New LaTeX Style for FAO Yearbook *

Boris Veytsman†

2013/08/18, v0.1

Abstract

This package provides class for typesetting FAO Yearbook. This is a refactoring of the faoyeabook package

1 Introduction

The package faoyearbook [1] was written in 2011 for FAO Statistical Yearbook.

The package faosyb is a refactoring of this package. We use the lessons learned and incorporate new design requirements. We use some (actually plenty) code from the previous version, but since we do not have to be compatibility, we can correct some unfortunate decisions.

2 User Guide

The installation of the class follows the usual practice [2] for LaTeX packages:

1. Run latex on `faosyb.ins`. This will produce the LaTeX class `faosyb.cls`.
2. Put the file `faosyb.cls` to the place where LaTeX can find it (see [2] or the documentation for your TeX system).
3. Update the database of file names. Again, see [2] or the documentation for your TeX system for the system-specific details.
4. The file `faosyb.pdf` provides the documentation for the package (this is the file you are probably reading now).

As an alternative to items 2 and 3 you can just put the file `faosyb.cls` in the working directory where your `.tex` file is.

†borisv@lk.net, boris@varphi.com

2.1 Invocation

To use the class, put in the preamble of your document

`\documentclass[`⟨*options*⟩`]{faosyb}`

If the option `web` (default) is chosen, the pages of the book have the dimensions corresponding to A4 paper. However, if the option `print` is chosen, then the pages are printed on a wider area, and crop marks are added for the trimming.

If the option `issuu` is chosen, the internal links are transformed to external in the form suitable for http://www.issuu.com. Note that this option probably does not make much sense unless `web` option is also chosen. However, it is still possible to select both `print` and `issuu` option if someone needs it for an obscure purpose.

The option `Draft` (note the capitalization!) leads to the the large word 'DRAFT' printed across the pages. The standard LaTeX option `draft` leads to the same result, but it also makes other changes, most notably, in the behavior of the `\includegraphics` command and warnings.

`\ifprint` It is possible to query the current mode using the macro `\ifprint`, for example

```
\ifprint
  Stuff for print version
\else
  Stuff for web version
\fi
```

Any branch of this conditional may be empty, so web-only stuff can be coded as

```
\ifprint\else Web-only stuff\fi
```

`\includegraphics` There is a special facilty for `\includegraphics` command to choose a file depending on the current mode of the package. Namely, if there is a file `image_print.pdf` visible by LaTeX, then the commands `\includegraphics{image}` or `\includegraphics{image.pdf}` selects the file `image_print.pdf`. In the case this file is not found, the file `image.pdf` is selected instead. Similarly in the web mode the file `image_web.pdf` will be selected first, and only if it does not exist, `image.pdf` is selected. This rule works also for commands `\includeLargeGraphics` and `\includeExtraLargeGraphics` described below.

Note that at this time there is no similar facility for the `\input` command.

2.2 Setting Parameters

`\faoset` Some parameters in the class can be set with the command `\faoset{`⟨*key=value*⟩`}`, for example

```
\faoset{bgcolor=blue}
```

Most of the parameters are explained below.

2.3 Fonts

\narrowfamily
\textnarrow
\captionfamily
\textcaption

The class uses PT Sans fonts [3] for body text and Arev fonts [4] for math. It defines two additional families: Narrow and Caption, corresponding to the PT Sans Narrow and PT Sans Caption font. They can be selected by the declarations `\narrowfamily` and `\captionfamily` or by the commands `\textnarrow{`⟨*text*⟩`}` and `\textcaption{`⟨*text*⟩`}` following the usual LaTeX conventions. Note that since PT Sans does not provide math alphabet, this choice does not change the mathematical text.

PT Sans Narrow may be useful for typesetting tables, for example,

```
{\scriptsize\narrowfamily
\rowcolors{4}{@bgcolor!30}{@bgcolor!20}
\input{./Tables/P1.DEM_1.tex}}
```

2.4 Colors and Icons for Parts

A Yearbook is separated into parts (more on this below). Each part has its own color and icon. They are set by the keys `bgcolor` and `icon` of the `\faoset` command, for example,

```
\faoset{icon=./Icons/agriculture.png}
\faoset{icon=./Icons/population}
\faoset{bgcolor=blue}
\faoset{bgcolor=green!25!yellow}
```

The parameter for the `icon` key can be any file name (with or without extension), suitable for the `\includegraphics` command. The parameter for the `bgcolor` key can be specified in any form acceptable by xcolor package [5].

The key `tableheadcolor` sets the color for the headers of tables defined by `H` or `P` key (see Section 2.6). Normally it is 30% of the current `@bgcolor` color, but it can be set to any required value.

\selecticon
\selectcolor

Note that `\faoset` command does not change the icon or background color immediately. When issued *before* `\part` command, it sets up icon and color for the next part. If needed, you can manually change this using `\selecticon` and `\selectcolor` commands. In most cases you should *not* use these commands.

@bgcolor
@tableheadcolor
\currenticon

After a `\part` command (or explicit `\selecticon` and `\selectcolor` command we can access the current values of the color in `@bgcolor`, `@tablecolor` colors and `\currenticon` macro.

2.5 Sectioning

\part
\section
\subsection

The main division of the text are `\part`s. The command `\part{`⟨*title*⟩`}` is used for numbered parts, while the command `\part*{`⟨*title*⟩`}` is used for unnumbered parts. The next division are `\section`s and `\subsection`s. They are never numbered. The style does not use `\chapter`s.

`\EndPartIntro` The sections immediately following new parts are special: they are typeset in one column and cannot have floats. The command `\EndPartIntro` switches to the "normal" sections.

2.6 Floats

One of the most important changes from the previous version of the class [1] is the treatment of floats.

In standard LaTeX floats "float": they can be placed by the algorithm anywhere. The previous version made them "sticky": the author explicitly tells TeX where floats should be placed. However, to do so the class required the author to make explicity page breaks, which was not very convenient.

This version has a completely rewritten interface and algorithm for placing floats:

1. Like in standard LaTeX, authors do not normally provide page breaks—TeX tries to make this decision for them.

2. Like in the previous version, floats are put exactly where the authors want them—no default placing and second-guessing.

Here is how it is done.

