Documentation

Typesetting Recipes

Jürgen Gilg

cookybooky

This package is dedicated to my little almost four year old buddy PAUL HENRY, *who likes to cook with me and is always interested in how and what we cook together (and what is even better – he as well helps to do the dishes afterwards).*

So I decided to write some macros to typeset recipes in an eye-pleasing way (with additional pictures), to archive the recipes we cooked and in future will cook and to someday bundle them together to a very personal recipe book and give it to PAUL HENRY *(when he will be able to read it), to remember the nice and funny times we had in the kitchen – and of course – to recook the recipes we tested out.*

I never wrote any LaTeX package before, so I tried to give my best and the result looks better than awaited.

Many thanks go to D. P. STORY *and* HERBERT VOSS *who patiently helped me, whenever I had some questions. I guess, they won't forget diverse emails, where the subject was "little q" or "Kleine Anfrage" . . .*

Prepared: October 12, 2009

Contents

1. Introduction

This is a simple style file that typesets recipes in a quite eye-pleasing layout. The `article` class is used and there is a switch set for the `twoside` option.

The layout is simply set with minipages – arranged in a kind of *two-column-style*. The kind of two columns are in a defined ratio which is not 1 : 1, that makes the layout look more pleasant for the eyes, but forces some ideas in how to arrange them properly on even and odd pages vice versa.

The main idea is a simple routine, that arranges the minipages with an `if/else` routine that asks for even or odd pagenumber and then arranges them appropriately.

The first two-pack of minipages is placed on top of the page and contains two graphics (a *small* one and a *bigger* one).

The second two-pack of minipages is placed below the graphics – one of these minipages is empty – and the other contains the *recipe name*, the *cooking time* of the recipe, the *portion* (for how many eaters is this recipe calculated) and the *energy* the food delivers.

The third two-pack of minipages is placed there below and contains the *ingredients* and the *preparation* of the recipe followed by a *hint* at the bottom, where some additional hints to the recipe can be typeset.

Behind this structure of minipages, a background template – if wanted with a smooth transparency setup – can be used, as well as the recipe name in huge transparent letters to bring in an additional special effect. This template managment and all the transparency stuff made me look for packages that are easy to handle and smart to use. That's why the list of the required packages is quite long.

The input mask for the recipes however is very easy to handle. Just a few commands with key-value-pairs and some other commands, which all will be explained in the following sections. The setting of the page is then done automatically.

2. System requirements

The default workflow for this package is

```
latex -> dvi -> dvips -> ps2pdf
```

Due to I have the opportunity to work with Adobe Acrobat, I chose a workflow including the Adobe Distiller with all its really fantastic features e. g. to reduce file size of the produced `pdf`. This workflow is

```
latex -> dvi -> dvips -> distiller -> Acrobat
```

This package is meant as an appetizer to typeset recipes in a quite professional way. The Adobe Distiller is needed to easily implement the background templates (and then turns them to transparent if wanted) and to embed this background graphic only once for the whole document, which saves a lot of file size. I don't know any other package than `graphicxsp` which offers that and this one needs the `distiller -> Acrobat` part of the workflow.

2.1. Required packages

There are quite a lot of required packages with a small explanation, why they are required:

- `xkeyval`: To define some key value pairs for diverse commands.
  http://www.ctan.org/tex-archive/macros/latex/contrib/xkeyval/
- `web`: To have a proper background template managment and some `pdf` document declarations. The `web` package is part of the *AcroTEX eDucation Bundle*, to be downloaded as `acrotex_pack.zip` on the following web site:
  http://www.math.uakron.edu/~dpstory/webeq.html
  The manual is located at
  http://www.math.uakron.edu/~dpstory/acrotex/aeb_man.pdf,
- `graphicx`: To embed graphic `eps` files.
  http://www.ctan.org/tex-archive/macros/latex/required/graphics/
- `pstricks`: To easily draw some lines and geometric objects and have some color managment from `xcolor`.
  http://www.ctan.org/tex-archive/graphics/pstricks/
- `pst-text`: To easily draw text with transparent settings.
  ftp://ftp.dante.de/tex-archive/graphics/pstricks/contrib/pst-text/
- `pst-abspos`: To position elements absolutely on a page.
  http://voss.homedns.org/packages/pst-abspos/
- `nicefrac`: To get some nice typeset fractions.
  http://www.ctan.org/tex-archive/help/Catalogue/entries/nicefrac.html
- `lettrine`: To get initialized paragraphs.
  http://www.ctan.org/tex-archive/help/Catalogue/entries/lettrine.html
- `fancyhdr`: To get a highly customizable header/footer managment.
  http://www.ctan.org/tex-archive/macros/latex/contrib/fancyhdr/

3. Installation

To install the package, copy the `cookybooky.zip` file into the search path of your TEX-system and unzip it.

