PocketDiary, a Personal Organizer

New Version of an Old Module

Willi Egger

The first version of this calendar module for a personal organizer was created in 2010/2011. It is intended to prepare a personal pocket planner of eight small pages arranged on an A4 sheet. Although the sheet is printed only single sided, it can be folded into a booklet. Now that we use LuaMetaTEX with ConTEXt, the code needed an overhaul because it didn't work anymore. Talks about calendars during the ConTEXt meeting 2021 were the reason for picking up this work. Beyond just making the module work again, it should also be capable of calculating moon phases and sun data. And indeed, the new version can now provide lunar days, pictograms of moon phases as well as sunrise and sunset times and the number of hours of light. – The collection of Lua functions can be considered a toolbox, and with a little bit of imagination it can be used to produce other lists where calendar information is required.

Detailed documentation is provided in the module files, including how to setup a production file. At the end of the article there are instructions on how to fold the booklet.

The source files are published in the ConTEXt modules directory[1].

1. Introduction

Since I use ConTEXt, I am involved in establishing imposition schemes that I can use for my personal book projects. One of these schemes is to place eight pages on a single-sided printed A4 sheet that can be folded into a booklet that shows only printed pages. It was originally implemented for my brother Heinz.

At the time of the first version of this module (2010/2011), I was receiving DANTE's "TEXnische Komödie" as the secretary of the NTG. Uwe Ziegenhagen published an article in no. 3/2010 [5] about the preparation of a 'PocketMod' in LaTEX, which is a personal organizer based on the aforementioned arranging scheme. PocketMod is available online at *https://www.pocketmod.com*. After reading the article and visiting

[1] *https://modules.contextgarden.net/*

the website, I decided to give it a try to make such a personal planner using ConTEXt and LuaTEX.

During the ConTEXt meeting 2021, Tomáš Hála presented the production of planners; he needed a new diary during the Covid-19 pandemic while all shops were closed. In the discussion following his presentation, someone mentioned that it would be nice to also have information on the moon phases and time of sunrise/sunset in a personal planner.

This was the moment for remembering that I already had a diary program in which I could try to implement moon cycles and sun data. When I tried to run the old module it no longer worked, which is not surprising given that a lot has changed in the ConTEXt environment over these ten years.

Anyway, after adjusting the module for LuaMetaTEX, I was able to begin building the new functionality.

The calculations related to the moon cycle were not a big problem, because there are fairly simple algorithms available for the calculation of lunar days. Normally, astronomical calculations quickly become complicated, and if you use a simplified approach you have to accept that the results might just be an educated guess. Now, the formulae used for the moon data have an accuracy of about one day, and this is acceptable for our purpose.

The situation is a little different for calculating sun data. Although there is a substantial number of publications on the internet concerning these astronomical calculations, it appears that the translation of the algorithms provided is not entirely trivial. While the information provided is probably correct, for the layman it would seem that more background info is needed for the calculations to result in the required data. I finally discovered the code for an on-screen display of the sun data, already written in Lua. This is now adapted for the current version of the PocketDiary module.

2. General Description

The PocketDiary is a personal organizer. It is like a section of a book, but a very small one. The PocketDiary is based on the idea that it should be possible to repeatedly produce such a personal organizer during the year by altering a minimal number of variables, based on a week number and, of course, the year. Although this information is sufficient for a week-based diary, it is not enough if you want to include more options like month tables, a year table, personal data etc.

The module contains a number of different files. On top there is the user file `PocketDiary.tex`. The module itself is based on `t-pocketdiary.tex`; this file loads a couple of Lua files. The module is setup as a CLD (ConTEXt Lua Document). The Lua file `t-pocketdiary.lua` contains the cld elements. `t-calendar.lua` contains calendar calculations. In the subdirectory `Moonphase` you will find the Lua file `t-moonphase.lua` for the calculations of lunar data and `moon-MP.tex` that contains the MetaFun pictograms for the moon phases. In the subdirectory `Solar` there is the file `t-solar.lua` for the sun data calculations, and `sun-MP.tex` contains the MetaFun symbols for sunrise, sunset and daylight hours.