The main unit of the book is *spread:* a verso page and the corresponding recto page. Each page is divided into four quarters, upper left, upper right, lower left and lower right. We will denote them `ul`, `ur`, `ll`, `lr` for the verso page and `UL`, `UR`, `LL`, `LR` for the recto page (Figure 1). We allow four kinds of floats:

Single floats occupy exactly one quarter. They are denoted as `S`.

Tall floats occupy two quarters stacked vertically (for example, `ul` and `ll`). They are denoted as `T`.

Wide floats occupy two quarters adjacent horizontally (for example, `LL` and `LR`). They are denoted as `W`.

Big floats occupy all four quarters on a page. They are denoted as `B`.

The parameters `{`⟨*type*⟩`}` and `{`⟨*location*⟩`}` are mandatory for floats, for example

```
\begin{map}{T}{ur}
  ...
\end{map}
\begin{chart}{S}{UL}
  ...
\end{chart}
```

For multiquarter floats the location is the location of the upper left corner, so Big float can use only `ul` or `UL` location.

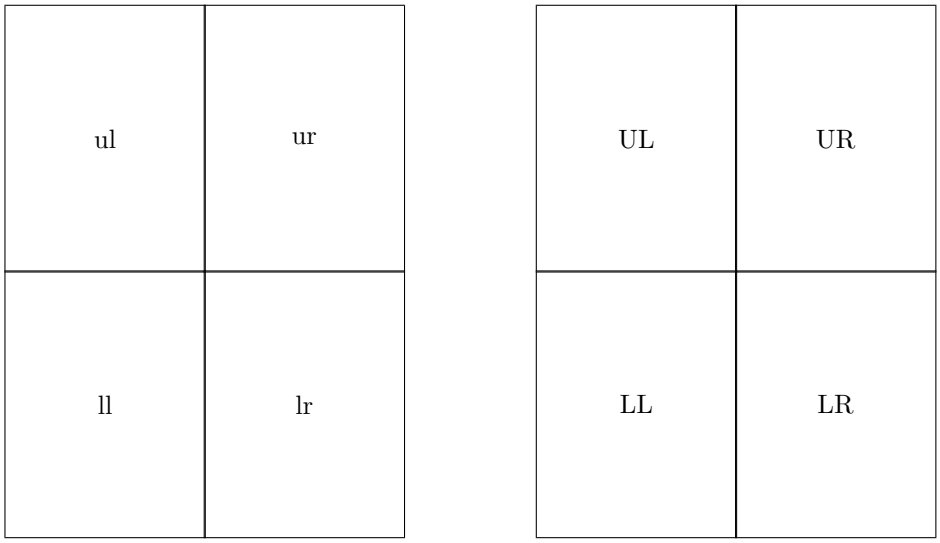

Figure 1: A Spread

Of course, not all combinations are valid: you cannot specify float as `{T}{ll}` or `{W}{UR}`, for example. If you use such combinations, the results may be unpredictable. Also it is not predictable what happens if you try to put overlapping floats (e.g. `{S}{UR}` and `{W}{UL}`).

There are two additional rules:

1. A verso page may have text and floats (still it is recommended that if it has text, then it should not have floats occupying the upper left corner).
2. A recto page may have *either* text `or` floats: if there are floats for this page, all text is moved to the following verso page.

`chart` `map` `table`

There are three types of floats defined by the class:

chart plots and other charts,

map mapped data.

table mini tables.

`caption`

Each of these kinds of material is typeset using the corresponding environment: `chart`, `table` or `map`. Note that the caption for each of these environments *must* precede the graphical material, for example:

```
\begin{chart}{B}{UL}
  \caption{Hunger Data}
  \label{chart:hunger}
  \includegraphics{hunger.pdf}
\end{chart}
```

Note that our class redefines `table` environemnt!. For tables on separate pages use `longtable`.

`\chartwidth` `\chartheight` Inside a `chart`, `map` or `table` it is useful to know the size allocated for the graphics or table, for example, to be able to scale the graphics. Two lengths, `\chartwidth` and `\chartheight` provide this information, so the user can say, for example,

```
\includegraphics[width=\chartwidth, height=\chartheight]{theChart}
```

`\source` Inside a a `chart`, `map` or `table` the macro `\source{`⟨*source*⟩`}` gives the source of the information, for example,

```
\Source{FAO, Statistical Division [FAOSTAT]}
```

`\listoftables` `\listofcharts` `\listofmaps` The standard LaTeX has the command `\listoftables` to produce the list of tables in the document. Our class retains this command and produces two additional commands `\listofcharts` and `\listofmaps` with the obvious meaning.

2.7 Page Breaks

`\clearpage` `\cleardoublepage` `\clearspread` Standard LaTeX has commands for immediate page break (e.g. `\clearpage`) and for switching to the next recto page, possibly ejecting the next verso page (`\cleardoublepage`). The class provides another command `\clearspread`. It switches to the next *verso* page, possibly ejecting the next recto page (and putting there floats intended for this page, if any).

2.8 Tables

To typeset numericall items one should use `d` column identifier with the format `d{`⟨$a.b$⟩`}`, where a is the number of decimal in the integer part of the number, and b is the number of decimal digitst in the fractional part. For example, a number 12.345 corresponds to `d{2.3}`. The column headers are usually *not* numerical, so one need to use `\multicolumn` entries to typeset them. The class defines several such entries:

H produces a centered entry.

P produces an entry of a given length, for example, `P{1.5cm}`

C produces an entry of the length corresponding to the given number of numerical columns. For example, `C{2}` corresponds to a header of two numerical columns. Each column is assumed to be of the size enough to store -99.999.

\hhline

For the rules that do not span the table width `\hhline{`⟨*specificaiton*⟩`}` command from the hhline package should be used. The `{`⟨*specification*⟩`}` argument of this command has many variants, but for our purposes we need only one variant: the command `-` produces a horizontal line spanning one column. The color of this line is determined by the command `\arrayrulecolor{`⟨*color*⟩`}`, issued in the last `>{`⟨*argument*⟩`}` command before the `-` specification. Therefore the command `>{\arrayrulecolor{@tableheadcolor}}-` produces a line of the color `@tableheadcolor`, which is seen as the absence of line. The command `>{\arrayrulecolor{black}}---` produces a black line spanning three columns. Thus if we have a four-column table and want a rule spanning columns 2–3, the following command should be issued:

```
\hhline{>{\arrayrulecolor{@tableheadcolor}}-% Column 1, no rule
  >{\arrayrulecolor{black}}--%  Columns 2 and 3, black rule
  >{\arrayrulecolor{@tableheadcolor}}-}% Column 4, no rule
```

The usual `*` specification may be used for repeating patterns, for example, `*{5}{-}` is equivalent to `-----`.

