semesterplannerLua — Semesterplanner package in lua with tikz only

Lukas Heindl
oss.heindl+latex@protonmail.com
: https://github.com/atticus-sullivan/semesterplannerLua

v1.1.2 from 2024/05/19

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 of making 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

Load with `\usepackage{semesterplannerlua}`

1.1 timetable

`timetable` (*env.*) `\begin{timetable}[opts]...\end{timetable}`
This is the core environment of this package. Within it you can use `\lecture`, `\seminar`, `\tutorial`, `\officehour`, `\laboratory`, `\fieldstudy` and `\meeting`. All these commands are only defined inside the `timetable` environment, and have the same structure. It typesets a timetable with the specified entries. If you have a look at the `.sty`

file you'll see that these macros just take a general macro and set some default values before. Thus it is easy to define new ones (keep in mind that you'll need `\makeatletter`) to access the general macro.

Writes out a file containing the data from the timetable (your events) to be able to have just the `.tex` file containing the timetable, but do some scriping on the data (e.g. show the last, current and next events). For simplicity the data is stored in lua-syntax (`return`s a table of the events). This way we don't need to worry about any string escaping (like with csv) and no complicated parsers are needed. Be aware though that a simple `events = dofile"path"` has risks since `"path"` might contain lua code which is simply executed that way.

`opts` are optional arguments (enclosed with `{}`) separated with `,`:

`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,Tue,Wend,Thur,Fri`

`dayse` List of strings used in the code to idenfity the day. This list is kind of a mapping string → integer.

`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*

```
   \lecture    \lecture   [opts]
  \tutorial    \tutorial  [opts]
   \seminar    \seminar   [opts]
\officehour    \officehour[opts]
   \meeting    \meeting   [opts]
\fieldstudy    \fieldstudy[opts]
\laboratory    \laboratory[opts]
```

`opts` are optional arguments (enclosed with `{}`) separated with `,`:

`title` Give the name of the lecture

`speaker` Give the name of the lecturer

`location` Give the place of the event (most probably the room or an online plattform, see 1.3). If you want to use `\href{url}{string repr}`, keep in mind that you need to `\unescaped{}` it, since all input to lua has to be text

`day` The weekday on which the event takes place. Has to be one of those you specified in `dayse` (or by default: `M`,`T`,`W`,`Th`,`F`)

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

`prio` The priority of the event (see 1.3)

`scale width` Specify the width of the entry in fractions of column (use `0.5` to span half the column).

`offset` Shift the entry to the right. Specified in fractions of column.

`password` Original purpose: Only included in the file written out with the timetable data, to be able to make a script which quickly copies url (from `location`) and `password` to the clipboard.

`type` Only included in the file written out with the timetable data as well (Original purpose: To be able to exclude some types)

`tikz` 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`)

`content` Is usually set automatically based on the other keys (`title`for instance). This key can override this.

`textcolor` Usually set by the type of macro used

`formatter` internal stub for providing different formatting options. Formatting functions need to be specified in the `-timetable.lua` file

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.

