

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

Lukas Heindl

🐙: <https://gitlab.com/AtticusSullivan/semesterplanner-lua>

Released ?

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 simultaneous events).

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

Contents

1	Usage	1
1.1	timetable	1
1.1.1	Special Notes	2
1.1.2	Example	3
1.2	Icons	3
2	Implementation	4
2.1	semesterplanner-lua.sty	4
2.1.1	Global Stuff	4
2.2	Tikz Calendar add weekday labels	7
2.2.1	Local Stuff (timetable-env local)	7
2.3	semesterplanner-lua-timetable.lua	10
2.4	semesterplanner-lua-calendar.lua	14
3	Change History	16
4	Index	16

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*

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: ""*

*This file describes version ?, last revised ?.

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.

<code>\lecture</code>	<code>\lecture</code>	<code>{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}</code>
<code>\tutorial</code>	<code>\tutorial</code>	<code>{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}</code>
<code>\seminar</code>	<code>\seminar</code>	<code>{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}</code>
<code>\officehour</code>	<code>\officehour</code>	<code>{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}</code>
<code>\meeting</code>	<code>\meeting</code>	<code>{Name}{Lecturer}{Place}{Day}{Time}{Priority}{Event-code}</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 platform, see [1.2](#))

Day The weekday on which the event takes place. Has to be one of M, T, W, Th, F for Monday, Tuesday, Wednesday, Thursday, Friday. 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 not 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-length 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 want 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 mustn'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},time
13:30},offset=0.5,scale width=0.5]
  \lecture[title={TestingLectureLongOne},speaker={Heindl},location={\zoom},day={T},time
13:30},prio={\phigh}]
\end{timetable}
```

⌚ Timetable

	Mon	Thue	Wend	Thur	Fri
11:00					
12:00					
13:00		<div>TestingLectureLongOne</div> <div>Heindl</div> <div>   </div> <div>12:30-13:30</div>	<div>TestingLectureLongOne</div> <div>Heindl</div> <div>RN1</div> <div>12:30-13:30</div>	<div>TestingLectureLongOne</div> <div>Heindl</div> <div>RN1</div> <div>12:30-13:30</div>	
14:00					
15:00					

1.2 Icons

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

<code>\zoom</code>		<code>\teams</code>	
<code>\BBB</code>		<code>\youtube</code>	
<code>\pmandatory</code>		<code>\phigh</code>	
<code>\pmid</code>		<code>\plow</code>	
<code>\pnone</code>			
<code>\tbd</code>		<code>\tba</code>	

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}}}
```

```

\youtube
24 \protected\def\youtube{\semesterplannerLua@encircle{\textcolor{red}{\faYoutubePlay}}}

\BBB
25 \protected\def\BBB{\semesterplannerLua@encircle{\textcolor{DodgerBlue}{\faBold}}}

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

\tbd
26 \protected\def\tbd{\faQuestion}

\tba
27 \protected\def\tba{\faBullhorn}

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}, days/.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

```

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      prio/.initial=, prio/.default=,  
57      formatter/.initial=timetableformatter, formatter/.default=timetableformatter,  
58      %
```

calendar/:

draw

room

prio

course

desc

start

end

tikz

period

shift

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

```
59      /semesterplanner-lua/calendar/.cd,  
60      draw/.initial={true}, draw/.default={true},  
61      room/.initial={}, room/.default={},  
62      time/.initial={}, time/.default={},  
63      prio/.initial={}, prio/.default={},
```

```

64     course/.initial={}, course/.default={},
65     desc/.initial={}, desc/.default={},
66     type/.initial={}, type/.default={},
67     date/.initial={}, date/.default={},
68     end/.initial={}, end/.default={},
69     tikz/.initial={}, tikz/.default={},
70     period/.initial={nil}, period/.default={nil},
71     shift/.initial={true}, shift/.default={true},
72     print/.initial={true}, print/.default={true},
73 }

```

2.2 Tikz Calendar add weekday labels

```

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

```

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.

```

90 \newenvironment{timetable}[1][]{
91     \section*{\faClockO-Timetable}

```

Read the arguments 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.