The vertical bar `|` specification in the `\hhline` argument means an interruption of the line. The interruption is by defalut a black interval, to make it the same color as the header background, use `>{\arrayrulecolor{@tableheadcolor}}|`.

2.9 Publication Descriptions

publication

FAO yearboook describes some FAO publications. These publications should be put inside the environment `publication`. The environment has one mandatory argument, which is the title of the publication, and one optional argument, which sets the file name of the publication cover. Note that the option argument, if present, must precede the mandatory one. If this argument is absent, no cover is included. Inside the environment the macros `\pDescription{`⟨*description*⟩`}`, `\pEdition{`⟨*year*⟩`}{`⟨*edition*⟩`}`, `\pWeb{`⟨*URL*⟩`}` and `\pCycle{`⟨*date*⟩`}` are used to typeset the corresponding items related to the publication. For example,

\pDescription
\pEdition
\pCycle
pWeb

```
\begin{publication}[./Plots/StateOfFoodAndAgriculture.png]{The State
    of Food and Agriculture}
  \pDescription{The State of Food and Agriculture, FAO's major
    annual flagship publication, aims at bringing to a wider
    audience balanced science-based assessments of important issues
    in the field of food and agriculture.  Each edition of the
    report contains a comprehensive, yet easily accessible, overview
    of a selected topic of major relevance for rural and
    agricultural development and for global food security.  This is
    supplemented by a synthetic overview of the current global
    agricultural situation.}
  \pEdition{2010}{Livestock in the balance}
  \pEdition{2011}{Women in Agriculture Closing the gender gap for
    development}
  \pCycle{May each year}
```

```
 \pWeb{http://www.fao.org/docrep/013/i2050e/i2050e00.htm}
\end{publication}
```

Note that, as in the example, some fields may be repeated.

Two spacing parameters can be used for typesetting of publications: `publicationskip` is the amount of additional space between the publications, while `publicationparskip` is the space between the paragraphs inside the publication environment. The default values correspond to the command

```
\faoset{publicationskip=6pt plus 2pt minus 2pt,
        publicationparskip=6pt plus 6pt minus 4pt}
```

2.10 Metadata

`MetadataCollection` `metadata`

Each chart, map of table in the book has a *source*. Soruces are collected in the environemnt `MetadataCollection`, which consists of separate `metadata` environments. Each `metadata` environment has two obligatory arguments—the name of the source and the key. The key is used to identify the metadata in the charts, maps, tables and other objects. The environment may include other commands.

`\source` `\source{`⟨***source***⟩`}` sets the source of the data.

`\owner` `\owner{`⟨***owner***⟩`}` sets the owner of the data.

Note that there is no "description" command because any text which is not an argument of the commands above is considered to belong to the description of the data.

Example of the usage of these commands:

```
\begin{MetadataCollection}
\begin{metadata}{Agricultural population}{P1.DEM.FAO.POP.AGR}

   Agricultural population is defined as all persons depending for
   their livelihood on agriculture, hunting, fishing and forestry.
   It comprises all persons economically active in agriculture as
   well as their non-working dependents. It is not necessary that
   this referred population exclusively come from rural population.

   \source{FILL ME}
   \owner{FILL ME}
\end{metadata}
\end{MetadataCollection}
```

`\refMetadata` The metadata is referenced by the command `\refMetdata{`⟨*key*⟩`}`, for example

```
\refMetadata{P1.DEM.FAO.POP.AGR}
```

This command will be typset as

Source: Agricultural population, page NNNN.

This command must *not* occur in the caption of the chart, map or table.

Note that the package automatically provides backreferencing: all charts, maps and tables where the medatada is referenced, are mentioned in the corresponding metadata section.

The sources of each chart, map or table can be shown in the lists of charts, tables, maps or not. The key `metadataInLists` (by default `false`) determines whether they are shown there. To make them visible, put before the lists

```
\faosetup{metadataInLists=true}
```

2.11 Concepts and Methods

`ConceptsAndMethods` The environment `ConceptsAndMethods` starts a new section "Concepts and Methods". Concepts and methods are collected in the series of `concept` environments. Each environment has one obligatory field: the name of the concept, for example:

```
\begin{ConceptsAndMethods}
  \begin{concept}{Gross domestic product}
    Gross domestic product (GDP) is the market value of all officially
    recognized final goods and services produced within a country in a
    given period of time.
  \end{concept}
  \begin{concept}{Gross state product}
    Gross state product (GSP), or gross regional product (GRP), is a
    measurement of the economic output of a state or province (i.e.,
    of a subnational entity). It is the sum of all value added by
    industries within the state and serves as a counterpart to the
    gross domestic product (GDP).
  \end{concept}
\end{ConceptsAndMethods}
```

2.12 Further Reading

`freading` The special environment `freading` is used for the "further reading" sections of the book. It starts the text from the new page and changes some defaults.

2.13 Subscripts in Text

`\textsubscript` The standard LaTeX defines `\textsuperscript`. The class adds a similar `\textsubscript` command.