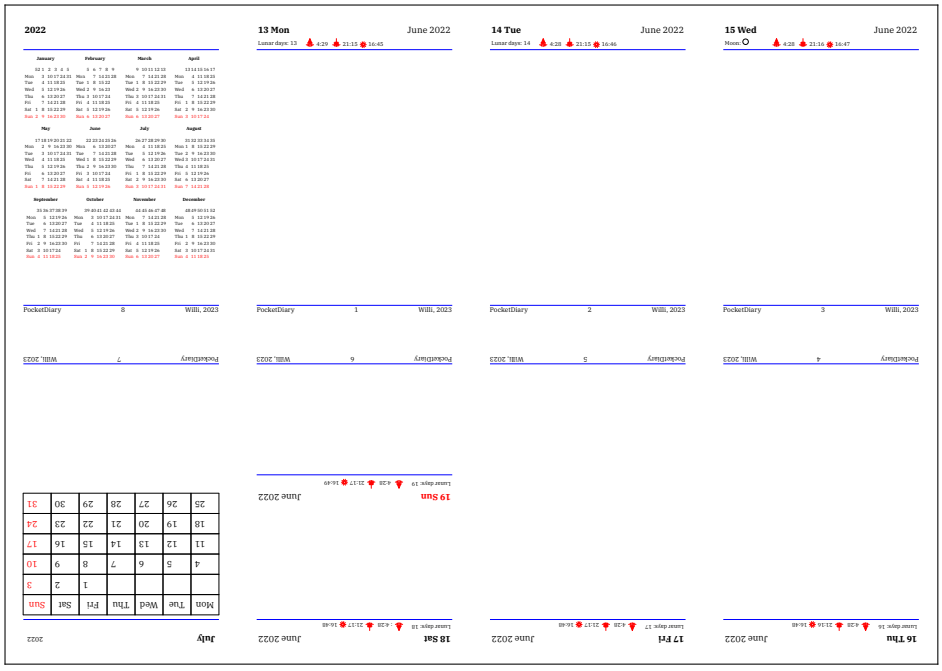


Figure 1. Example: Dayplan Calendar with Month and Year Calendar

Figure 2. Example: Week Calendar, Month and Next Month Calendar

3. PocketDiary Layout

The following few lines of code are all you need in your personal PocketDiary file to produce a personal organizer.

First we tell ConTEXt to load the module `t-pocketdiary`:

```
\usemodule[pocketdiary]
```

By default, PocketDiary uses TEX Gyre Pagella. If you have your own favorite font, set it up here:

```
\setupbodyfont[dejavu,ss,8pt]
```

Keep in mind that the resulting pages for the organizer are fairly small. So the body font size should be adapted to the the properties of the selected font.

PocketDiary has a multilingual interface, so we need to tell it which language to use. So far we support English, Dutch, German, Italian, Spanish and French.

```
\mainlanguage[de]
```

The module automatically sets the layout of the PocketDiary page.

The PocketDiary's page design is based on header, footer and main text area. Depending on what kind of calendar page you generate, the header contains the calendar data, e. g. on the one day page the header is *Day of the month – Day name (short) – Holiday name – Week number – Year*. Additionally, it shows data concerning the moon phase and the sunrise/sunset time including daylight hours. If the generated page uses the text area for the data, the header contains information about the generated page e. g. *Week calendar – month number – week number and year*.

All configurable options are accessible in six sets of variables.

First, setup the calendar calculations in the *PocketDiary* set:

```
\setvariables
   [PocketDiary]
   [WeekDay=7,
    Week=28,
    Month=7,
    Year=2022,
    Nextyear=yes,
    Daybyday=no]
```

Variable	**Value**	**Comment**
WeekDay	number	Values between 1 and 7
Week	number	Values between 1 and 53
Month	number	Values between 1 and 12
Year	number	Year numbers in the range 1900 to 4099. The lower limit is computer dependent (OS timestamp), the upper limit depends on the Easter Sunday calculation (7).
Nextyear	yes/no	If set to 'yes', the next year instead of the current year is used for the calculation of the year calendar.

