jlreq

1 What is this?

This package provides the class file and JFM (Japanese font metric) files for LuaTeX-ja / pLaTeX / upLaTeX. This aims to implement Requirements for Japanese Text Layout.

2 Installation

Run `make` to generate JFM files. Move the files as follows:

- *.tfm -> $TEXMF/fonts/tfm/public/jlreq
- *.vf -> $TEXMF/fonts/vf/public/jlreq
- jfm-jlreq.lua, jfm-jlreqv.lua, jfm-jlreq-jidori.lua jfm-jlreqv-jidori.lua -> $TEXMF/tex/lua-tex/jlreq
- jlreq.cls, jlreq-helpers.sty -> $TEXMF/tex/latex/jlreq

The other way to install is just to run `make install`. It will install all files in `$TEXMFHOME`.

3 Usage

To use, simply write

```
\documentclass{jlreq}
```

This will set up a document class equivalent to an article in horizontal writing. The engine is automatically determined, but if you want to specify it, pass one of `platex/uplatex/lualatex` as a class option. To switch to vertical writing, pass the `tate` option. Additionally, to make the document class equivalent to report or book, pass the `report` or `book` option respectively. For example, to create a vertical book, write `\documentclass[tate,book]{jlreq}`.
Other common options accepted are `oneside/twoside/onecolumn/twocolumn/titlepage/notitlepage/draft`
Also, passing the `disablejfam` option will not register Japanese fonts for use in mathematics.
You can write the content as you would in a standard document class, but the following commands have been added or extended. Note that this document uses terms from Requirements for Japanese Text Layout without explanation.

3.1 `\jlreqsetup`

This is a command for settings. It can only be used in the preamble. Settings for the document are made either as class options or through `\jlreqsetup`, depending on the setting item.

3.2 `\section`

The `\section` command is extended to accept subtitles in addition to the usual format, as in `\section*[running head]{Heading}[Subtitle]`. Other commands such as `\part` (only in article), `\chapter` (only in book/report), `\subsection`, and `\subsubsection` also accept subtitles.

3.3 `abstract` environment

You can place it in the preamble, and it will be output with `\maketitle`. In the case of two columns, it allows you to output the abstract without columnization. However, this method is now deprecated. By specifying `abstract_with_maketitle=true` in `\jlreqsetup` and writing it before `\maketitle`, you can achieve the same effect.

3.4 `\sidenote`

This command is only defined when the width of the sidenote is positive. In the default basic layout, this width is set to 0. Therefore, `\sidenote` is not defined. Please refer to the later basic layout settings. `\sidenote` outputs a sidenote (or footnote in vertical writing). Internally, it uses `\marginpar`. By default, it has the same format as `\footnote`, but if `sidenote_type=symbol` is specified in `\jlreqsetup`, the format becomes `\sidenote{relevant item}{note}`. For example, you would write:

```
capable of publishing \sidenote{manuscript}{by methods such as printing…}
```

Please refer to the later explanation as well.

3.5 `\endnote`

Specifies an endnote. It has the same format as `\footnote`. By default, the note itself is output just before the heading. This behavior can be controlled by passing `endnote_position` to `\jlreqsetup`. For more details, please see the later explanation of notes. Also, executing `\theendnotes` will output it on the spot.

3.6 `\warichu`

Outputs a warichu (split annotation). The line division positions are calculated automatically (multiple compilations are required). With `\warichu*`, you can manually specify these positions. The format is `\warichu*{(before first line) & (after first line)\\ (before second line) & (after second line)...}`. If `&` is omitted, it will be adjusted automatically.

3.7 `\tatechuyoko`

Outputs horizontal-in-vertical text. You use it as `\tatechuyoko{<content>}`. `\tatechuyoko` will result in an error if used in a non-vertical writing place, but `\tatechuyoko*` will be output as is in a non-vertical writing place.

3.8 `\jidori`

By using `\jidori{<dimension>}{<content>}`, you can output the content justified to the length of the dimension.

3.9 `\akigumi`

By using `\akigumi{<dimension>}{<content>}`, you can output the result of spacing the characters of the content by the length of the dimension. However, the correct output result is not obtained except when using LuaLaTeX.

3.10 `\jafontsize`

Specifies the Japanese font size, similar to `\fontsize`. If `jafontscale=0.9` is specified as a class option, specifying `\fontsize{9pt}{15pt}` will result in a Japanese font size of 8.1pt, but `\jafontsize{9pt}{15pt}` will result in 9pt (with the European font size being 10pt). The second argument is exactly the same as the second argument of `\fontsize`.

3.11 `\`

A macro consisting of a single full-width space (U+3000). It inserts a Japanese character space. In LuaLaTeX, you can also input a Japanese character space with alone.