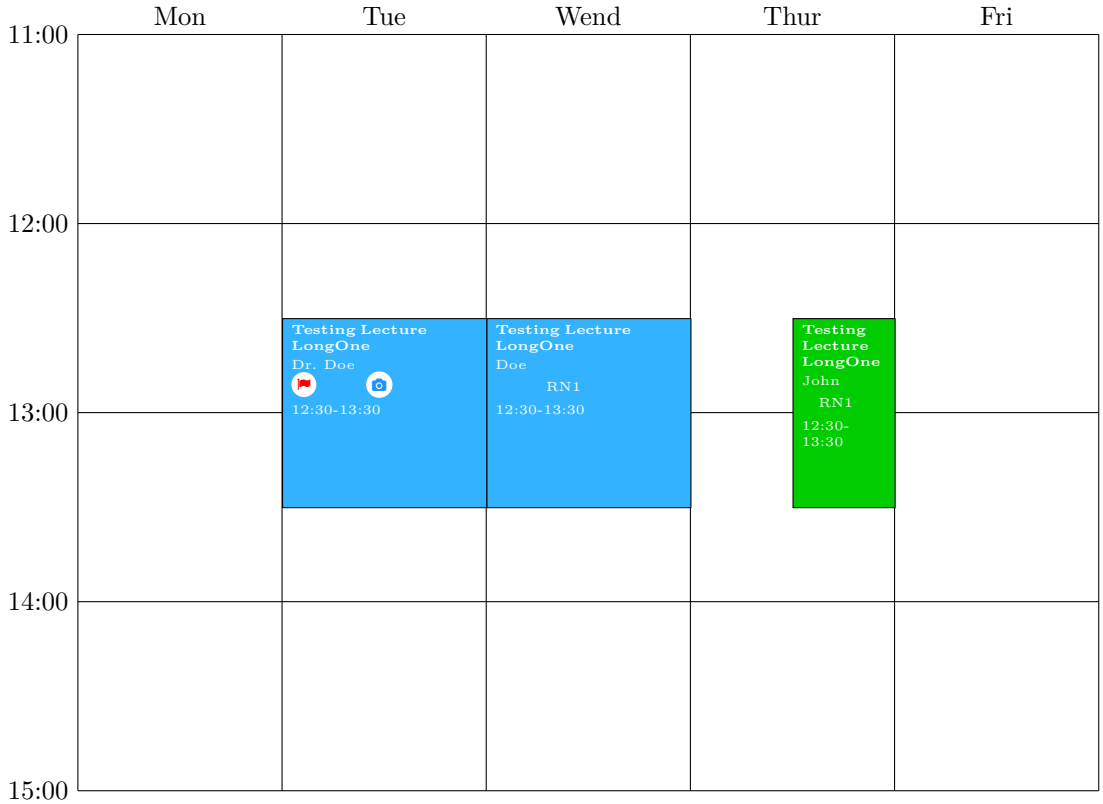
Same goes for the `width` parameter regarding the labels containing the time on the right. Since in this case any tex-lenght is allowed, you can simply try to subtract the length of the clock label using something like `\settowidth{\length}{12:30}` to set a length to the length of a clock label and then 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`,`\fieldstudy`,`\laborato` 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,Tue,Wend,Thur,Fri},
        start time/.evaluated={11*60}, end time/.evaluated={15*60}
    ]
    \lecture[title={Testing Lecture LongOne},speaker={Doe},
        location={RN1},day={W},time={12:30-13:30}]
    \tutorial[title={Testing Lecture LongOne},speaker={John},
        location={RN1},day={Th},time={12:30-13:30},offset=0.5,scale width=0.5]
    \lecture[title={Testing Lecture LongOne},speaker={Dr. Doe},
        location={\zoom},day={T},time={12:30-13:30},prio={\phigh}]
\end{timetable}
```

Timetable



1.2 Calendar related things

Now some environments and commands come which in generally are being used to typeset a table with the detailed information, but which also store the gathered data internally. This saved data can be used to output a calendar where the events (from the saved data) are highlighted with a mark (already passed days are crossed and the current day is highlighted as well).

The different environments described below are similar in general, but differ in detail (mostly the list of argument that are passed and printed in the table).

Example

```
\begin{appointments}{true}
    \appointment[print=true,shift=false,tikz={fill=black,rectangle},
        date={2022-04-15}, course={Appointment with my dog},
        end={2022-04-18}, period={1}]
    \appointment[date={2022-04-14}, course={Appointment with my dog},
        room={at home}, time={12:00}, prio={\pmandatory}]
\end{appointments}
\begin{exams}{false}
    \exam[tikz={fill=yellow}, date={2022-06-20}, time={arround midday},
        course={Driving}, desc={Mid-Term}, prio={\phigh},
        room={University}, type={\oral}]
\end{exams}
\begin{deadlines}{false}
    \deadline[tikz={fill=blue}, date={2022-06-22},
        course={Submitting Driver-license form}, prio={\phigh}]
\end{deadlines}