Daybyday	yes/no	If set to 'yes', the weekdays are typeset with one page per day and the weekend with Saturday and Sunday on one page. This uses six pages, and we can freely choose the content of pages 7 and 8.

You can define your own individual layout in the variable set *PocketDiaryLayout* for the eight pages of the PocketDiary (Page1 to Page8). It is no problem to use the same page layout for several pages.

```
\setvariables
   [PocketDiaryLayout]
   [Page1=Lost-Return,
    Page2=Weekcurrentplan,
    Page3=Dayplan,
    Page4=Monthcurrentplan,
    Page5=Blank,
    Page6=Contact,
    Page7=Caro,
    Page8=Lines]
```

Variable	**Page content**
Dayplan	The weekday indicated by the variable 'WeekDay' from the previous section is used to make a PocketDiary page. See figure 3a.
Weekendplan	The weekend of the selected week (variable 'Week') is typeset on a single page. See figure 3b.
Weekcurrentplan	A tabular week calendar of the selected week (variable 'Week'). See figure 4a
Weeknextplan	A tabular week calendar of next week ('Week' + 1).
Monthcurrentplan	A month table based on the variable 'Month'. See figure 4b
Monthnextplan	A month table of the next month ('Month' + 1).
Yearcurrentplan	A complete year calendar of the year selected by variable 'Year'. See figure 5
Yearnextplan	If the 'Nextyear' variable in the previous section is 'yes', then a complete year calendar for the next year indicated in the variable 'Year' of the previous section is typeset.
Lost-Returnto	A page with the indicated personal information from the *PocketDiaryAddress* variable set is used to compose a lost and return page (see next section). Example page shown in figure 6
Blank	This page has a header and footer but is otherwise empty.
Todo	A to-do-list template. See figure 6
Caro	A page of square-ruled paper. See figure 6
Lines	A page of ruled paper. See figure 6
Contact	A form with two sets of preprinted fields for marking down contact information. See figure 6

The third block of variables *PocketDiaryAddress* contains information used for the footer and the lost-return form:

```
\setvariables
   [PocketDiaryAddress]
   [Familyname=Egger,
    Forename=Willi,
    Street=Townstreet 3B,
    Zipcode=5000,
    City=New Beach,
    Country=TEX-world,
    Phone=+22 444 55 88 66,
    Mobile=+22 6 19 19 17 17,
    E-mail=info at pocketdiary.org,
    Web=www.pocketdiary.org]
```

The footer line contains three fields that you can customize with the *PocketDiaryFooter* variables:

```
\setvariables
   [PocketDiaryFooter]
   [Lefttext=PocketDiary,
    Centertext=\pagenumber,
    Righttext={\getvariable{PocketDiaryAddress}
     {Forename},~{\currentdate[year]}}]
```

To calculate the sun data, we have to fill in another set of variables, *PocketDiaryGeoPosition*:

```
\setvariables
  [PocketDiaryGeoPosition]
  [lat=52.467860,
   lon=16.981240,
   timeoffset=1,
   continent=EU]
```

Variable	**Meaning**
lat	The latitude of the place you want to calculate sun data for
lon	The longitude of the place you want to calculate sun data for
timeoffset	Time offset from UTC in hours
continent	Valid entries are EU and US. When no DST should be calculated then this variable remains empty.

Daylight saving time (DST) starts on different days in the USA and Europe; also, not all countries have DST.

In **Europe** DST starts on the last Sunday of March and ends on the last Sunday of October.

In the **USA** DST starts on the second Sunday in March and ends on the first Sunday in November (according to Energy Policy Act 2005). Except that a part of Arizona and Hawaii have no DST.

How DST calculation is applied depends on the value of the variable ‘continent’.

The yearly date on which DST starts and ends is calculated automatically.

For the sun data calculations you need to know the latitude (lat) and longitude (lon) of the location for which the diary is typeset.

For correct calculations you also need to pass the timezone offset.

Sun data is typeset on every ‘Dayplan’ and the days of the ‘Weekendplan’. On the week calendar this data is provided only on Tuesday.