2.14 Options

\faoyearbook@size@warning The font-changing options are not used in our setup, so we just produce a warning:

```
\long\def\faoyearbook@size@warning#1{%
  \ClassWarning{faoyearbook}{Size-changing option #1 will not be
    honored}}%
\DeclareOption{8pt}{\faoyearbook@size@warning{\CurrentOption}}%
\DeclareOption{9pt}{\faoyearbook@size@warning{\CurrentOption}}%
\DeclareOption{10pt}{\faoyearbook@size@warning{\CurrentOption}}%
\DeclareOption{11pt}{\faoyearbook@size@warning{\CurrentOption}}%
\DeclareOption{12pt}{\faoyearbook@size@warning{\CurrentOption}}%
```

\ifprint We have a flag shich shows whether we are in Web or print mode

```
\newif\ifprint
\printfalse
\DeclareOption{web}{\printfalse}
\DeclareOption{print}{\printtrue
  \PassOptionsToPackage{papersize={230mm,317mm},layout=a4paper,
    layouthoffset=1cm,layoutvoffset=1cm,twoside}{geometry}}
```

\ifDraft If we are in 'Draft' or 'draft mode', we print a word 'draft' across the page:

```
\newif\ifDraft
\Draftfalse
\DeclareOption{Draft}{\Drafttrue}
\DeclareOption{draft}{\Drafttrue}
```

\if@issuumode Whether we need issuu-style links

```
\newif\if@issuumode
\@issuumodefalse
\DeclareOption{issuu}{\@issuumodetrue}
```

All other options are just sent to the main class:

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{report}}
\ProcessOptions\relax
```

2.15 Loading Class and Packages

We start with the base class and some packages

```
\LoadClass[10pt,twoside,twocolumn]{report}
\RequirePackage{graphicx,xkeyval}
\RequirePackage[table,cmyk]{xcolor}
\RequirePackage{tikz,geometry,dcolumn}
\usetikzlibrary{calc}
\RequirePackage{fancyhdr}
\RequirePackage{lscape,longtable,siunitx,booktabs}
\RequirePackage{multicol,atbegshi,picture,hhline,afterpage}
\RequirePackage[T1]{fontenc}
\RequirePackage[utf8x]{inputenc}
\RequirePackage{pdfpages}
```

```
\RequirePackage[authoryear]{natbib}
\RequirePackage[breaklinks]{hyperref}
\RequirePackage{bookmark}
\if@issuumode
\RequirePackage{issuulinks}
\fi
```

Options for the hyperef package are set as follows:

```
\ifprint
\hypersetup{breaklinks,colorlinks=false,pdfborder=0 0 0,
  pdfauthor={FAO},
  pdfsubject={Statistical Yearbook of the Food And Agricultural Organization for the United Na
  pdftitle={Statistical Yearbook of the Food And Agricultural Organization for the United Nati
  pdfkeywords={FAO, Food Security, Undernourishment, Sustainable agriculture},
  pdfpagelayout=TwoColumnLeft,
  pdfnewwindow=true
}
\else
\hypersetup{breaklinks,colorlinks=false,pdfborder=0 0 0,
  pdfauthor={FAO},
  pdfsubject={Statistical Yearbook of the Food And Agricultural Organization for the United Na
  pdftitle={Statistical Yearbook of the Food And Agricultural Organization for the United Nati
  pdfkeywords={FAO, Food Security, Undernourishment, Sustainable agriculture},
  pdfpagelayout=TwoColumnRight,
  pdfnewwindow=true
}
\fi
```

2.16 Color

We need to tell the printer that we are using CMYK color model. The following is taken from the `pdfx` package (the package itself is not too easy to make work).

```
\def\@pctchar{\expandafter\@gobble\string\%}
\def\@bchar{\expandafter\@gobble\string\\}
\immediate\pdfobj stream attr{/N 4}  file{FOGRA39L.icc}
\edef\OBJ@CVR{\the\pdflastobj}
\pdfcatalog{/OutputIntents [ <<
  /Type/OutputIntent
  /S/GTS_PDFX
  /OutputCondition (FOGRA39)
  /OutputConditionIdentifier (FOGRA39 \@bchar(ISO Coated v2
   300\@pctchar\space \@bchar(ECI\@bchar)\@bchar))
  /DestOutputProfile \OBJ@CVR\space 0 R
  /RegistryName(http://www.color.org)
 >> ]}
```

2.17 Key-Value Interface

`\faoset` We define the family `fao` for our keys:

```
\def\faoset#1{\setkeys{fao}{#1}}
```

2.18 Fonts

We use arev for mathematics:

```
\RequirePackage{arevmath}
```

For body text we use PT Sans:

```
\def\PTSans@scale{0.95}
\def\PTSansNarrow@scale{0.95}
\def\PTSansCaption@scale{0.95}
\renewcommand{\sfdefault}{PTSans-TLF}
\renewcommand{\familydefault}{\sfdefault}
\renewcommand{\bfdefault}{b}
```

`\narrowfamily` We declare a new family, `\narrowfamily`:

```
\DeclareRobustCommand\narrowfamily{\fontfamily{PTSansNarrow-TLF}\selectfont}
```

`\textnarrow` And the matching `\textnarrow` command:

```
\DeclareTextFontCommand{\textnarrow}{\narrowfamily}
```

`\captionfamily` Same with `\captionfamily`:

```
\DeclareRobustCommand\captionfamily{\fontfamily{PTSansCaption-TLF}\selectfont}
```

`\textcaption` And the matching `\textcaption` command:

```
\DeclareTextFontCommand{\textcaption}{\captionfamily}
```

`\normalsize` The basic size is 9.6pt:

```
\renewcommand\normalsize{%
   \@setfontsize\normalsize{9.6pt}{\@xiipt}%
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
   \belowdisplayskip \abovedisplayskip
   \let\@listi\@listI}
\normalsize
```

`\small` This is the small size:

```
\renewcommand\small{%
   \@setfontsize\small\@ixpt{10}%
   \abovedisplayskip 8.5\p@ \@plus3\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus2\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep}%
   \belowdisplayskip \abovedisplayskip}
```

We use `rm` style of URL:

```
\urlstyle{sf}
```

2.19 Margins and Paragraphing

We use a4paper.