```

92     \pgfkeys{/semesterplanner-lua/timetable/env/.cd, days,start time,end time, width,length,
93     \directlua{sp.init{
94         days=[[ \pgfkeysvalueof{/semesterplanner-lua/timetable/env/days}]],
95         min=[[ \pgfkeysvalueof{/semesterplanner-lua/timetable/env/start time}]],
96         max=[[ \pgfkeysvalueof{/semesterplanner-lua/timetable/env/end time}]],
97         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)

```

98     \newcommand{\semesterplannerLua@event}[1][]{
99         \pgfkeys{/semesterplanner-lua/timetable/event/.cd,content,time,day,tikz,scale
100         width,offset,textcolor,title,speaker,location,prio,formatter, ##1}
101         \directlua{
102             sp.addEvent{
103                 time="\pgfkeysvalueof{/semesterplanner-lua/timetable/event/time}",
104                 day="\pgfkeysvalueof{/semesterplanner-lua/timetable/event/day}",
105                 tikz=[[ \pgfkeysvalueof{/semesterplanner-lua/timetable/event/tikz}]],
106                 offset=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/offset},
107                 scale_width=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/scale width}

```

```

108         formatter=\pgfkeysvalueof{/semesterplanner-lua/timetable/event/formatter},
109         textcolor=[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/textcolor}],
110         title=[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/title}],
111         speaker=[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/speaker}],
112         location=[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/location}],
113         prio=[\pgfkeysvalueof{/semesterplanner-lua/timetable/event/prio}],
114     }
115 }
116 }

```

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

\lecture

```

117 \newcommand{\lecture}[1][]{
118     \semesterplannerLua@event[tikz={fill=lecture,}, textcolor=white, ##1]
119     \ignorespaces
120 }

```

\seminar

```

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

```

\tutorial

```

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

```

\meeting

```

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

```

\officehour

```

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

```

```

137 }{

```

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

```

138 \directlua{sp.draw(
139     [[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/length}]],
140     [[\pgfkeysvalueof{/semesterplanner-lua/timetable/env/width}]])}
141 }

```

```

142

```

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

```

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

```

```

144

```

```

145 \newenvironment{appointments}[2][Room]{

```

```

146     \directlua{cal.init(#2)}

```

```

147     \newcommand{\appointment}[1][]{%

```

```

148         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,t

```

```

149         \directlua{

```

```

150             cal.addAppointment{draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},

```



```

151         room=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/room}]],
152         time=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/time}]],
153         prio=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/prio}]],
154         course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
155         desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
156         date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
157         endDate=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/end}]],
158         tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
159         period=\pgfkeysvalueof{/semesterplanner-lua/calendar/period},
160         shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
161         print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
162     \ignorespaces
163 }
164 \section*{\faCalendar~Appointments}
165 \begin{tabular}{rlllll}
166     \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Description}&\textbf{\#1}&\textbf{\#2}
167 }{
168     \end{tabular}
169 }
170
171 \newenvironment{exams}[1]{
172     \directlua{cal.init(#1)}
173     \newcommand{\exam}[1][{}]{%
174         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,tikz}
175         \directlua{
176             cal.addExam{
177                 draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},
178                 room=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/room}]],
179                 time=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/time}]],
180                 course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
181                 desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
182                 date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
183                 tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
184                 type=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/type}]],
185                 shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
186                 print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
187             \ignorespaces
188         }
189     \section*{\faStickyNote0~Exams}
190     \begin{tabular}{rlllll}
191         \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Type}&\textbf{Note}
192     }{
193         \end{tabular}
194 }
195
196 \newenvironment{deadlines}[1]{
197     \directlua{cal.init(#1)}
198     \newcommand{\deadline}[1][{}]{%
199         \pgfkeys{/semesterplanner-lua/calendar/.cd,draw,room,time,prio,course,desc,date,end,tikz}
200         \directlua{
201             cal.addDeadline{
202                 draw=\pgfkeysvalueof{/semesterplanner-lua/calendar/draw},
203                 course=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/course}]],
204                 desc=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/desc}]],
205                 date=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/date}]],
206                 tikz=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/tikz}]],
207                 prio=[[\pgfkeysvalueof{/semesterplanner-lua/calendar/prio}]],
208                 shift=\pgfkeysvalueof{/semesterplanner-lua/calendar/shift},
209                 print=\pgfkeysvalueof{/semesterplanner-lua/calendar/print}}}%
210             \ignorespaces
211         }
212     \section*{\faStickyNote0~Deadlines}
213     \begin{tabular}{rlll}