The main text is separated from header and footer with a colored line, ‘Separatorline’; its default color is blue. The color of grid lines, ‘Gridline’, defaults to light grey. You can customize both in *PocketDiaryColors*:

```
\setvariables
  [PocketDiaryColors]
  [Separatorline=blue,
   Gridline={s=.55}]
```

The last block of code states:

```
\starttext
  \setuparranging[1*8]
  \getbuffer[PocketDiary]
\stoptext
```

The command `\setuparranging[1*8]` could also be activated in `t-pocketdiary.tex`. However it is better to keep it in the user’s PocketDiary setup file for easier debugging.

4. Available templates

Dayplan and Weekendplan

A day calendar is shown in figure 3a and a weekend calendar in figure 3b.

Week Calendar and Month Calendar

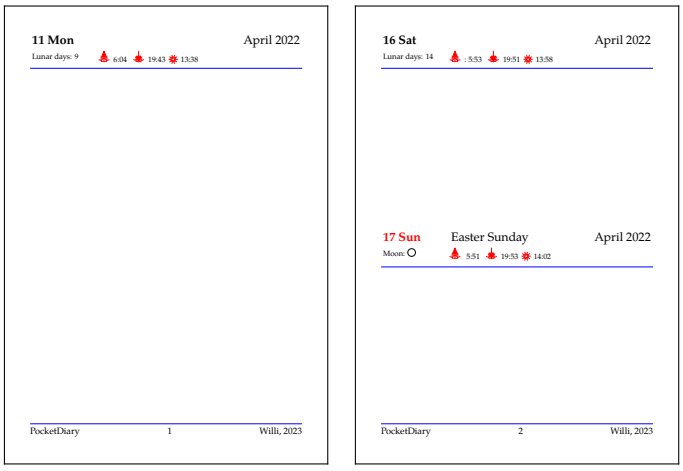
A week calendar is shown in figure 4a and a month calendar in figure 4b.

Year Calendar

An example of the current year calendar is in figure 5.

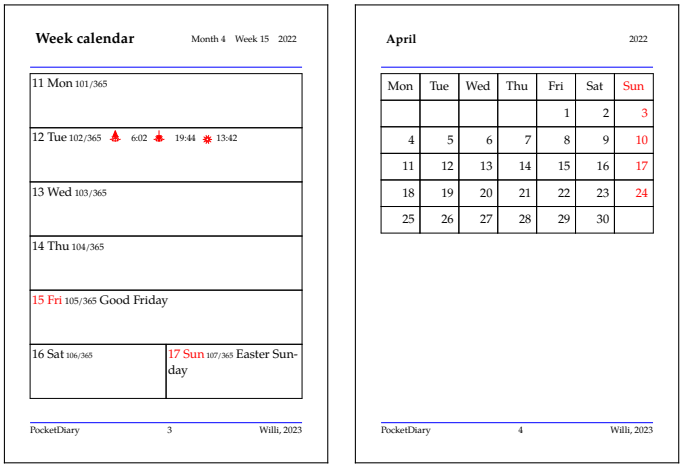
Templates Unrelated to Calendar Calculations

The PocketDiary comes with a couple of templates for notes. They are shown in figure 6.



a b

Figure 3. Dayplan and Weekendplan



a b

Figure 4. Week and Month Calendar

5. The Lua Part

Back in 2010/11, I thought it couldn't be that hard to build a calendar for this kind of personal organizer. And it's true – as long as your expectations of what the organizer will be able to do are not too high. So, when I started this, I wanted to get a production file that would be easy to maintain, and there shouldn't be too many changes during repeated cycles of making such a personal organizer. This implies however that the background machinery must be able to calculate everything based on year, week number, month and day number. I soon found out that there are more and less difficult areas in the year's calendar. The easier things comprise leap year calculations and calculations which do not involve January and December. The beginning and the end of the year are quite tricky moments. Think e. g. whether the last days

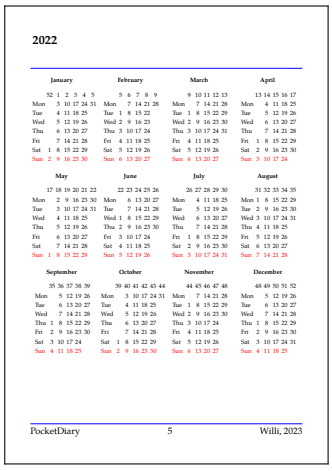


Figure 5. Year Calendar

of December are in week 52 or 53 and whether the first days of the next year belong to week 52, 53 or represent the first week of the new year.