```
\geometry{layout=a4paper,
  left=2cm,right=2cm,bottom=2cm,top=2cm,twoside,
  columnsep=30pt, twoside}%
\savegeometry{standard}
```

\parindent
\parskip

We use not indented paragraphs with paragraph borders given by skips

```
\setlength\parindent\z@
\setlength\parskip{6\p@ plus 6\p@ minus 4\p@}
```

2.20 Cropmarks

There are several packages that provide crop marks. Unfortunately they do not work for us because they put crop marks at the background. Since we have colored pages, we want crop marks to be on the foreground.

In this section we re-implement cropmarks of the geometry package, putting the marks on the foreground.

We postpone the code to the beginning of the document to get the proper value of the switch

```
\AtBeginDocument{\ifprint
  \AtBeginShipout{%
    \AtBeginShipoutUpperLeftForeground{%
      \color{black}%
      \@tempdima=\Gm@layouthoffset
      \@tempdimb=\Gm@layoutvoffset
      \put(\@tempdima,-\@tempdimb+6\p@){\line(0,1){50}}%
      \put(\@tempdima-6\p@,-\@tempdimb){\line(-1,0){50}}%
      \advance\@tempdima by \Gm@layoutwidth
      \put(\@tempdima,-\@tempdimb+6\p@){\line(0,1){50}}%
      \put(\@tempdima+6\p@,-\@tempdimb){\line(1,0){50}}%
      \advance\@tempdimb by \Gm@layoutheight
      \put(\@tempdima,-\@tempdimb-6\p@){\line(0,-1){50}}%
      \put(\@tempdima+6\p@,-\@tempdimb){\line(1,0){50}}%
      \advance\@tempdima by -\Gm@layoutwidth
      \put(\@tempdima-6\p@,-\@tempdimb){\line(-1,0){50}}%
      \put(\@tempdima,-\@tempdimb-6\p@){\line(0,-1){50}}%
    }}\fi}
```

In draft mode we put the word ‘DRAFT’ across the page:

```
\AtBeginDocument{\ifDraft
  \AtBeginShipout{%
    \AtBeginShipoutUpperLeft{%
      \color{black!25}%
      \@tempdima=\Gm@layouthoffset
      \@tempdimb=\Gm@layoutvoffset
      \advance\@tempdima by 0.2\Gm@layoutwidth
      \advance\@tempdimb by 0.7\Gm@layoutheight
```

```
      \put(\@tempdima,-\@tempdimb){%
        \rotatebox{45}{%
          \fontsize{6cm}{6cm}\selectfont
          DRAFT}}}}\fi}
```

2.21 Nonfloats

In Faoyearbook we used float package. Since we changed too much in the internals, here we just rewrite the code from scratch.

`\nf@vert@sep` Vertical separation between the floats

```
\newlength\nf@vert@sep
\setlength\nf@vert@sep{30pt}
```

`\nf@width` The width of the nonfloat

```
\newlength\nf@width
```

`\nf@height` The height of the nonfloat

```
\newlength\nf@height
```

`\nf@captionheight` The height reserved for the caption

```
\newlength\nf@captionheight
\setlength\nf@captionheight{32\p@}
```

`\nf@sourceheight` The height reserved for the source lines

```
\newlength\nf@sourceheight
\setlength\nf@sourceheight{48\p@}
```

`\nf@margin` Margin for floats

```
\newlength\nf@margin
\setlength\nf@margin{12\p@}
```

`\nf@trianglebase` The design requres a triangle under the caption. Here it is

```
\newlength\nf@trianglebase
\setlength\nf@trianglebase{12\p@}
```

`\chartwidth` The resulting width of a chart

```
\newlength\chartwidth
```

`\chartheight` The resulting width of a chart

```
\newlength\chartheight
```

`\nf@topskip` Top separation for a nonfloat @topskip

`\nf@bottomskip` Bottom separation for a nonfloat @bottomskip

`\nonfloat@type` The counter to keep the next type to assign

```
\newcount\nonfloat@type
\nonfloat@type=4\relax
```

`\nf@contentsbox` The box to keep the contents of the float

```
\newbox\nf@contentsbox
```

`\nf@mainbox` The box for the float

```
\newbox\nf@mainbox
```

`\newnon@float` The macro `\newnon@float` has the following arguments: TYPE, EXT, NAME, LISTNAME, for example

```
\newnon@float{map}{lom}{Map}{List of Maps}
```

It defines a nonfloat with these parameters.

```
\def\newnon@float#1#2#3#4{%
```

First, we need to define `\ftype@TYPE`: the type of the float. Note that tables are taken, so we need to make a special care of nonfloats that correspond to floats.

```
  \expandafter\ifx\csname ftype@#1\endcsname\relax
    \expandafter\edef\csname ftype@#1\endcsname{\the\nonfloat@type}%
    \multiply\nonfloat@type by 2\relax
  \fi
```

Now we define the extension for the floats

```
  \expandafter\def\csname ext@#1\endcsname{#2}%
```

The macro `\fnum@TYPE` formats the line like "Figure 1". We need to check whether the counter is defined

```
  \expandafter\ifx\csname the#1\endcsname\relax
  \newcounter{#1}\fi
  \expandafter\def\csname fnum@#1\endcsname{#3~\csname
    the#1\endcsname}%
```

Now we want to define the environment TYPE. Since it might be already defined, we first delete this definition, otherwise `\newenvironment` might throw an error

```
  \expandafter\let\csname #1\endcsname\relax
  \expandafter\let\csname end#1\endcsname\relax
```

And the actual definition

```
  \newenvironment{#1}{\non@float{#1}}{\endnon@float}}
```

`\@getfirstletter` An aux macro to get a first letter of a word. Used in constructs

```
\edef\U{\@getfirstletter{AAAAA\@endword}}}
```

```
\def\@getfirstletter#1{\@@getfirstletter#1}
\def\@@getfirstletter#1{#1\@gobbleword}
\def\@gobbleword#1\@endword{}
```

`\non@float` Now we are ready to define the `\non@float` macro. It has three parameters: TYPE, SIZE and PLACEMENT. `\nf@source` is the source of the float.