Open the `cookybooky.ins` your favorite TEX-Editor and latex it once – the `cookybooky.sty` and the `myRecipe.cfg` file will be generated. Be sure to refresh your filename database, so that the new installed package is found by your TEX-system.

In diverse standard TEX-distributions, the fonts Brush Script, Lucida Handwriting and Lucida Calligraphy are installed. If not, install them as well as the required packages.

Then try the example file `ex_1.tex` and see if it works.

4. Package options

The `cookybooky` package offers the following option:

- `myconfig`: This option loads the `myRecipe.cfg` file in which the customer can write his preferences in colors, heading names etc.

5. The commands

5.1. The \ingredients command

The command \ingredients consists of a "&"-delimited, 3-column tabular environment, where the ingredients are set with their numbers and units.

Here a small example:

```
\ingredients{%
2   & EL & Honey\\
1   &    & Lemon\\
100 & g  & Strawberries
}
```

5.2. The \preparation command

The command \preparation is a list of paragraphs that always begins with \init. Within the paragraphs there is explained how to cook the recipe. A counter is set to enumerate the paragraphs automatically.

Here a small example:

```
\preparation{%
\init Preparation Blablabla first step.
\init Preparation Blablabla second step.
...
\init Preparation Blablabla n-th step.
}
```

Here the \init comes from the lettrine package and builds up some initials for a paragraph. The used setting is:

```
\newcounter{init}\setcounter{init}{0}
\renewcommand{\LettrineFontHook}{\color{\initialscolor}}
\newcommand{\init}{%
\lettrine[lines=2, lhang=0.53, loversize=0.15,nindent=13pt]{%
\stepcounter{init}\theinit}{\quad}}
```

5.3. The \hint command

The command \hint is an additional optional command that gives some hints for the recipe, placed at the most bottom of the page.

This is accompanied by some crossed red lines which are generated by simple lines from the pstricks package. The setting is:

```
\newcommand*{\hint}[1]{%
\def\myhint{
\psline[linecolor=\linecolor,linewidth=1.5pt](-0.5,0)(2,0)
\psline[linecolor=\linecolor,linewidth=1.5pt](-0.25,0.25)(-0.25,-1.75)

\hinthead

\begin{minipage}{\linewidth}%
```

```
\itshape#1
\end{minipage}}
}
```

5.4. The \graph command

The command \graph has some optional keys named:

- recipename: The name of the recipe – the default value is TestRecipe.
- recipetime: The time needed to prepare the recipe – the default value is TestTime.
- portion: For how many people is the recipe – the default value is TestPortion.
- joule: How many kJ the food consists – the default value is TestEnergy.
- sgraph: The name of the **s**mall picture placed on top of the page – the default value is erdbeere.
- sdx: The horizontal shift of the **s**mall picture – the default value is 0.
  A negative value – like (−1) – shifts the small graphic 1 cm to the left, a positive value – like 2 – shifts the small graphic 2 cm to the right.
- sdy: The vertical shift of the **s**mall picture – the default value is 0.
  A negative value – like (−1) – shifts the small graphic 1 cm downwards, a positive value – like 2 – shifts the small graphic 2 cm upwards.
- bgraph: The name of the **b**ig picture placed on top of the page – the default value is erdbeere.
- bdx: The horizontal shift of the **b**ig picture – the default value is 0.
  A negative value – like (−1) – shifts the big graphic 1 cm to the left, a positive value – like 2 – shifts the big graphic 2 cm to the right.
- bdy: The vertical shift of the **b**ig picture – the default value is 0.
  A negative value – like (−1) – shifts the big graphic 1 cm downwards, a positive value – like 2 – shifts the big graphic 2 cm upwards.

! The file names of the pictures are given without their extensions and all picture files need to be in eps format. If your graphics are not located in the folder where your .tex file is, you have to define the following path to the graphics \def\graphPath{graphics/} relative to the folder where the .tex file is. Note, that slashes are used for the path.