\printSpCalendar[2]{2022-04-01}{2022-07-31}
```

Appointments

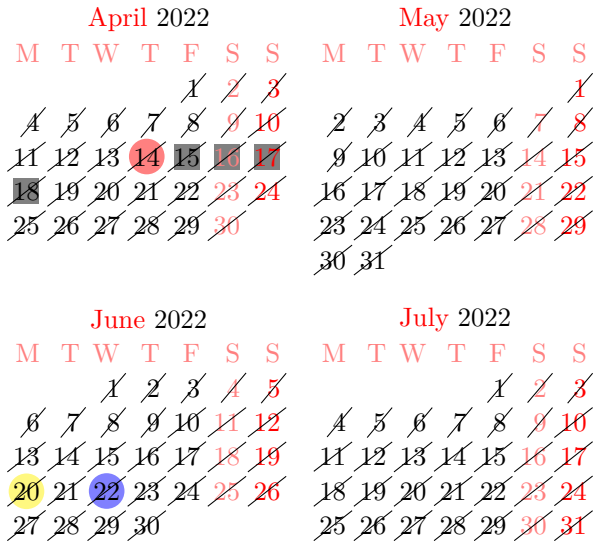
Date	Time	Course	Description	Room	Prio.
2022-04-15		Appointment with my dog			
2022-04-14	12:00	Appointment with my dog		at home	

Exams

Date	Time	Course	Type	Note
2022-06-20	arround midday	Driving		Mid-Term

Deadlines

Date	Course	Description	Prio
2022-06-22	Submitting Driver-license form		



`\printSpCalendar` `\printSpCalendar[cols]{start_date}{end_date}`

This is the macro to output such a calendar populated with the gathered events.
Use the `cols` argument to specify the amount of columns being used (months set side by side).
`start_date` and `end_date` both have to be specified in the `YYYY-MM-DD` format.

1.2.1 Appointments

`appointments` (*env.*) `\begin{appointments}[Room]{true/false}...\end{appointments}`
Environment to typeset a table of appointments. Stores the data internally to be able to typeset a calendar (with `\printSpCalendar`) with a mark for each event.
The optional argument is used as header for the `Room` column (maybe for some appointments another title is more suitable).
The mandatory argument which is `true/false` decides if the internally stored data used to typeset the calendar is being erased before this environment.

`\appointment` `\appointment[opts]`
`opts` are optional arguments (enclosed with `{}`) separated with `,`:

`date` Date of the appointment formatted as `YYYY-MM-DD`

`course` Name of the course

`room` Room (only shown in the table)

`time` Time of the event (free text independant of any formatting, only shown in the table)

`prio` priority of the event (shown in the table) (see 1.3)

`desc` Description of the event (shown in the table)

`end` upper boundary when the event should stop to repeat (with `period`). Has to be set to make event periodic/repeating.

`period` Set the period of the repeating event.

`draw true/false` wether to draw a mark in the calendar

`print true/false` wether to display te entry in the appointments table.

`shift true/false` wether to shift the mark in the calendar if there is already another event on that date (to be able to see both marks)

`tikz` Free customizable tikz code used for drawing the mark in the calendar. See the documentation at the end for keys that can be used here (all keys in `/cal`). To simply pass arguments to the tikz-node that is being created for the event use `tikz/.append={your arguments}`
Use e.g. to set the color of the mark via `tikz/.append={green}`

1.2.2 Exams

`exams` (*env.*) `\begin{exams}{true/false}...\end{exams}`
Environment to typeset a table of exams. Stores the data internally to be able to typeset a calendar (with `\printSpCalendar`) with a mark for each event.
The mandatory argument which is `true/false` decides if the internally stored data used to typeset the calendar is being erased before this environment.

`\exam` `\exam[opts]`
`opts` are optional arguments (enclosed with `{}`) separated with `,`:

`date` Date of the exam formatted as `YYYY-MM-DD`

`course` Name of the course

`room` Room (only shown in the table)

`type` Type of the exam (see 1.3) (intended: written/oral)

`time` Time of the event (free text independant of any formatting, only shown in the table)

`desc` Description of the event (shown in the table)

`draw true/false` wether to draw a mark in the calendar

`print true/false` wether to display te entry in the exams table.

`shift true/false` wether to shift the mark in the calendar if there is already another event on that date (to be able to see both marks)

`tikz` Free customizable tikz code used for drawing the mark in the calendar. See the documentation at the end for keys that can be used here (all keys in `/cal`). To simply pass arguments to the tikz-node that is being created for the event use `tikz/.append={your arguments}`
Use e.g. to set the color of the mark via `tikz/.append={green}`

1.2.3 Deadlines

`deadlines` (*env.*) `\begin{deadlines}{true/false}...\end{deadlines}`
Environment to typeset a table of deadlines. Stores the data internally to be able to typeset a calendar (with `\printSpCalendar`) with a mark for each event.
The mandatory argument which is `true/false` decides if the internally stored data used to typeset the calendar is being erased before this environment.

`\deadline` `\deadline[opts]`
`opts` are optional arguments (enclosed with `{}`) separated with `,`:

`date` Date of the deadline formatted as `YYYY-MM-DD`

`course` Name of the course

`prio` priority of the event (shown in the table) (see 1.3)

`desc` Description of the event (shown in the table)

`draw true/false` wether to draw a mark in the calendar

`print true/false` wether to display te entry in the deadlines table.

`shift true/false` wether to shift the mark in the calendar if there is already another event on that date (to be able to see both marks)

`tikz` Free customizable tikz code used for drawing the mark in the calendar. See the documentation at the end for keys that can be used here (all keys in `/cal`). To simply pass arguments to the tikz-node that is being created for the event use `tikz/.append={your arguments}`
Use e.g. to set the color of the mark via `tikz/.append={green}`

1.3 Icons

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

`\zoom`		`\teams`	
`\BBB`		`\youtube`	
`\pmandatory`		`\phigh`	
`\pmid`		`\plow`	
`\pnone`			
`\tbd`		`\tba`	

2 Implementation

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

2.1 semesterplannerlua.sty

2.1.1 Global Stuff