```
\def\non@float#1#2#3{
```

```
  \def\@captype{#1}%
  \def\nf@size{#2}%
  \def\nf@placement{#3}%
```

The macro `\nf@vert@pos` is either `u` or `l`

```
  \lowercase{\xdef\nf@vert@pos{\@getfirstletter#3\@endword}}
  \global\let\nf@source\@empty
```

Define the source command inside float

```
  \def\source##1{\gdef\nf@source{##1}}
```

Define the caption producing command:

```
\long\def\@makecaption##1##2{\long\gdef\nf@caption{%
    {\bfseries\large\color{white}
      \MakeUppercase{##1}: ##2}}}%
\gdef\nf@caption{}%
```

We calculate the size of the float and skips

```
  \nf@width=\columnwidth
  \nf@height=\dimexpr(\textheight/2-\nf@vert@sep)%
  \if\nf@vert@pos u\relax
    \nf@topskip=\z@
    \nf@bottomskip=\nf@vert@sep
  \else
    \nf@topskip=\nf@vert@sep%
    \nf@bottomskip=\z@
  \fi
  \def\tempW{W}%
  \def\tempT{T}%
  \def\tempB{B}%
  \ifx\nf@size\tempW
    \nf@width=\textwidth
  \fi
  \ifx\nf@size\tempT
    \nf@height=\textheight
    \nf@topskip=\z@
    \nf@bottomskip=\z@
  \fi
  \ifx\nf@size\tempB
    \nf@width=\textwidth
    \nf@height=\textheight
    \nf@topskip=\z@
    \nf@bottomskip=\z@
  \fi
  \chartheight=
    \dimexpr(\nf@height-\nf@captionheight-\nf@sourceheight
    -2\nf@margin-\nf@trianglebase)%
  \chartwidth=\dimexpr(\nf@width-2\nf@margin-0.5\nf@trianglebase)%
  \nf@height=\dimexpr(\nf@height+\nf@topskip+\nf@bottomskip)%
```

Now we construct the main box.

```
  \global\setbox\nf@contentsbox
```

```
    \color@vbox
     \normalcolor
     \vbox to \chartheight
     \bgroup
     \hsize\chartwidth
     \@parboxrestore
     \@floatboxreset
}
```

`\endnon@float` The actual typesetting

```
\def\endnon@float{\@endfloatbox\par
  \hsize=\nf@width
  \setbox\nf@mainbox=\vbox to \nf@height\bgroup
    \hsize=\chartwidth
    \vskip\nf@topskip
    \noindent
    \begin{picture}(0,0)%
      \put(0,0){\color{@bgcolor}%
        \begin{tikzpicture}[baseline=(current bounding box.north)]
          \fill (0,0) -- (\nf@trianglebase,0) --
          (0.5\nf@trianglebase,-\nf@trianglebase) -- cycle;
        \end{tikzpicture}}
    \end{picture}%
    \def\@tempa{chart}%
    \ifx\@tempa\@captype
    \begin{picture}(0,0)%
      \put(0,0){\color{@bgcolor}%
        \begin{tikzpicture}[baseline=(current bounding box.north)]
          \draw(0,0) -- (\nf@width,0);
          \draw (0.5\nf@trianglebase,-2\nf@trianglebase) --
          (0.5\nf@trianglebase,-\chartheight-2\nf@trianglebase
          -\nf@margin) --
          (\nf@width-\pgflinewidth, -\chartheight-2\nf@trianglebase
          -\nf@margin) -- (\nf@width-\pgflinewidth, 0);
        \end{tikzpicture}}
    \end{picture}%
    \fi
    {\color{@bgcolor}\color@block{\nf@width}{\nf@captionheight}{.1\p@}}%
    \hskip\dimexpr(\nf@margin+0.5\nf@trianglebase)%
    \vbox to \nf@captionheight\bgroup
    \nf@caption\vfill
    \egroup\par\nointerlineskip\vskip\nf@trianglebase
    \vskip\nf@margin
    \noindent\hskip\dimexpr(\nf@margin+0.5\nf@trianglebase)%
    \box\nf@contentsbox\par\nointerlineskip
    \vskip\nf@margin
    \hskip\dimexpr(\nf@margin+0.5\nf@trianglebase)%
    \vbox to \nf@sourceheight\bgroup
    \ifx\nf@source\@empty\else
    \noindent{\color{@bgcolor}%
```

```
      \rule{.2em}{.2em}~\rule{.2em}{.2em}~%
      \rule{.2em}{.2em}~\rule{.2em}{.2em}~%
      \rule{.2em}{.2em}\par}
    \noindent Source: \nf@source\par\vfill\fi\egroup
    \vfill\egroup
    \edef\nf@currbox{\expandafter\csname nfbox@\nf@size
      @\nf@placement\endcsname}%
    \global\setbox\nf@currbox=
    \vbox{\box\nf@currbox\nointerlineskip\penalty0\box\nf@mainbox}}
```

`\map` A standard nonfloat:

```
\newnon@float{map}{lom}{Map}{List of Maps}
```

`\table` Another one

```
\newnon@float{table}{lot}{Table}{List of Tables}
```

`\chart` And another one

```
\newnon@float{chart}{loc}{Chart}{List of Charts}
```

2.22 Output Routine

This is hairy because output routines are hairy...

We need several insert boxes. Naming convention: the letter for the box size and two letter code for the location. We use `\newbox` instead of `\newinsert` since we do not use associated `\count`, `\dimen` and `\skip` registers.

```
\newbox\nfbox@S@ul
\newbox\nfbox@S@ur
\newbox\nfbox@S@ll
\newbox\nfbox@S@lr
\newbox\nfbox@S@UL
\newbox\nfbox@S@UR
\newbox\nfbox@S@LL
\newbox\nfbox@S@LR
\newbox\nfbox@T@ul
\newbox\nfbox@T@ur
\newbox\nfbox@T@UL
\newbox\nfbox@T@UR
\newbox\nfbox@W@ul
\newbox\nfbox@W@ll
\newbox\nfbox@W@UL
\newbox\nfbox@W@LL
\newbox\nfbox@B@ul
\newbox\nfbox@B@UL
```

`\@tempboxb` Standard LaTeX has `\@tempboxa`. We need more...

```
\ifx\@tempboxb\@undefined
  \newbox\@tempboxb
\fi
```

`\standard@output` The standard LaTeX output routine is saved as `\standard@output`. We use it for one column pages—maybe one even wants a standard float here?

```
\edef\standard@output{\the\output}
```

`\output` Right now we use standard output on one column pages and the new one with two columns

```
\output{\if@twocolumn\the\nf@output\else\standard@output\fi}
```

`\nf@output` Here we define our own output routine.