Here a small example how to enter the keys with some appropriate values:

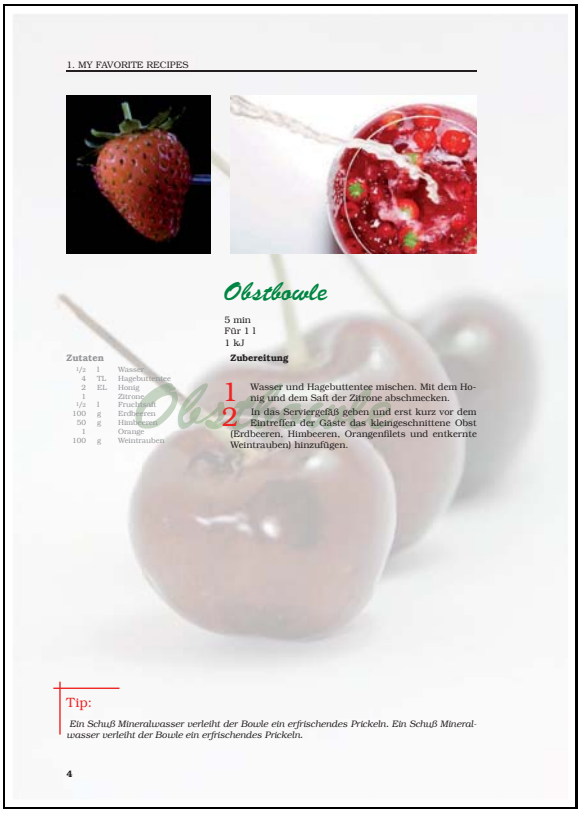
```
\graph[%
recipename=Obstbowle,
recipetime={5 min},
portion={F\"{u}r 1 l},
joule={1 kJ},
sgraph=erdbeere,
sdx=-2,
sdy=0,
bgraph=obstbowle_2,
bdx=0,
bdy=0
]
```

! Be sure to always put the \graph command last in the source code of a recipe. The other commands are enclosed in the graph command, so they need to be defined before.

6. Demo example

On the left side you see the input mask (source code) on the right side the result.

```
\ingredients{%
\nicefrac{1}{2} & l & Wasser\\
4   & TL & Hagebuttentee\\
2   & EL & Honig\\
1   &    & Zitrone\\
\nicefrac{1}{2} & l & Fruchtsaft\\
100 & g  & Erdbeeren\\
50  & g  & Himbeeren\\
1   &    & Orange\\
100 & g  & Weintrauben
}
\preparation{%
\init Wasser und Hagebuttentee mischen.
Mit dem Honig und dem Saft der Zitrone
abschmecken.
\init In das Serviergef\"{a}{\ss} geben und
erst kurz vor dem Eintreffen der G\"{a}ste
das kleingeschnittene Obst (Erdbeeren,
Himbeeren, Orangenfilets und
entkernte Weintrauben) hinzuf\"{u}gen.
}
\hint{
Ein Schu{\ss} Mineralwasser verleiht der
Bowle ein erfrischendes Prickeln.
Ein Schu{\ss} Mineralwasser verleiht der
Bowle ein erfrischendes Prickeln.
}
\graph[%
recipename=Obstbowle,
recipetime={5 min},
portion={F\"{u}r 1 l},
joule={1 kJ},
sgraph=erdbeere,
sdx=-2,
sdy=0,
bgraph=obstbowle_2,
bdx=0,
bdy=0
]
```



7. Handwriting font usage

In this section is described how to choose one of the three predefined handwriting fonts. The command `\selectFont` placed within the preamble does that job:

```
\selectFont[%
%font = hlce
%font = pbsi
font = hlcw
]
```

In the example above the the font family `hlcw` is chosen.

- `hlce` is an abbreviation for the Lucida Calligraphy font family.
- `hlcw` is an abbreviation for the Lucida Handwriting font family.
- `pbsi` is an abbreviation for the Brush Script font family.

Following some example text to see the three fonts.

7.1. Lucida Handwriting

0123456789
abcdefghijklm

nopqrstuvwxyz
ABCDEFGHIJKLM
NOPQRSTUVWXYZ

7.2. Lucida Calligraphy

0123456789
abcdefghijklm
nopqrstuvwxyz
ABCDEFGHIJKLM
NOPQRSTUVWXYZ

7.3. Brush Script

0123456789
abcdefghijklm
nopqrstuvwxyz
ABCDEFGHIJKLM
NOPQRSTUVWXYZ

8. Background templates

The `web` package offers some colored backgrounds or some graphics used as background templates.

8.1. Having a colored background

This can easily be done with the command:

```
\textBgColor{<named_color>}
```

8.2. Embedding a transparent background template

This can be done with the template features of the `web` package:

```
\template{bg_transparent}
```

Here, `bg_transparent.eps` is a graphic – tranparent by nature – and appears on every page from where on the command is set.

! This graphic file is loaded several times (the number of times on how many pages it appears), so the file size might increase linear. Note that `graphicx` has no transparency option, so the graphic needs to be transparent by nature.

8.3. Embedding a transparent background template with Distiller

Load `\usepackage[dvips]{graphicxsp}` in your preamble and as well

```
\embedEPS[transparencyGroup]{p1}{bg}
```

`p1` is a unique name that refers to the embedded graphic and `bg` is the name for the embedded graphic, which has to be an `eps` file originally named `bg.eps` – just entered without its extension.

