global variables
231     EVENTS={}
232     DAYS  = prepareDays(opts.days) -- header with names of the days set from tex currently
233     DAYSE = prepareDays(opts.dayse) -- day representation in source code
234     MIN = 25*60 -- bigger than any allowed value could be
235     MAX = 0
236     MIN_BYPASS = false -- weather min is fixed by the user
237     MAX_BYPASS = false -- weather max is fixed by the user
238
239     if(opts.min == "") then
240     else
241         assert(opts.min:match("^%d+"), "start time has to be an integer representing the HH*60
242         MIN = tonumber(opts.min)
243         MIN_BYPASS = true
244     end
245
246     if(opts.max == "") then
247     else
248         assert(opts.max:match("^%d+"), "end time has to be an integer representing the HH*60+
249         MAX = tonumber(opts.max)
250         MAX_BYPASS = true
251     end
252 end
253
254 function defaultFormatter(opts)
255     local ret = ""
256     for k,v in pairs(opts) do
257         if type(k) == "string" then k = k:gsub("[_^]", "") end
258         if type(v) == "string" then v = v:gsub("[_^]", "") end
259         ret = string.format("%s, %s: %s", ret, tostring(k), tostring(v))
260     end
261     print(ret)
```

```
262    return ret
263 end
264
265 function timetableformatter(opts)
266    return string.format(
267        [[\textcolor{%s}{\textbf{%s}}\\[.2em]\raggedright{%s}\\[0.5em]\raggedright{%s}\hfil\ra
268            opts.textcolor, opts.title, opts.speaker, opts.prio, opts.location, opts.time)
269 end
```

addEvent   Adds the event to the EVENTS array after some validiy checks, modifys MIN/MAX if necessary

```
270 -- result are the global variables EVENTS, MIN and MAX
271 function addEvent(opts)
272    print("Reading event on line ", tex.inputlineno)
273    opts.inputlineno = tex.inputlineno
274    if(not checkKeys(opts, {"time", "day", "tikz"})) then
275        error("missing argument")
276    end
277
278    if opts.content == nil then
279        if opts.formatter == nil then
280            opts.content = defaultFormatter(opts)
281        else
282            opts.content = opts.formatter(opts)
283        end
284    end
285
286    opts.from,opts.to = dur2Int(opts.time)
287
288    tex.print(string.format([[\immediate\write\timetableATdataOutput{\unexpanded{%s;%s;%s;%s;
289
290    if(not MIN_BYPASS and opts.from < MIN) then MIN = opts.from end
291    if(not MAX_BYPASS and opts.to   > MAX) then MAX = opts.to    end
292    assert(opts.from < opts.to, "From has to be before to")
293
294    table.insert(EVENTS, opts)
295 end
```

draw   Draws the tikz-timetable with the global variables EVENTS, MIN, MAX, DAYSE and DAYS. In addition `length` and `width` are given as direct parameters.

```
296 -- parameters are all global variables
297 function draw(length, width)
298    -- copy relevant variables for working on local copies
299    local events = copy_array(EVENTS)
300    local days = copy_array(DAYS)
301    local min, minH, max, maxH = prepareMinMax(MIN, MAX)
302
303    assert(length:match("%d*%.?%d*"), "Length must be a valid length measured in cm")
304    length = tonumber(length)
305
306    textwidth = width
307
308    tex.print([[\begin{tikzpicture}]])
309    tex.print([[\tikzset{defStyle/.style={font=\tiny,anchor=north west,fill=blue!50,draw=blac
```

Draw the grid of the timetable along with clock and day labels

```
310    -- print the tabular with the weekday headers
311    tex.print(string.format(
312        [[\foreach \week [count=\x from 0, evaluate=\x as \y using \x+0.5] in {%s}{ ]],
313        table.concat(days, ",")
314        )
315    )
316    tex.print(string.format(
317        [[\node[anchor=south] at (\y/%d* %s, 0) {\week};]], #days, textwidth))
```