```
\newtoks\nf@output
\nf@output {%
```

We define the current boxes `\curr@nfbox...`. Also, `uc` or `lc` mean Upper or Lower Current column

```
  \ifodd\c@page
    \global\let\curr@nfbox@S@ul\nfbox@S@UL
    \global\let\curr@nfbox@S@ur\nfbox@S@UR
    \global\let\curr@nfbox@S@ll\nfbox@S@LL
    \global\let\curr@nfbox@S@lr\nfbox@S@LR
    \global\let\curr@nfbox@T@ul\nfbox@T@UL
    \global\let\curr@nfbox@T@ur\nfbox@T@UR
    \global\let\curr@nfbox@W@ul\nfbox@W@UL
    \global\let\curr@nfbox@W@ll\nfbox@W@LL
    \global\let\curr@nfbox@B@ul\nfbox@B@UL
  \else
    \global\let\curr@nfbox@S@ul\nfbox@S@ul
    \global\let\curr@nfbox@S@ur\nfbox@S@ur
    \global\let\curr@nfbox@S@ll\nfbox@S@ll
    \global\let\curr@nfbox@S@lr\nfbox@S@lr
    \global\let\curr@nfbox@T@ul\nfbox@T@ul
    \global\let\curr@nfbox@T@ur\nfbox@T@ur
    \global\let\curr@nfbox@W@ul\nfbox@W@ul
    \global\let\curr@nfbox@W@ll\nfbox@W@ll
    \global\let\curr@nfbox@B@ul\nfbox@B@ul
  \fi
  \if@firstcolumn
    \global\let\curr@nfbox@S@uc\curr@nfbox@S@ul
    \global\let\curr@nfbox@S@lc\curr@nfbox@S@ll
    \global\let\curr@nfbox@T@uc\curr@nfbox@T@ul
  \else
    \global\let\curr@nfbox@S@uc\curr@nfbox@S@ur
    \global\let\curr@nfbox@S@lc\curr@nfbox@S@lr
    \global\let\curr@nfbox@T@uc\curr@nfbox@T@ur
  \fi
  \let \par \@@par
%
% There are several possibilities when we start the output routine for
% a single column in a two-column layout.
% \begin{enumerate}
```

```
% \item Wide or big non-floats completely cover the page.  In this
% case we do not need to create columns, and directly go to the
% output.
% \item The columnd is occupied by tall or single nonfloats.  We make
% a column of nonfloats and send it further.
% \item There is room for text on the page, but its height
% (\cs{@colroom}) is different from the one known to the page builder
% (\cs{vsize}).  In this case we change \cs{vsize} and return.
% \item The room for text is exactly \cs{vsize}.  In this case we form
% a column and return.
% \end{enumerate}
%    \begin{macrocode}
  \global\@colht=\textheight
  \ifdim\ht\curr@nfbox@B@ul>0.5\baselineskip
    \global\advance\@colht by -\textheight
  \fi
  \ifdim\ht\curr@nfbox@W@ul>0.5\baselineskip
    \global\advance\@colht by -0.5\textheight
  \fi
  \ifdim\ht\curr@nfbox@W@ll>0.5\baselineskip
    \global\advance\@colht by -0.5\textheight
  \fi
  \ifdim\@colht < \baselineskip
    \nf@output@widepage
  \else
    \nf@makecol
  \fi
}
```

`\nf@output@widepage` The macro `\nf@output@widepage` outputs a page completely filled by wide pictures.

```
\def\nf@output@widepage{%
  \if@firstcolumn\else
  \ClassError{faosyb}{Wide or big nonfloats defined too late.  Move
    them up}{I encountered Big or Wide floats when I already made the
    first column.  Please move them up}
  \fi
  \ifdim\ht\curr@nfbox@B@ul>0.5\baselineskip
     \global\setbox\@outputbox\vsplit\curr@nfbox@B@ul to \textheight
  \else
     \setbox\@tempboxa\vsplit\curr@nfbox@W@ul to \textheight
     \setbox\@tempboxb\vsplit\curr@nfbox@W@ll to \textheight
     \setbox\@outputbox\vbox\bgroup
        \box\@tempboxa
        \nointerlineskip
        \box\@tempboxb
     \egroup
  \fi
  \global\vsize\textheight
  \global\@colht\textheight
```

```
  \@outputpage
}
```

`\nf@makecol` This macro tries to make one column of text. If successful, it puts first column into temporary storage, and outputs the page when or if the second column is ready.

When we start `\nf@makecol`, `\@colht` already reflects possible wide nonfloats. This to get `\@colroom`, we need to take into account only the narrow ones

```
\def\nf@makecol{%
  \global\@colroom\@colht
  \ifdim\ht\curr@nfbox@T@uc>0.5\baselineskip
     \global\@colroom=0pt
  \fi
  \ifdim\ht\curr@nfbox@S@uc>0.5\baselineskip
     \global\advance\@colroom by -0.5\textheight
  \fi
  \ifdim\ht\curr@nfbox@S@lc>0.5\baselineskip
     \global\advance\@colroom by -0.5\textheight
  \fi
```

Now there could be two cases. If `\@colroom` is small, we fill the column with the non-floats only. Otherwise we have a "mixed" column with both text and nonfloats.

```
  \ifdim\@colroom<0.5\baselineskip
    \nf@makenfcol
    \unvbox\@cclv
  \else
    \nf@makemixedcol
  \fi}
```

`\nf@makenfcol` This macro outputs a column with only non-floats. If it is called, we already know that the narrow non-floats would fill the column, so we do not do any additional checks.

```
\def\nf@makenfcol{%
  \ifdim\@colht>0.9\textheight  % one tall or two squares
    \ifdim\ht\curr@nfbox@T@uc>0.5\baselineskip
      \setbox\@outputbox\vbox\bgroup
      \boxmaxdepth \@maxdepth
      \vsplit \@curr@nfbox@T@uc to \textheight
      \egroup
    \else
     \setbox\@tempboxa\vbox\bgroup
     \boxmaxdepth \@maxdepth
     \vsplit\curr@nfbox@S@uc to 0.5\textheight
     \egroup
     \setbox\@tempboxb\vbox\bgroup
     \boxmaxdepth \@maxdepth
     \vsplit\curr@nfbox@S@lc to 0.5\textheight
     \egroup
```