```
1 ⟨*package⟩
2 \RequirePackage{tikz}
3 \usetikzlibrary{calendar, positioning, shapes.misc, backgrounds}
4 \RequirePackage{pgfkeys}
5 \RequirePackage{xcolor}
6 \RequirePackage{fontawesome}
7 \RequirePackage{luapackageloader} % use the default lua path as well
```

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

```
8 \definecolor{seminar}{rgb}{1.0, 0.8, 0.0}
9 \definecolor{lecture}{rgb}{0.2, 0.7, 1.0}
10 \definecolor{tutorial}{rgb}{0.0, 0.8, 0.0}
11 \definecolor{meeting}{rgb}{0.8, 0.0, 0.0}
12 \definecolor{laboratory}{rgb}{0.8, 0.0, 0.0}
13 \definecolor{fieldstudy}{rgb}{0.8, 0.0, 0.0}
14 \definecolor{officehour}{rgb}{0.0, 0.4, 0.6}
15 \definecolor{DodgerBlue}{HTML}{1E90FF}
```

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

```
16     \newcommand*{\semesterplannerLua@encircle}[1]{
17         \begin{minipage}[b][1em][c]{1.5em}
18             \begin{tikzpicture}
19                 \node[fill,circle,inner sep=1pt, color = white] {#1};
20             \end{tikzpicture}
21         \end{minipage}
22     }
```

Commands for exams

`\oral`

```
23 \protected\def\oral{\faComment}
```

`\written`

```
24 \protected\def\written{\faPencil}
```

Commands for symbols of priority

`\pmandatory`

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

`\phigh`

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

`\pmid`

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

`\plow`

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

`\pnone`

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

Commands for online platforms.

\teams

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

\zoom

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

\youtube

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

\BBB

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

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

\tbd

```
\protected\def\tbd{\faQuestion}
```

\tba

```
\protected\def\tba{\faBullhorn}
```

Load the lua modules

```
\directlua{sp = require("semesterplannerLua_timetable.lua")}
\directlua{cal = require("semesterplannerLua_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

```
\pgfkeys{
```

`/semesterplannerLua` 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.

```
        /semesterplannerLua/timetable/env/.cd,
        days/.initial={Mon,Tue,Wend,Thur,Fri}, days/.default={Mon,Tue,Wend,Thur,Fri},
        dayse/.initial={M,T,W,Th,F}, dayse/.default={M,T,W,Th,F},
        %
        start time/.initial=, start time/.default=,
        end time/.initial=, end time/.default=,
        %
        width/.initial=\textwidth, width/.default=\textwidth,
        length/.initial=10, length/.default=10,
        %
```

`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

```
49        /semesterplannerLua/timetable/event/.cd,
50        % event arguments
51        content/.initial=, content/.default=,
52        %
53        time/.initial=, time/.default=,
54        day/.initial=, day/.default=,
55        %
56        tikz/.initial=, tikz/.default=,
57        scale width/.initial=1, scale width/.default=1,
58        offset/.initial=0, offset/.default=0,
59        %
60        textcolor/.initial=, textcolor/.default=,
61        title/.initial=, title/.default=,
62        speaker/.initial=, speaker/.default=,
63        location/.initial=, location/.default=,
64        password/.initial=, password/.default=,
65        prio/.initial=, prio/.default=,
66        type/.initial=, type/.default=,
67        formatter/.initial=timetableformatter, formatter/.default=timetableformatter,
68        %
```