I decided to use the European default for date calculations. The ISO 8061 standard defines that the first week of a year is the week including the year's first Thursday. Furthermore it states that a week starts on Monday, which is different from what Lua does, where day 1 is Sunday. But the `os` library of Lua can do a lot for date calculations, so it is used as far as possible.

Consulting an internet search engine helps a lot to gain insight into date calculations. Sohael Babwani's work [6] is an interesting source. He describes different methods for calculating dates that I used for the module. Since I wanted the PocketDiary to indicate the main Christian holidays as well, I found Ronald W. Mallen's website [7] to be a big help. He developed an Easter Sunday calculation based on six tables. He also presents a BASIC program (programming algorithm by Greg Mallen) on the Astronomical Society of South Australia website that calculates the date automatically. I transcribed this program to Lua.

There are more questions to be answered, e.g. how to calculate a week number from a certain date. Thanks to Ferry van Schaik [8], I got hold of a Java snippet which I have converted to Lua.

The formula for the calculation of the lunar days was also obtained from the internet, provided by SubsySTEMs [9], an American educational institution. For the PocketDiary, I decided to show the lunar days if the stage of the moon is not 'New Moon', 'Growing Moon', 'Full Moon' or 'Waning moon'. For those four moments in the lunar month I prepared pictograms in MetaFun. The rather simple calculation does not account for a few factors in the moon phase so it is not as accurate as more rigorous calculations, but it is close enough to get a good idea of the moon phase. There is also no adjustment for the time of day.

To also provide sun related data, I collected information from the internet again. As mentioned in the introduction, there is a good number of articles on sun data