! This graphic file is embedded only once for the whole document – which means it counts in files size just a little more than once for the complete file size, which saves lots of kB (and depending on the number of pages maybe MB).

The background template finally is inserted with the command

```
\template[%
    name=p1,
    transparency={/ca .15 /BM/Screen}
    ]{bg}
```

from on the line in the source file from which on it should appear. Here you see that the transparency is set to `0.15` – which is almost invisible and smoothens elegantly in the background of every page.

9. Transparency effects

9.1. Embedding the transparent recipe name in the middle of the page

This can be easily done with the command

```
\pstPutAbs(0.5\paperwidth,-0.5\paperheight){%
        \rput(0,0){\parbox{\linewidth}{\centering%
        \pscharpath[linestyle=none,
                    fillstyle=solid,
                    fillcolor=\recipecolorop,
                    opacity=\transpCoeff]{%
        {\bsi{60pt}{75pt}\recipename}}}}
        }
```

where the `\pstPutAbs(0.5\paperwidth,-0.5\paperheight)` defines the center of the page. This command comes from the `pst-abspos` package, `\rput(0,0)` comes from the `pstricks` package and

```
\pscharpath[linestyle=none,
                    fillstyle=solid,
                    fillcolor=\recipecolorop,
                    opacity=\transpCoeff]{%
        {\bsi{60pt}{75pt}\recipename}}
```

comes from the `pst-text` package and produces a transparency of `\transpCoeff` of the recipe name. The transparency coefficient is defined as `\def\transpCoeff{0.3}` and can be defined with any coefficient between 0 and 1, and setup in the `myRecipe.cfg` file.

10. Commands to be individually setup by the customer

10.1. Setting the widths of the minipages

Just define the following commands within the preamble and change the values to your choice:

```
\def\lwA{0.60\linewidth}
\def\lwB{0.35\linewidth}
```

! `\lwA` defines the width of the wider minipage, so please note, that `\lwA` should be the larger number.

10.2. Custom headings

Here are some predefined commands I used:

```
\newcommand{\inghead}{%
\textcolor{\ingheadcolor}{\textbf{Zutaten}\ }
}
\newcommand{\prephead}{%
\textcolor{\prepheadcolor}{\textbf{Zubereitung}\ }
}
\newcommand{\hinthead}{%
\textcolor{\linecolor}{\Large{Tip:}}
}
```

Insert the following commands into the preamble – like this to fit them to your personal needs:

```
\renewcommand{\inghead}{%
\textcolor{\ingheadcolor}{\textbf{<myIngredients>}\ }
}
\renewcommand{\prephead}{%
\textcolor{\prepheadcolor}{\textbf{<myPreparation>}\ }
}
\renewcommand{\hinthead}{%
\textcolor{\linecolor}{\Large{<myHint>:}}
}
```

where `<myIngredients>`, `<myPreparation>` and `<myHint>` could be some translations into another language or some whatever text.

10.3. Custom color managment

There is an easy way to setup the colors for the various commands. The command `\selectRecipeColors` with some optional keys will do that. This command is to be placed into the preamble.

```
\selectRecipeColors[%
recipecolor = webgreen,
ingredcolor = gray,
ingheadcolor = gray,
prepheadcolor = black,
linecolor = red,
recipecolorop = webgreen,
initialscolor = red
]
```

Here you see the default settings, that easily can be redefined by the usage of other colors. The required package `pstricks` is loaded with its options `dvipsnames` and `svgnames` – these options offer many predefined colors by their names.

See

http://melusine.eu.org/syracuse/pstricks/couleurs/couleurs.pdf

for an overview of those predefined colors.

As `xcolor` is a required package by `pstricks`, you as well can define some colors on your own, like

```
\definecolor{<myColorName>}{rgb}{0.97,0.65,0.00}
```

where `<myColorName>` is a unique name you give to your wanted color and to use this name furtheron. `rgb` is the color-model and `0.97,0.65,0.00` are three decimal numbers between 0 and 1 that define the color in the chosen `rgb` color-model.

See more about colors and how to define them and all the possible color-models at

http://www.ctan.org/tex-archive/macros/latex/contrib/xcolor/

10.4. Running headers and footers

The following redefinitions come from the `fancyhdr` package and deliver some possibilities in different headers/footers on even/odd pages.

```
\pagestyle{fancy}
\renewcommand{\sectionmark}[1]%
{\markright{\MakeUppercase{\thesection.\ #1}}}
\renewcommand{\headrulewidth}{0.5pt}
\fancyhf{}
\fancyfoot[LE,RO]{\bf{\thepage}}
\fancyhead[LE,RO]{\rightmark}
%left always EVEN, right always ODD
```