```
318    tex.print(string.format(
319        [[\draw (\x/%d * %s, 0cm) -- (\x/%d * %s, %dcm);]],
320        #days,
321        textwidth,
322        #days,
323        textwidth, -length
324        )
325    )
326    tex.print("}")
327    tex.print(string.format(
328        [[\draw (%s, 0) -- (%s,%dcm);]],
329        textwidth,
330        textwidth,
331        -length
332        )
333    )
334
335    for i=minH,maxH do
336        tex.print(string.format(
337            [[\node[anchor=east] at (0,%fcm ) {%d:00};]],
338            minuteToFrac(i*60,min,max)*-length, i
339            )
340        )
341        tex.print(string.format(
342            [[\draw (0,%fcm ) -- (%s,%fcm );]],
343            minuteToFrac(i*60,min,max)*-length,
344            textwidth,
345            minuteToFrac(i*60,min,max)*-length
346            )
347        )
348    end
349
```

Draw the nodes of the events

```
350    local d
351    local red = 0.3333 -- calculated in em from inner sep
352    local red_y = 0.25 -- calculated in em
353    for _,e in ipairs(events) do
354        if e.from < max and e.to > min then -- only draw if event is in scope (part of the co
355            if e.to   > max then e.to   = max end
356            if e.from < min then e.from = min end
357            print("Drawing event on line ", e.inputlineno)
358            d = day2Int(e.day)
359            tex.print(string.format(
360                [[\node[defStyle,text width=-%fem+%f%s/%d, text depth=%fcm-%fem, text height=%
361                2*red, -- text width
362                e.scale_width, -- text width
363                textwidth,
364                #days, -- text width
365                length*(e.to-e.from)/(max-min), -- text depth
366                2*red+red_y, -- text depth
367                red_y, -- text height
368                e.tikz, -- free tikz code
369                (d+e.offset)/#days, -- xcoord
370                textwidth,
371                minuteToFrac(e.from,min,max)*-length, -- ycoord
372                e.content -- content
373                )
374            )
375        end
376    end
377    tex.print([[\end{tikzpicture}]])
378 end
```

searchArray	Searches an array for a given value and returns the index if found. On error `nil` is returned

```
379 function search_array(t, s)
380     for k,v in ipairs(t) do
381         if(v == s) then return k end
382     end
383     return nil
384 end
385
```

minuteToFrac	Calculates at which fraction of the total duration of `max-min` the time `minute` is located

```
386 function minuteToFrac(minute, min, max)
387     return (minute-min)/(max-min)
388 end
```

prepareMinMax	Calculates the next hour of `MIN` (next before) and `MAX` (next after) and returns it (the hour) and the corresponding `min/max` (same in minutes)

```
389 function prepareMinMax(min, max)
390     local minH = math.floor(min/60)
391     local maxH = math.ceil(max/60)
392     local min = minH*60
393     local max = maxH*60
394     return min, minH, max, maxH
395 end
```

checkKeys	Checks if all `ks` are present in table `t`

```
396 function checkKeys(t, k)
397     for _,x in ipairs(k) do
398         if(t[x] == nil) then
399             return false
400         end
401     end
402     return true
403 end
```

dur2Int	Takes a clock duration formatted as `HH:MM-HH:MM`, splits it, checks for validity and returns begin/end time in minutes

```
404 function dur2Int(clk)
405     local f1,f2, t1,t2 = clk:match("^(%d%d?):(%d%d)-(%d%d?):(%d%d)$")
406     if(f1 ~= nil and f2 ~= nil and t1 ~= nil and t2 ~= nil) then
407         f1 = tonumber(f1) f2 = tonumber(f2)
408         t1 = tonumber(t1) t2 = tonumber(t2)
409         assert(f1 >= 0 and f1 < 24, "Hours have to be >= 0 && < 24")
410         assert(f2 >= 0 and f2 < 60, "Mins have to be >= 0 && < 60")
411         assert(t1 >= 0 and t1 < 24, "Hours have to be >= 0 && < 24")
412         assert(t2 >= 0 and t2 < 60, "Mins have to be >= 0 && < 60")
413         return f1*60 + f2, t1*60 + t2
414     else
415         error("clk string \"" .. clk .. "\" was no valid clock string")
416     end
417 end
```

prepareDays	Splits the comma-sep string `days` into an array