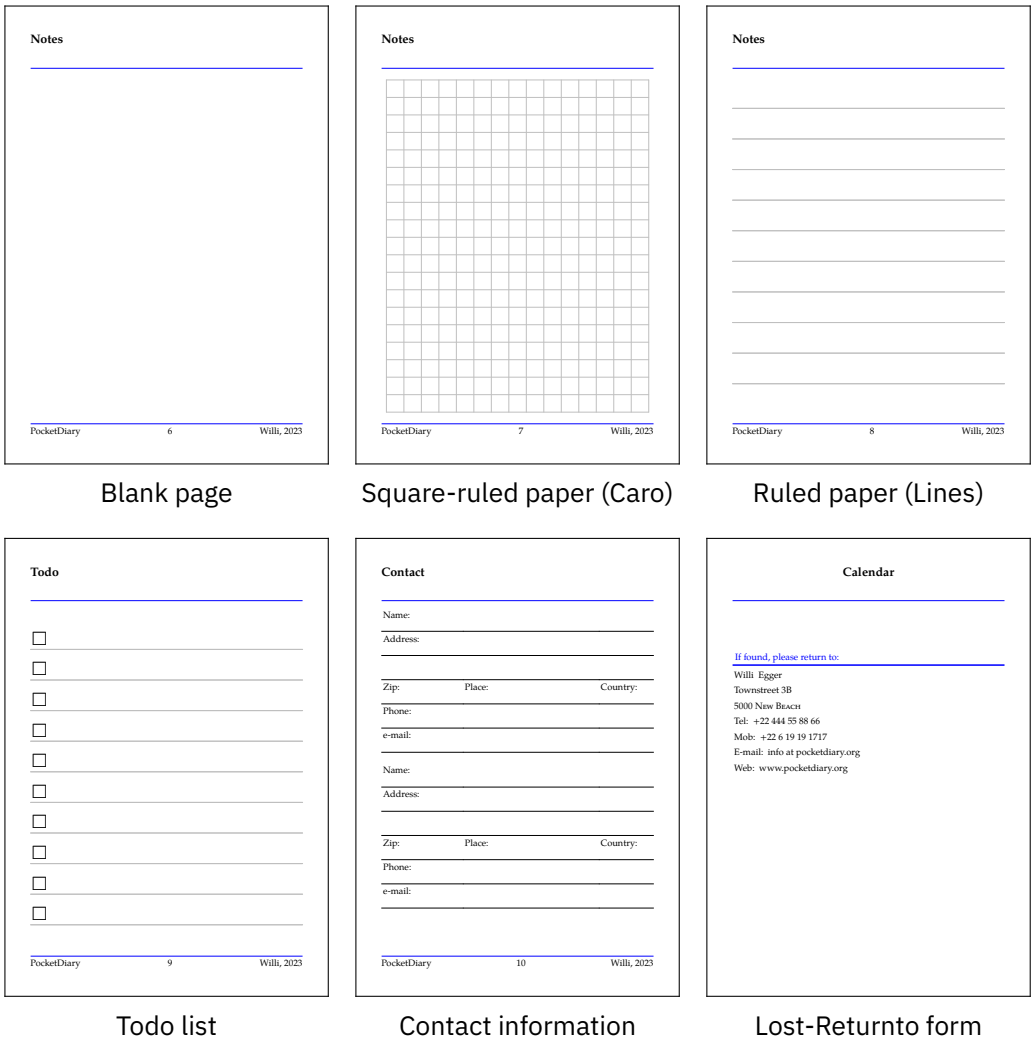


Figure 6. Note templates of PocketDiary

calculations. However I lack the knowledge to understand what these formulae do more precisely. While trying to implement different calculations, I ended up with results that never resembled something that I could convert into hours and minutes. But thanks to the work of Alexander Yakushev [10] I am now able to run sun data calculations; his code was already written in Lua. My module contains his calculations adapted to the needs of the diary. It seems that the calculations is very precise; my weather station collects weather and sun data from the Open Weather Map website (*https://openweathermap.org*), and their sunrise and sunset times vary only by a minute from the module's calculations.

6. How to Fold the PocketDiary

The eight printed pages are folded in such a way that the PocketDiary presents itself as a small booklet without visible empty pages.

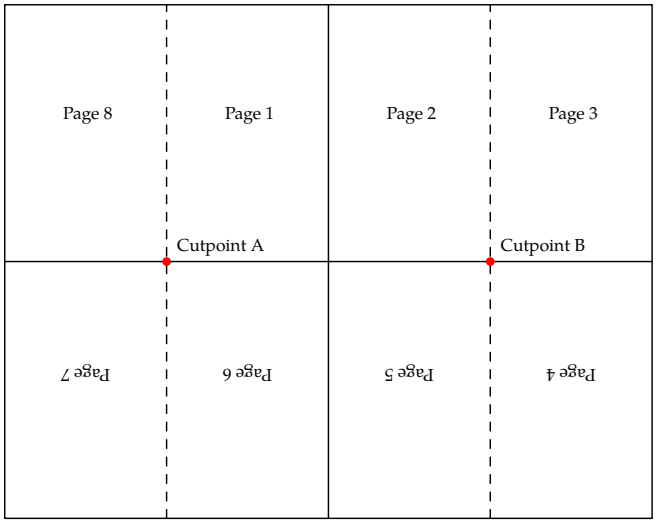


Figure 7. The basic folding scheme

Start by turning the print-side upside down. Then make two mountain-folds as indicated by the straight lines in figure 7. Unfold the paper and turn it face up and 90° to the left. Make a valley-fold with the lower part of the sheet onto the previously made mountain-fold. Unfold and rotate the sheet by 180°. Make another valley-fold as described before. Unfold the sheet.

Take a sharp knife and a ruler. Cut the paper open between cutting points A and B (see figure 7).

Now you can fold the booklet. First, fold the paper again lengthwise. Then hold the double-folded paper with the mountain-fold up. Push from both sides towards the centre in order to get a form similar to figure 8. Then fold the upper double-page in direction B, the lower double-page in direction C and finally the left-hand double-sided page in direction D.