```

```

214         \textbf{Date}&\textbf{Course}&\textbf{Description}&\textbf{Prio}\\
215     }{
216     \end{tabular}
217 }
218 \end{package}

```

2.3 semesterplanner-lua-timetable.lua

```

219 (*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.

220 function init(opts)
221     if(not checkKeys(opts, {"days", "min", "max", "dayse"})) then
222         error("missing argument")
223     end
224     -- clean up first
225     -- global variables
226     EVENTS={}
227     DAYS = prepareDays(opts.days) -- header with names of the days set from tex currently
228     DAYSE = prepareDays(opts.dayse) -- day representation in source code
229     MIN = 25*60 -- bigger than any allowed value could be
230     MAX = 0
231     MIN_BYPASS = false -- weather min is fixed by the user
232     MAX_BYPASS = false -- weather max is fixed by the user
233
234     if(opts.min == "") then
235     else
236         assert(opts.min:match("^%d+"), "start time has to be an integer representing the HH*60+
237         MIN = tonumber(opts.min)
238         MIN_BYPASS = true
239     end
240
241     if(opts.max == "") then
242     else
243         assert(opts.max:match("^%d+"), "end time has to be an integer representing the HH*60+
244         MAX = tonumber(opts.max)
245         MAX_BYPASS = true
246     end
247 end
248
249 function defaultFormatter(opts)
250     local ret = ""
251     for k,v in pairs(opts) do
252         if type(k) == "string" then k = k:gsub("[_]", "") end
253         if type(v) == "string" then v = v:gsub("[_]", "") end
254         ret = string.format("%s, %s: %s", ret, tostring(k), tostring(v))
255     end
256     print(ret)
257     return ret
258 end
259
260 function timetableformatter(opts)
261     return string.format(
262         [[\textcolor{%s}{\textbf{%s}}\[\[.2em\]\raggedright{%s}\][0.5em]\raggedright{%s}\raggedr
263         opts.textcolor, opts.title, opts.speaker, opts.prio, opts.location, opts.time)
264 end

```

addEvent Adds the event to the `EVENTS` array after some validity checks, modifies `MIN/MAX` if necessary

```

265 -- result are the global variables EVENTS, MIN and MAX
266 function addEvent(opts)
267     print("Reading event on line ", tex.inputlineno)
268     opts.inputlineno = tex.inputlineno
269     if(not checkKeys(opts, {"time", "day", "tikz"})) then
270         error("missing argument")
271     end
272
273     if opts.content == nil then
274         if opts.formatter == nil then
275             opts.content = defaultFormatter(opts)
276         else
277             opts.content = opts.formatter(opts)
278         end
279     end
280
281     opts.from,opts.to = dur2Int(opts.time)
282
283     if(not MIN_BYPASS and opts.from < MIN) then MIN = opts.from end
284     if(not MAX_BYPASS and opts.to > MAX) then MAX = opts.to end
285     assert(opts.from < opts.to, "From has to be before to")
286
287     table.insert(EVENTS, opts)
288 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.