```
418 function prepareDays(days)
419     local ret = {}
420     for m in days:gmatch("[^,]+") do
421         table.insert(ret, m)
422     end
423     return ret
424 end
```

day2Int	converts the day-string to an integer (Monday is 0)

```
425 function day2Int(day)
426     return search_array(DAYSE, day) - 1
427 end
428 % \end{macro}
429 % \begin{macro}{copyArray}
430 % Returns a copy of the table |obj|
431 %    \begin{macrocode}
432
433 function copy_array(obj)
434     if type(obj) ~= 'table' then return obj end
435     local res = {}
436     for k, v in pairs(obj) do
437         local c = copy_array(v)
438         res[copy_array(k)] = c
439     end
440     return res
441 end
```

Prepare the module semesterplannerLua for exporting (only the functions that should be public)

```
442
443 semesterplannerLua = {
444     init = init,
445     addEvent = addEvent,
446     draw = draw,
447     day2Int = day2Int,
448 }
449 return semesterplannerLua
450 ⟨/luaTimetable⟩
```

2.4   semesterplanner-lua-calendar.lua

TODO how to set the paths right in this case Include the date module for time date calculations

```
451 ⟨*luaApp⟩
452 package.path='/usr/share/lua/5.3/?.lua;/usr/share/lua/5.3/?/init.lua;/usr/lib/lua/5.3/?.lua;/u
453 package.cpath='/usr/lib/lua/5.3/?.so;/usr/lib/lua/5.3/loadall.so;./?.so;/home/lukas/.luarocks,
454
455 local dateLib = require "date"
```

init   Initialize the EVENTS table as some sort of a reset, takes an argument wethet the reset should be executed (to enable concatenation)

```
456 function init(clear)
457     -- clean up first
458     -- global variable
459     if clear then
460         EVENTS = {}
461     end
462 end
463
464 text = {
465     print = function(s)
466         -- print("\"" .. s .. "\"")
467         tex.print(s)
468     end
469 }
470
471 function genDot(opts)
472     dot = ""
473     if opts.draw then
474         dot = string.format([[\tikz[baseline=(X.base)]\node (X) [fill opacity=.5,fill=red,circ
475     end
476     return dot
```

```
477 end
478
```

addEvent   Adds an event to the list, stores the date and how the event should be highlighted (tikz code for a node)

```
479 function addEvent(opts)
480     opts.inputlineno = tex.inputlineno
481     print(string.format("collecting from line %d", opts.inputlineno))
482     if opts.draw then
483         assert(opts.date ~= nil and opts.tikz ~= nil, "date and tikz has to be given")
484         if opts.endDate == nil or opts.endDate == '' then
485             table.insert(EVENTS, {shift=opts.shift,date=dateLib(opts.date), tikz=opts.tikz, p
486         else
487             table.insert(EVENTS, {shift=opts.shift,date=dateLib(opts.date), tikz=opts.tikz, p
488         end
489     end
490 end
491
492 function addAppointment(opts)
493     addEvent(opts)
494     dot = genDot(opts)
495     if opts.print then
496         tex.sprint(string.format([[\textit{%s} & %s & %s%s & %s & %s & %s\\]], opts.date, opt
497     else
498         tex.sprint("%")
499     end
500 end
501
502 function addExam(opts)
503     addEvent(opts)
504     dot = genDot(opts)
505     if opts.print then
506         tex.sprint(string.format([[\textit{%s} & %s & %s%s & %s & %s \\]], opts.date, opts.ti
507     else
508         tex.sprint("%")
509     end
510 end
511
512 function addDeadline(opts)
513     addEvent(opts)
514     dot = genDot(opts)
515     if opts.print then
516         tex.sprint(string.format([[\textit{%s} & %s%s & %s & %s \\]], opts.date, dot, opts.co
517     else
518         tex.sprint("%")
519     end
520 end
```

drawCalendar   Draw the calendar month by month in a matrix with given columns. The calendar starts and ends at the given dates (in YYYY-MM-DD or any other format the datelib understands)