`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

```
        /semesterplannerLua/calendar/.cd,
        draw/.initial={true}, draw/.default={true},
        room/.initial={}, room/.default={},
        time/.initial={}, time/.default={},
        prio/.initial={}, prio/.default={},
        course/.initial={}, course/.default={},
        desc/.initial={}, desc/.default={},
        type/.initial={}, type/.default={},
        date/.initial={}, date/.default={},
        end/.initial={}, end/.default={},
        tikz/.initial={}, tikz/.default={},
        period/.initial={nil}, period/.default={nil},
        shift/.initial={true}, shift/.default={true},
        print/.initial={true}, print/.default={true},
    }
```

2.2 Tikz Calendar add weekday labels

```
\tikzoption{day headings}{\tikzstyle{day heading}=[#1]}
\tikzstyle{day heading}=[]
\tikzstyle{day letter headings}=[
    execute before day scope={ \ifdate{day of month=1}{%
      \pgfmathsetlength{\pgf@ya}{\tikz@lib@cal@yshift}%
      \pgfmathsetlength\pgf@xa{\tikz@lib@cal@xshift}%
      \pgftransformyshift{-\pgf@ya}
      \foreach \d/\l in {0/M,1/T,2/W,3/T,4/F,5/S,6/S} {
        \pgf@xa=\d\pgf@xa%
        \pgftransformxshift{\pgf@xa}%
        \pgftransformyshift{\pgf@ya}%
        \node[every day,day heading]{\l};%
      }
    }{}%
  }%
]
```

2.2.1 Local Stuff (timetable-env local)

`timetable` (*env.*) 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.

```
\newenvironment{timetable}[1][]{
    \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.

```
    \pgfkeys{/semesterplannerLua/timetable/env/.cd,days,dayse,start time,end time,width,lengt
    \directlua{sp.init{
        days=[[\pgfkeysvalueof{/semesterplannerLua/timetable/env/days}]],
        min=[[\pgfkeysvalueof{/semesterplannerLua/timetable/env/start time}]],
        max=[[\pgfkeysvalueof{/semesterplannerLua/timetable/env/end time}]],
        dayse=[[\pgfkeysvalueof{/semesterplannerLua/timetable/env/dayse}]]}}
```

`\semesterplannerLua@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)

```
    \newcommand{\semesterplannerLua@event}[1][]{
        \pgfkeys{/semesterplannerLua/timetable/event/.cd,content,time,day,tikz,scale
```

```
        width,offset,textcolor,title,speaker,type,location,password,prio,formatter, ##1}
        \directlua{
            sp.addEvent{
                time="\pgfkeysvalueof{/semesterplannerLua/timetable/event/time}",
                day="\pgfkeysvalueof{/semesterplannerLua/timetable/event/day}",
                tikz=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/tikz}]],
                offset=\pgfkeysvalueof{/semesterplannerLua/timetable/event/offset},
                scale_width=\pgfkeysvalueof{/semesterplannerLua/timetable/event/scale width},
                formatter=\pgfkeysvalueof{/semesterplannerLua/timetable/event/formatter},
                textcolor=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/textcolor}]],
                title=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/title}]],
                speaker=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/speaker}]],
                location=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/location}]],
                password=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/password}]],
                prio=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/prio}]],
                type=[[\pgfkeysvalueof{/semesterplannerLua/timetable/event/type}]],
            }
        }
    }
```

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

\lecture

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

\seminar

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

\tutorial

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

\meeting

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

\officehour

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

\laboratory

```
    \newcommand{\laboratory}[1][]{
        \semesterplannerLua@event[tikz={fill=laboratory,}, textcolor=white, type=lab, ##1]
        \ignorespaces
    }
```

\fieldstudy

```
    \newcommand{\fieldstudy}[1][]{
        \semesterplannerLua@event[tikz={fill=fieldstudy,}, textcolor=white, type=fieldstudy, #
        \ignorespaces
    }
```