3.12 Others

- Ruby and encircled points are not provided. It is recommended to use PXrubrica or `luatexja-ruby` (for LuaLaTeX, included in the LuaTeX-ja package).
- When using pLaTeX / upLaTeX, macros `\zw` and `\zh` that expand to `zw` and `zh`, respectively, are defined. In LuaLaTeX, macros of the same name are defined within LuaTeX-ja.
- Requirements for Japanese Text Layout 2.3.2.d state that it is desirable to align the number of lines in each column on the last page of horizontal two-column typesetting, but this process is not performed. You can perform this process by using the `nidanfloat` package and specifying

  ```
  \usepackage[balance]{nidanfloat}
  ```

  However, `\newpage` and `\clearpage` on the last page will not work correctly. For more

details, please see the manual of the `nidanfloat` package.

- The functionality to set fonts is not provided. Japanese fonts can be set using `luatexja-fontspec` or `luatexja-preset` (both included in the LuaTeX-ja package) when using LuaLaTeX. If using dvipdfmx, you can set it with PXchfon.
- The space between Japanese characters (stored in `\kanjiskip` in (u)pTeX and in the `kanjiskip` parameter in LuateX-ja) by default allows for up to a quarter-character width of space, in accordance with the Requirements for Japanese Text Layout. However, due to limitations of TeX' s functionality, this may not always produce appropriate results. If you want to change this value, please redefine `\jlreqkanjiskip`. For example:

```
\documentclass{jlreq}
\renewcommand{\jlreqkanjiskip}{0pt plus .1\zw minus .01\zw}
\begin{document}
(Main text)
\end{document}
```

  The space between Japanese and European characters (stored in `\xkanjiskip` in (u)pTeX and in the `xkanjiskip` parameter in LuateX-ja) can also be changed by redefining `\jlreqxkanjiskip`.
- When specifying `book`, even if you specify `openany` as a class option, a blank page may be inserted after `\mainmatter`. This is to match the behavior of the standard class file. By specifying `\jlreqsetup{mainmatter_pagebreak=clearpage}`, you can prevent the insertion of a blank page, but by default, `\mainmatter` resets the page number, so there may be inconsistencies in the parity of page numbers. Please consider changing to a continuous pagination by specifying something like `\jlreqsetup{frontmatter_pagination={arabic,continuous}}`. For more details, please refer to the following "Front Matter, etc." section.

4 Design options

The design is specified in keyval format via class options or `\jlreqsetup`. However, due to LaTeX' s implementation, cases where input that should be possible is not accepted may occur with class options. In many cases, this can be resolved by removing spaces.
In the following, the following usage will be used:

- `[A/B]`: Either A or B. `[A/B/C]`, etc., are the same.
- `<dimension>`: A dimension recognized by TeX. You can also use simple expressions (like `10pt+10pt`). In class options, you may be able to use special values as follows (these are

available in pLaTeX / upLaTeX by default, but are processed to be available in LuaLaTeX as well): In places like `\jlreqsetup`, you can always describe the full-width width with `\zw` or `\zh`. Below, for example, if `Q` and `H` are available, it will be described as `<dimension;Q,H>`.
  - `Q`, `H`: Interpreted as 0.25mm.
  - `zw`, `zh`: Interpreted as full-width width.
- `<code>`: LaTeX code.
- `<font setting code>`: Commands for font settings, such as `\Large` or `\bfseries`. Multiple specifications like `\Large\bfseries` are also possible.

4.1 Kihon-hanmen

Class option.

- `paper=[<paper size name>/{<dimension>,<dimension>}]`: Specifies the paper size. You can specify from `a0paper` to `a10paper`, `b0paper` to `b10paper` (ISO B series), or `b0j` to `b10j` for JIS B series. Also available are `letterpaper`, `legalpaper`, and `executivepaper`. Alternatively, you can directly specify dimensions with `{<width>,<height>}`.
- `fontsize=<dimension;Q,H>`: Specifies the font size for Latin script. The default is 10pt.
- `jafontsize=<dimension;Q,H>`: Specifies the font size for Japanese script.
- `jafontscale=<real number>`: The ratio of Japanese font size to Latin font size (Japanese / Latin). Ignored if both `fontsize` and `jafontsize` are specified. The default is 1.
- `line_length=<dimension;zw,zh>`: The length of a line. By default, it's 0.75 times the paper width in the direction of character progression. The actual value is adjusted to be a multiple of the length of one character.
- `number_of_lines=<natural number>`: The number of lines per page. By default, it's a value that makes it 0.75 times the paper height in the direction of line progression.
- `gutter=<dimension;zw,zh>`: The size of the gutter margin.
  - Without `tate`option: It's the margin on the left for odd pages and on the right for even pages.
  - With `tate` option: It's the opposite.
  - If `twoside` is not specified in the class options, the margin is always set as for odd pages.
- `fore-edge=<dimension;zw,zh>`: The size of the fore-edge margin (the side opposite the gutter). It's implemented for convenience, although it's not used when margins are specified according to "Requirements for Japanese Text Layout".
- `head_space=<dimension;zw,zh>`: The amount of space at the top (head). The default value centers the text block vertically.
- `foot_space=<dimension;zw,zh>`: The amount of space at the bottom (foot). The default

value centers the text block vertically.