```

289 -- parameters are all global variables
290 function draw(length, width)
291     -- copy relevant variables for working on local copies
292     local events = copy_array(EVENTS)
293     local days = copy_array(DAYS)
294     local daysE = copy_array(DAYSE)
295     local min, minH, max, maxH = prepareMinMax(MIN, MAX)
296
297     assert(length:match("%d*%.?%d*"), "Length must be a valid length measured in cm")
298     length = tonumber(length)
299
300     textwidth = width
301
302     tex.print([[\\begin{tikzpicture}]]])
303     tex.print([[\\tikzset{defStyle/.style={font=\tiny,anchor=north west,fill=blue!50,draw=black}}]])
304
305     Draw the grid of the timetable along with clock and day labels
306
307     -- print the tabular with the weekday headers
308     tex.print(string.format(
309         [[\\foreach \\week [count=\\x from 0, evaluate=\\x as \\y using \\x+0.5] in {%s}{ } ],
310         table.concat(days, ",")
311     ))
312
313     tex.print(string.format(
314         [[\\node[anchor=south] at (\\y/%d* %s, 0) {\\week};]], #days, textwidth))
315
316     tex.print(string.format(
317         [[\\draw (\\x/%d * %s, 0cm) -- (\\x/%d * %s, %dcm);]],
318         #days,
319         textwidth,
320         #days,
321         textwidth, -length
322     ))
323
324     tex.print("}")
325     tex.print(string.format(

```

```

322     [[\draw (%s, 0) -- (%s,%dcm);]],
323     textwidth,
324     textwidth,
325     -length
326   )
327 )
328
329 for i=minH,maxH do
330   tex.print(string.format(
331     [[\node[anchor=east] at (0,%fcm ) {%d:00};]],
332     minuteToFrac(i*60,min,max)*-length, i
333   )
334 )
335   tex.print(string.format(
336     [[\draw (0,%fcm ) -- (%s,%fcm );]],
337     minuteToFrac(i*60,min,max)*-length,
338     textwidth,
339     minuteToFrac(i*60,min,max)*-length
340   )
341 )
342 end
343
Draw the nodes of the events
344   local d
345   local red = 0.3333 -- calculated in em from inner sep
346   local red_y = 0.25 -- calculated in em
347   for _,e in ipairs(events) do
348     if e.from < max and e.to > min then -- only draw if event is in scope (part of the cor
349       if e.to > max then e.to = max end
350       if e.from < min then e.from = min end
351       print("Drawing event on line ", e.inputlineno)
352       d = search_array(daysE, e.day) - 1
353       tex.print(string.format(
354         [[\node[defStyle,text width=-%fem+%f%s/%d, text depth=%fcm-%fem, text height=
355         2*red, -- text width
356         e.scale_width, -- text width
357         textwidth,
358         #days, -- text width
359         length*(e.to-e.from)/(max-min), -- text depth
360         2*red+red_y, -- text depth
361         red_y, -- text height
362         e.tikz, -- free tikz code
363         (d+e.offset)/#days, -- xcoord
364         textwidth,
365         minuteToFrac(e.from,min,max)*-length, -- ycoord
366         e.content -- content
367       )
368     )
369   end
370 end
371 tex.print([[\\end{tikzpicture}]]))
372 end

```

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

```

373 function search_array(t, s)
374   for k,v in ipairs(t) do
375     if(v == s) then return k end
376   end
377   return nil
378 end
379

```

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

```

380 function minuteToFrac(minute, min, max)
381     return (minute-min)/(max-min)
382 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)

```

383 function prepareMinMax(min, max)
384     local minH = math.floor(min/60)
385     local maxH = math.ceil(max/60)
386     local min = minH*60
387     local max = maxH*60
388     return min, minH, max, maxH
389 end

```

checkKeys Checks if all ks are present in table t

```

390 function checkKeys(t, k)
391     for _,x in ipairs(k) do
392         if(t[x] == nil) then
393             return false
394         end
395     end
396     return true
397 end

```

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

```

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

```

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

```

412 function prepareDays(days)
413     local ret = {}
414     for m in days:gmatch("[^,]+") do
415         table.insert(ret, m)
416     end
417     return ret
418 end

```

copyArray Returns a copy of the table **obj**

```

419
420 function copy_array(obj)
421     if type(obj) ~= 'table' then return obj end
422     local res = {}
423     for k, v in pairs(obj) do
424         local c = copy_array(v)
425         res[copy_array(k)] = c
426     end
427     return res
428 end

```

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

```

429
430 semesterplannerLua = {
431     init = init,
432     addEvent = addEvent,
433     draw = draw
434 }
435 return semesterplannerLua
436 </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