```
521
522 function drawCalendar(minDate, maxDate, cols)
523     minDate = dateLib(minDate)
524     maxDate = dateLib(maxDate)
525     text.print([[\begin{tikzpicture}[every calendar/.style={day headings=red!50,day letter he
    }, every month/.style={yshift=3ex}}] ]])
526     text.print([[\matrix[column sep=1em, row sep=1em]{]])
527         local i = 1
528         running = true
529         while running do
530             -- derive end from start, then check if maxDate is reached
531             endDate = minDate:copy():addmonths(1):setday(1):adddays(-1)
```

```
532            if endDate >= maxDate then
533                endDate = maxDate
534                running = false
535            end
536            text.print(string.format(
537            [[\calendar (%04d-%02d) [dates=%04d-%02d-%02d to %04d-%02d-%02d] if (Sunday) [red]
   \month-\day) [nodes={inner sep=.25em,rectangle,line width=1pt,draw}] if (at least=\year-
   \month-\day) {} else [nodes={strike out, draw}]; ]],
538                    minDate:getyear(), minDate:getmonth(), minDate:getyear(), minDate:getmonth
539
540            minDate:addmonths(1)
541            minDate:setday(1)
542
543            if i % cols == 0 or not running then
544                text.print([[\\]])
545            else
546                text.print([[&]])
547            end
548            i = i + 1
549        end
550        text.print([[ }; ]])
551
```

Draw highlighting on a background layer so that the calendar is not overdrawn

```
552        local usedDates = {}
553        text.print([[\begin{scope}[on background layer] ]])
554        for i,ele in ipairs(EVENTS) do
555            print(string.format("Drawing item from line %d", ele.inputlineno))
556            while ele.date <= maxDate and (ele.endDate == nil or ele.date <= ele.endDate) do
557                local xshift = 0
558                if ele.shift then
559                    if usedDates[tostring(ele.date)] ~= nil then
560                        xshift = math.ceil(usedDates[tostring(ele.date)] / 2)
561                        if usedDates[tostring(ele.date)] % 2 == 0 then
562                            xshift = -xshift
563                        end
564                        usedDates[tostring(ele.date)] = usedDates[tostring(ele.date)] + 1
565                    else
566                        usedDates[tostring(ele.date)] = 1
567                    end
568                end
569                text.print(string.format([[\node[xshift=%d mm, fill opacity=.5,fill=red,circle
   %02d-%04d-%02d-%02d) {};]],
570                        xshift, ele.tikz, ele.date:getyear(), ele.date:getmonth(), ele.date:getye
571                if ele.period == nil then break end
572                ele.date:adddays(ele.period)
573            end
574        end
575        text.print([[\end{scope}]])
576    text.print([[\end{tikzpicture}]])
577 end
```

Prepare the module for exporting (only the functions that should be public)

```
578
579 semesterplannerLuaCal = {
580    init = init,
581    addAppointment = addAppointment,
582    addDeadline = addDeadline,
583    addExam = addExam,
584    drawCalendar = drawCalendar,
585 }
586 return semesterplannerLuaCal
587 ⟨/luaApp⟩
```

2.5 semesterplanner-lua-calendar.lua

TODO how to set the paths right in this case Include the date module for time date calculations