Before creasing the booklet at the spine, it is worthwhile to put the section down on the table and adjust folds where needed. Finally, the spine is creased preferably with a bone folder. With needle and thread you might make a ‘cahier’-stitch to keep the booklet together.

7. The Module

The module is available on the ConTEXt garden and can be downloaded from *http://modules.contextgarden.net/dl/*.

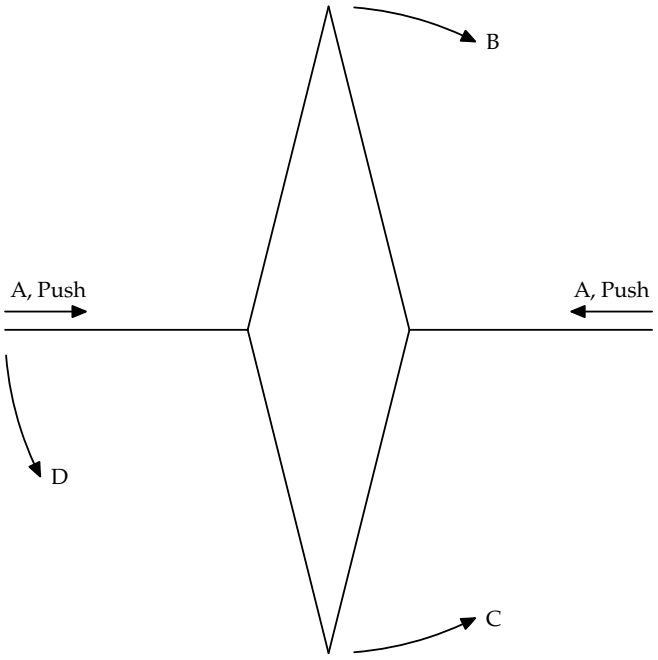


Figure 8. The basic folding scheme

8. References

1. Programming in Lua. Roberto Ierusalimschy. Lua.org, Rio de Janeiro. 2nd ed. 2006.
2. Lua 5.3 Reference Manual. Roberto Ierusalimschy, Luiz Henrique de Figueiredo, Waldemar Celes. Copyright © 2015–2020 Lua.org, PUC-Rio. *https://www.lua.org/manual/5.3/manual.html*
3. Metafun. Hans Hagen. PRAGMA Advanced Document Engineering, Hasselt NL. Copyright: 1999-2017 / version 4: March 31, 2017.
4. ConTEXt Lua documents. Hans Hagen. PRAGMA ade, Hasselt. prelim. version. July 25, 2021.
5. Pocketmods mit LaTeXerstellen. Uwe Ziegenhagen. In TEXnische Komödie 3/2010, Dante e.V, Heidelberg. pp. 27-32. 2010.
6. An extended Approach to the Julian and Gregorian Calendar, Babwani's Congruence and Babwani's Algorithm. Sohael Babwani. 2010. From: *http://www.babwani-congruence.blogspot.com/*
7. Easter Dating Method. Ronald W. Mallen. Astronomical Society of Southern Australia. 2002. From: *http://www.assa.org.au/edm.html*.
8. How can I calculate the week number of the current date? Ferry van Schaik. 2001. From: *http://www.irt.org/script/914.htm*
9. Calculate the Moon Phase. SubsySTEMs. 2017. From: *https://www.subsystems.us*
10. Module for calculating sunrise/sunset times for a given location based on algorithm by United States Naval Observatory, Washington. Alexander Yakushev. From: *https://gist.github.com/alexander-yakushev/88531e23a89a0f2acbf1*

The base for A. Yakushev's work originated from: *http://williams.best.vwh.net/sunrise_sunset_algorithm.htm*
(This website is unfortunately nowadays unavailable)

9. Thanks

I would like to thank Hans Hagen for ConTEXt with LuaMetaTEX and all his patience to help me with solving the encountered struggles. Also I would like to thank Wolfgang Schuster for supporting me in tackling the multilingual interface of the first version of this module. Last but not least I thank Hraban Ramm to push me to improve some typographical aspects and streamlining some naming issues.