```

437 <luaApp>
438 package.path='/usr/share/lua/5.3/?.lua;/usr/share/lua/5.3/?/init.lua;/usr/lib/lua/5.3/?.lua;/usr/lib/lua/5.3/?/init.lua'
439 package.cpath='/usr/lib/lua/5.3/?.so;/usr/lib/lua/5.3/loadall.so;./?.so;/home/lukas/.luarocks/?.so'
440
441 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)

```

442 function init(clear)
443     -- clean up first
444     -- global variable
445     if clear then
446         EVENTS = {}
447     end
448 end
449
450 text = {
451     print = function(s)
452         -- print("\n" .. s .. "\n")
453         tex.print(s)
454     end
455 }
456
457 function genDot(opts)
458     dot = ""
459     if opts.draw then
460         dot = string.format([[\\tikz[baseline=(X.base)\\node (X) [fill opacity=.5,fill=red,circle,inner sep=0pt] {\\draw (X) circle (0.5pt);}\\draw (X) circle (0.5pt);}]] .. s .. "\n")
461     end
462     return dot
463 end
464

```

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

```

465 function addEvent(opts)
466     opts.inputlineno = tex.inputlineno
467     print(string.format("collecting from line %d", opts.inputlineno))
468     if opts.draw then
469         assert(opts.date ~= nil and opts.tikz ~= nil, "date and tikz has to be given")
470         if opts.endDate == nil or opts.endDate == '' then
471             table.insert(EVENTS, {shift=opts.shift,date=dateLib(opts.date), tikz=opts.tikz, p=opts.p})
472         else
473             table.insert(EVENTS, {shift=opts.shift,date=dateLib(opts.date), tikz=opts.tikz, p=opts.p})
474         end
475     end
476 end
477

```

```

478 function addAppointment(opts)
479     addEvent(opts)
480     dot = genDot(opts)
481     if opts.print then
482         tex.sprint(string.format([[\\textit{%s} & %s & %s%s & %s & %s & %s\\]], opts.date, opts.time, dot, opts.title, opts.location))
483     else
484         tex.sprint("%")
485     end
486 end
487
488 function addExam(opts)
489     addEvent(opts)
490     dot = genDot(opts)
491     if opts.print then
492         tex.sprint(string.format([[\\textit{%s} & %s & %s%s & %s & %s \\]], opts.date, opts.time, dot, opts.title))
493     else
494         tex.sprint("%")
495     end
496 end
497
498 function addDeadline(opts)
499     addEvent(opts)
500     dot = genDot(opts)
501     if opts.print then
502         tex.sprint(string.format([[\\textit{%s} & %s%s & %s & %s \\]], opts.date, dot, opts.title, opts.location))
503     else
504         tex.sprint("%")
505     end
506 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)

```

507
508 function drawCalendar(minDate, maxDate, cols)
509     minDate = dateLib(minDate)
510     maxDate = dateLib(maxDate)
511     text.print([[\\begin{tikzpicture}[every calendar/.style={day headings=red!50,day letter headings=red!50}, every month/.style={yshift=3ex}}] ]])
512     text.print([[\\matrix[column sep=1em, row sep=1em]{
513         local i = 1
514         running = true
515         while running do
516             -- derive end from start, then check if maxDate is reached
517             endDate = minDate:copy():addmonths(1):setday(1):adddays(-1)
518             if endDate >= maxDate then
519                 endDate = maxDate
520                 running = false
521             end
522             text.print(string.format(
523                 [[\\calendar (%04d-%02d) [dates=%04d-%02d-%02d to %04d-%02d-%02d] if (Sunday) [red!50]
\\month-\\day) [nodes={inner sep=.25em,rectangle,line width=1pt,draw}] if (at least=\\year-\\month-\\day) {} else [nodes={strike out, draw}]; ]],
                    minDate:getyear(), minDate:getmonth(), minDate:getyear(), minDate:getmonth(),
                    minDate:addmonths(1)
                    minDate:setday(1)
                )
524
525
526
527
528
529             if i % cols == 0 or not running then
530                 text.print([[\\]])
531             else
532                 text.print([[&]])
533             end

```