```
}{
```

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

```
    \directlua{sp.draw(
        [[\pgfkeysvalueof{/semesterplannerLua/timetable/env/length}]],
        [[\pgfkeysvalueof{/semesterplannerLua/timetable/env/width}]])}
}


```

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

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

\newenvironment{appointments}[2][Room]{
    \directlua{cal.init(#2)}
    \newcommand{\appointment}[1][]{%
        \pgfkeys{/semesterplannerLua/calendar/.cd,draw,room,time,prio,course,desc,date,end,til
        \directlua{
            cal.addAppointment{draw=\pgfkeysvalueof{/semesterplannerLua/calendar/draw},
            room=[[\pgfkeysvalueof{/semesterplannerLua/calendar/room}]],
            time=[[\pgfkeysvalueof{/semesterplannerLua/calendar/time}]],
            prio=[[\pgfkeysvalueof{/semesterplannerLua/calendar/prio}]],
            course=[[\pgfkeysvalueof{/semesterplannerLua/calendar/course}]],
            desc=[[\pgfkeysvalueof{/semesterplannerLua/calendar/desc}]],
            date=[[\pgfkeysvalueof{/semesterplannerLua/calendar/date}]],
            endDate=[[\pgfkeysvalueof{/semesterplannerLua/calendar/end}]],
            tikz=[[\pgfkeysvalueof{/semesterplannerLua/calendar/tikz}]],
            period=\pgfkeysvalueof{/semesterplannerLua/calendar/period},
            shift=\pgfkeysvalueof{/semesterplannerLua/calendar/shift},
            print=\pgfkeysvalueof{/semesterplannerLua/calendar/print}}}%
        \ignorespaces
        }
    \section*{\faCalendar~Appointments}
    \begin{tabular}{rllll}
        \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Description}&\textbf{#1}&\textbf{
}{
    \end{tabular}
}

\newenvironment{exams}[1]{
    \directlua{cal.init(#1)}
    \newcommand{\exam}[1][]{%
        \pgfkeys{/semesterplannerLua/calendar/.cd,draw,room,time,prio,course,desc,date,end,til
        \directlua{
            cal.addExam{
                draw=\pgfkeysvalueof{/semesterplannerLua/calendar/draw},
                room=[[\pgfkeysvalueof{/semesterplannerLua/calendar/room}]],
                time=[[\pgfkeysvalueof{/semesterplannerLua/calendar/time}]],
                course=[[\pgfkeysvalueof{/semesterplannerLua/calendar/course}]],
                desc=[[\pgfkeysvalueof{/semesterplannerLua/calendar/desc}]],
                date=[[\pgfkeysvalueof{/semesterplannerLua/calendar/date}]],
                tikz=[[\pgfkeysvalueof{/semesterplannerLua/calendar/tikz}]],
                type=[[\pgfkeysvalueof{/semesterplannerLua/calendar/type}]],
                shift=\pgfkeysvalueof{/semesterplannerLua/calendar/shift},
                print=\pgfkeysvalueof{/semesterplannerLua/calendar/print}}}%
        \ignorespaces
        }
    \section*{\faStickyNoteO~Exams}
    \begin{tabular}{rllll}
        \textbf{Date}&\textbf{Time}&\textbf{Course}&\textbf{Type}&\textbf{Note}\\
}{
```

```
    \end{tabular}
}

\newenvironment{deadlines}[1]{
    \directlua{cal.init(#1)}
    \newcommand{\deadline}[1][]{%
        \pgfkeys{/semesterplannerLua/calendar/.cd,draw,room,time,prio,course,desc,date,end,til
        \directlua{
            cal.addDeadline{
                draw=\pgfkeysvalueof{/semesterplannerLua/calendar/draw},
                course=[[\pgfkeysvalueof{/semesterplannerLua/calendar/course}]],
                desc=[[\pgfkeysvalueof{/semesterplannerLua/calendar/desc}]],
                date=[[\pgfkeysvalueof{/semesterplannerLua/calendar/date}]],
                tikz=[[\pgfkeysvalueof{/semesterplannerLua/calendar/tikz}]],
                prio=[[\pgfkeysvalueof{/semesterplannerLua/calendar/prio}]],
                shift=\pgfkeysvalueof{/semesterplannerLua/calendar/shift},
                print=\pgfkeysvalueof{/semesterplannerLua/calendar/print}}}%
        \ignorespaces
        }
    \section*{\faStickyNoteO~Deadlines}
    \begin{tabular}{rlll}
        \textbf{Date}&\textbf{Course}&\textbf{Description}&\textbf{Prio}\\
}{
    \end{tabular}
}
```

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.