```
588 ⟨∗luaDmenu⟩
589 -- TODO install to e.g. `/usr/share/texmf-dist/scripts/`
590
591 local function dur2str(dur)
592     return string.format("%02d:%02d", dur//60, dur%60)
593 end
594
595 local maxes = {title=0, location=0, type=0}
596
597 local before = {diff=25*60*7}
598 local after  = {diff=25*60*7}
599
600 local c = {}
601
602 local current   = os.date("*t")
603 local today     = tonumber(os.date("%u")) - 1
604 current.minutes = current.min + (today*24 + current.hour)*60
605
606
607 local fields = {"start", "end", "title", "location", "password", "type"}
608 for line in  io.lines(arg[1]) do
609     local ele,i = {},1
610     for e in line:gmatch("[^;]*") do
611         ele[fields[i]] = e
612         i = i+1
613     end
614     ele.start, ele["end"] = tonumber(ele.start), tonumber(ele["end"])
615     ele.url      = ele.location:match([[\href ?{(.*)}{.*}]])
616     ele.location = ele.location:match([[\href ?{.*}{(.*)}]]) or ele.location
617
618     if maxes.title    < #ele.title    then maxes.title    = #ele.title    end
619     if maxes.location < #ele.location then maxes.location = #ele.location end
620     if maxes.type     < #ele.type     then maxes.type     = #ele.type     end
621
622     if current.minutes > ele.start and current.minutes < ele["end"] then
623         ele.diff = current.minutes-ele["start"]
624         table.insert(c, ele)
625     elseif current.minutes < ele.start then
626         if ele.start - current.minutes < before.diff then
627             before = {diff=ele.start-current.minutes}
628         end
629         if ele.start - current.minutes <= before.diff then
630             ele.diff = before.diff
631             table.insert(before, ele)
632         end
633     elseif current.minutes > ele["end"] then
634         if current.minutes - ele["end"] < after.diff then
635             after = {diff=current.minutes-ele["end"]}
636         end
637         if current.minutes - ele["end"] <= after.diff then
638             ele.diff = after.diff
639             table.insert(after, ele)
640         end
641     end
642 end
643
644 local dmenu_in,map = {},{}
645 for _,x in ipairs(after) do
646     local s = string.format("%-".. maxes.title+1+maxes.type .."s %-".. maxes.location+2 .."s
  "..x.type, "("..x.location..")", dur2str(x.diff))
```

```
647     table.insert(dmenu_in, s)
648     map[s] = x
649 end
650
651 for _,x in ipairs(c) do
652     local s = string.format("%-".. maxes.title+1+maxes.type .."s %-".. maxes.location+2 .."s
    "..x.type, "("..x.location..")", dur2str(x.diff))
653     table.insert(dmenu_in, s)
654     map[s] = x
655 end
656
657 for _,x in ipairs(before) do
658     local s = string.format("%-".. maxes.title+1+maxes.type .."s %-".. maxes.location+2 .."s
    "..x.type, "("..x.location..")", dur2str(x.diff))
659     table.insert(dmenu_in, s)
660     map[s] = x
661 end
662
663 local dmenu_ins = table.concat(dmenu_in, "\n")
664
665 print(dmenu_ins)
666 -- TODO check for forbidden strings in dmenu_in
667 local proc = io.popen(string.format("echo '%s' | dmenu -i -l %d -p 'Select entry'", dmenu_ins
668 local dmenu_out = proc:read("l")
669 if not dmenu_out then
670     os.execute("notify-send 'Uni' 'Nothing selected'")
671     os.exit()
672 end
673
674 local sel = map[dmenu_out]
675 assert(sel, "Error building/reading the lookup table")
676 if sel.url and sel.url ~= "" then
677     os.execute(string.format("echo -n '%s' | xsel -b && notify-send -h string:x-
    canonical-private-synchronous:uni 'Uni' 'Copied URL to clipboard'", sel.url))
678 elseif sel.location and sel.location ~= "" then
679     os.execute(string.format("echo -n 'https://portal.mytum.de/campus/roomfinder/search_room_
    b && notify-send -h string:x-canonical-private-synchronous:uni 'Uni' 'Copied Roomfinder-
    url to clipboard'", sel.location))
680 end
681
682 if sel.password and sel.password ~= "" then
683     os.execute(string.format("sleep 3 && echo -n '%s' | xsel -b && notify-send -
    h string:x-canonical-private-synchronous:uni 'Uni' 'Copied password to clipboard'", sel.passw
684 end
685 ⟨/luaDmenu⟩
```

3 Change History

4 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

18