```

534         i = i + 1
535     end
536     text.print([[ ]; ])
537
Draw highlighting on a background layer so that the calendar is not overdrawn
538     local usedDates = {}
539     text.print([[ \begin{scope}[on background layer] ]])
540     for i,ele in ipairs(EVENTS) do
541         print(string.format("Drawing item from line %d", ele.inputlineno))
542         while ele.date <= maxDate and (ele.endDate == nil or ele.date <= ele.endDate) do
543             local xshift = 0
544             if ele.shift then
545                 if usedDates[tostring(ele.date)] ~= nil then
546                     xshift = math.ceil(usedDates[tostring(ele.date)] / 2)
547                     if usedDates[tostring(ele.date)] % 2 == 0 then
548                         xshift = -xshift
549                     end
550                     usedDates[tostring(ele.date)] = usedDates[tostring(ele.date)] + 1
551                 else
552                     usedDates[tostring(ele.date)] = 1
553                 end
554             end
555             text.print(string.format([[ \node[xshift=%d mm, fill opacity=.5,fill=red,circle
%02d-%04d-%02d-%02d] { };]],
xshift, ele.tikz, ele.date:getyear(), ele.date:getmonth(), ele.date:getyear()
556             if ele.period == nil then break end
557             ele.date:addDays(ele.period)
558         end
559     end
560 end
561 text.print([[ \end{scope} ]])
562 text.print([[ \end{tikzpicture} ]])
563 end

```

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

```

564
565 semesterplannerLuaCal = {
566     init = init,
567     addAppointment = addAppointment,
568     addDeadline = addDeadline,
569     addExam = addExam,
570     drawCalendar = drawCalendar,
571 }
572 return semesterplannerLuaCal
573 \luaApp

```

3 Change History

v1.00

General: First public release 1

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.

Symbols	A	B
	<code>\addEvent</code> 265 , 465	
<code>%</code> 511	<code>\appointment</code> 147	<code>\BBB</code> 25

C		\lecture 2, <u>117</u>	\semesterplannerLua@event 98, 118, 122, 126, 130, 134
\calendar 523			
\checkKeys <u>390</u>	M	\matrix 512	\seminar 2, <u>121</u>
\copyArray <u>419</u>	\meeting 2, <u>129</u>		
D		\minuteToFrac <u>380</u>	T
\d 81, 82	\month 523		\tba 27
\day 523		O	\tbd 26
\deadline 198		\officehour 2, <u>133</u>	\teams 22
\draw 289, 313, 322, 336	\oral 15		\textit 482, 492, 502
\drawCalendar <u>507</u>		P	\tikz 460
\dur2Int <u>398</u>		\pgf@xa 79, 82, 83	\tikz@lib@cal@xshift . 79
E		\pgf@ya 78, 80, 84	\tikz@lib@cal@yshift . 78
environments:		\pgfmathsetlength . 78, 79	\tikzoption 74
timetable 1, <u>90</u>		\pgftransformxshift . . 83	\tikzset 303
\exam 173		\pgftransformyshift 80, 84	\tikzstyle 74, 75, 76
F		\phantom 460	timetable (environment) 1, <u>90</u>
\faBold 25		\phigh 18	\tiny 303
\faCalendar 164		\plow 20	\tutorial 2, <u>125</u>
\faClockO 91		\pmandatory 17	W
\faComment 15		\pmid 19	\week 306, 311
\faPencil 16		\pnone 21	\written 16
\faStickyNoteO . . 189, 212		\prepareDays 412	
\foreach 81, 306		\prepareMinMax <u>383</u>	X
I		\printSpCalendar <u>143</u> , <u>143</u>	\x 306, 313
\ifdate 77		S	Y
\ignorespaces		\searchArray <u>373</u>	\y 306, 311
. . . 119, 123, 127,		\section . 91, 164, 189, 212	\year 523
131, 135, 162, 187, 210		\semesterplanner@event <u>98</u>	\youtube 24
\init <u>220</u> , <u>442</u>		\semesterplannerLua@encircle 8, 17, 18, 19, 20, 21, 22, 23, 24, 25	Z
L			\zoom 23
\l 81, 85			