```
    \setbox\@outputbox\vbox\bgroup
      \boxmaxdepth \@maxdepth
       \unvbox\@tempboxa
       \nointerlineskip
       \unvbox\@tempboxb
    \egroup
   \fi
 \else  % one square
   \ifdim\ht\curr@nfbox@S@uc>0.49\textheight
     \setbox\@outputbox\vsplit \curr@nfbox@S@uc to 0.5\textheight
   \else
     \setbox\@outputbox\vsplit \curr@nfbox@S@lc to 0.5\textheight
   \fi
 \fi
 \nf@opcol}
```

`\nf@makemixedcol` This macrois used when we have a mix of text with nonfloats (or possibly just text).

We check whether the page builder has the right idea about the text size; if not, we return from the output routine

```
\def\nf@makemixedcol{%
 \ifdim\@colroom=\vsize
   \nf@makemixedcol@
 \else
   \global\vsize=\@colroom
   \unvbox\@cclv
 \fi}
```

`\nf@makmixedcol@` And now the real work of `\nf@makemixedcol@`

```
\def\nf@makemixedcol@{%
  \ifvoid\footins
    \setbox\@outputbox \box\@cclv
  \else
    \setbox\@outputbox \vbox {%
      \boxmaxdepth \@maxdepth
      \box \@cclv
      \vskip \skip\footins
      \color@begingroup
        \normalcolor
        \footnoterule
        \unvbox \footins
      \color@endgroup
      }%
  \fi
  \ifdim\ht\curr@nfbox@S@uc>0.49\textheight
    \setbox\@tempboxa\vsplit\curr@nfbox@S@uc to 0.5\textheight
    \setbox\@outputbox \vbox
      \bgroup
        \box\@tempboxa
```

```
        \nointerlineskip
        \box\@outputbox
      \egroup
  \fi
  \ifdim\ht\curr@nfbox@S@lc>0.49\textheight
    \setbox\@tempboxa\vsplit\curr@nfbox@S@lc to 0.5\textheight
    \setbox\@outputbox \vbox
      \bgroup
        \box\@outputbox
        \nointerlineskip
        \box\@tempboxa
      \egroup
  \fi
  \nf@opcol}
```

`\nf@opcol` This is like the standard LaTeX `\@outputdblcol`, but with the treatment of wide nonfloats.

```
\def\nf@opcol{%
  \if@firstcolumn
    \global\@firstcolumnfalse
    \global\setbox\@leftcolumn\box\@outputbox
  \else
    \global\@firstcolumntrue
    \ifdim\ht\curr@nfbox@W@ul>0.5\baselineskip
      \setbox\@tempboxa\vsplit \curr@nfbox@W@ul to 0.5\textheight
    \else
      \setbox\@tempboxb\box\@tempboxa
    \fi
    \setbox\@outputbox \vbox\bgroup
      \box\@tempboxa
      \nointerlineskip
      \hb@xt@\textwidth {%
        \hb@xt@\columnwidth {%
          \box\@leftcolumn \hss}%
        \hfil
        {\normalcolor\vrule \@width\columnseprule}%
        \hfil
        \hb@xt@\columnwidth {%
          \box\@outputbox \hss}%
      }%
    \egroup
    \ifdim\ht\curr@nfbox@W@ll>0.5\baselineskip
      \setbox\@tempboxa\vsplit \curr@nfbox@W@ll to 0.5\textheight
      \setbox\@ouputbox\vbox\bgroup
        \box\@outputpage
        \nointerlineskip
        \box\@tempboxa
      \egroup
    \fi
    \@outputpage
```

```
    \global\vsize\textheight
    \global\@colht\textheight
    \global\@colroom\textheight
  \fi}
```

`\standard@clearpage` The usual `\clearpage` flushes the floats. We keep it in `\standard@clearpage`

```
\let\standard@clearpage\clearpage
```

`\clearpage` Now we can define `\clearpage` to take care of the mode:

```
\def\clearpage{%
  \if@twocolumn
    \nf@clearpage
  \else
    \standard@clearpage
\fi}
```

`\nf@totalheight` The total height of all non-floats

```
\def\nf@totalheight{\dimexpr(%
  \ht\nfbox@S@UL+
  \ht\nfbox@S@UR+
  \ht\nfbox@S@LL+
  \ht\nfbox@S@LR+
  \ht\nfbox@T@UL+
  \ht\nfbox@T@UR+
  \ht\nfbox@W@UL+
  \ht\nfbox@W@LL+
  \ht\nfbox@B@UL+
  \ht\nfbox@S@ul+
  \ht\nfbox@S@ur+
  \ht\nfbox@S@ll+
  \ht\nfbox@S@lr+
  \ht\nfbox@T@ul+
  \ht\nfbox@T@ur+
  \ht\nfbox@W@ul+
  \ht\nfbox@W@ll+
  \ht\nfbox@B@ul)}
```

`\nf@clearpage` We keep ejecting pages until get rid of nf stuff

```
\def\nf@clearpage{%
  \ifvmode
    \ifnum \@dbltopnum =\m@ne
      \ifdim \pagetotal <\topskip
        \hbox{}%
      \fi
    \fi
  \fi
  \newpage
  \write\m@ne{}%
  \vbox{}%
```

```
  \penalty -\@Mi
  \ifdim\nf@totalheight>\baselineskip
  \null\vfill\clearpage\fi
}
```

`\clearspread` This is like `\cleardoublepage`, but with the logic inverted:

```
\def\clearspread{\clearpage\if@twoside \ifodd\c@page
    \hbox{}\newpage\if@twocolumn\hbox{}\newpage\fi\fi\fi}
```

We need to clear everything at the end

```
\AtEndDocument{\clearpage}
```

References

[1] Boris Veytsman. *LaTeX Style for FAO Yearbook*. FAO UN, 2011.

[2] UK TeX Users Group. UK list of TeX frequently asked questions. http://www.tex.ac.uk/cgi-bin/texfaq2html, 2008.

[3] Pavel Farář. *Support Package for Free Fonts by ParaType*, May 2011. http://mirrors.ctan.org/fonts/paratype.

[4] Stephen G. Hartke. *Arev Sans for TeX and LaTeX*, May 2006. http://mirrors.ctan.org/fonts/arev.

[5] Uwe Kern. *Extending LaTeX's Color Facilities: the xcolor Package*, January 2007. http://mirrors.ctan.org/macros/latex/contrib/xcolor.

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.

H

I

L

M

N

O

P

R

S

T

U

V

W

X

Z