- `baselineskip=<dimension;Q,H,zw,zh>`: The line feed. By default, it's 1.7 times the `jafontsize`.
- `linegap=<dimension;Q,H,zw,zh>`: The line gap.
- `headfoot_sidemargin=<dimension;zw,zh>`: The left and right margin for headers and footers.
- `column_gap=<dimension;zw,zh>`: The gap between columns (only for `twocolumn`).
- `sidenote_length=<dimension;zw,zh>`: Specifies the width of the sidenote.

4.2

class option

- `open_bracket_pos=[zenkaku_tentsuki/zenkakunibu_nibu/nibu_tentsuki]`: Specifies the positioning of opening brackets at the start of a line. Options are:
  - `zenkaku_tentsuki`: Full-width characters are indented at the start of a paragraph.
  - `zenkakunibu_nibu`: Full-width at the start of a paragraph, half-width at a line break.
  - `nibu_tentsuki`: Half-width indentation at the start of a line.
- `hanging_punctuation`: Enables hanging punctuation.

4.3 Reverse pagination

class option.

- `use_reverse_pagination`: Enables reverse pagination functionality. Defines a 'read-only counter' named `jlreqreversepage`. Commands like `\arabic` and `\value` can be applied to it, and `\thejlreqreversepage` is defined as `\arabic{jlreqreversepage}`.

4.4 Note related

Specified by `\jlreqsetup`.

- `reference_mark=[inline/interlinear]`: Specifies the placement of reference marks. Options are `inline` (in the line after the relevant item) or `interlinear` (above the line for horizontal writing, or to the right for vertical writing).
- `footnote_second_indent=<dimension>`: Sets the indent for the second and subsequent lines of footnotes (horizontal writing) or sidenotes (vertical writing).
- `sidenote_type=[number/symbol]`: Determines the correspondence between the sidenote and the text. Options are `number` (default, with a serial number indicating the note) or `symbol` (with a specific symbol and the word with the note emphasized).

- `sidenote_symbol=<code>`: When `sidenote_type=symbol` is specified, this sets the symbol to be used at the note's location. THe default is *
- `sidenote_keyword_font=<font setting code>`: Specifies the font for the word with the sidenote when `sidenote_type=symbol` is used.
- `endnote_second_indent=<dimension>`: Sets the indent for the second and subsequent lines of endnotes. `endnote_position=[headings/paragraph/{_<heading name 1>,_<heading name 2>,...}]`: The output location of the endnote can be specified. `headings` will output the note before each heading (default), `paragraph` will output it when a new paragraph starts, and specifying `endnote_position={_chapter,_section}` will output it before `\chapter` and `\section`. To specify `<_heading name>`, the target heading must be created using the functionality of this class file.
- `warichu_opening=<code>`, `warichu_closing=<code>`: These are inserted before and after the warichu (split annotation). The default is parentheses ().

4.5 Caption

You can change the captions for figures and tables using `\jlreqsetup`. All settings allow for environment-specific customization. For example, `caption_font=\normalsize,table=\Large` means that within the table environment, `\Large` will be applied, while `\normalsize` will be applied in other environments. Other settings follow the same pattern.

- `caption_font=<font setting code>`: Specifies the font for the caption itself.
- `caption_label_font=<font setting code>`: Specifies the font for the caption label.
- `caption_after_label_space=<dimension>`: Specifies the space between the label and the caption text.
- `caption_label_format=<code>`: Specifies the format of the label. For example, `caption_label_format={#1:}` where `#1` is replaced with something like “Figure 1” .
- `caption_align=[left/right/center/bottom/top]`: Specifies the position of the caption. For instance, `{center,*left}` means it’ s usually centered, but aligned left when the caption is large.

4.6 Quotation

- `quote_indent=<dimension>`: Specifies the indentation for quotes. The default is `2\zw`. The length of a line is adjusted to be an integer multiple of the character size.
- `quote_end_indent=<dimension>`: Specifies the indentation for the end of quotes. The default is `0\zw`.
- `quote_beforeafter_space=<dimension>`: Specifies the space before and after quotes. Set-

ting quote_beforeafter_space=1\baselineskip will result in a one-line space.

- quote_fontsize=[normalsize/small/footnotesize/scriptsize/tiny]: Specifies the font size for quotes.

4.7 Itemization

- itemization_beforeafter_space=<dimension>: Specifies the space before and after the itemization. For example, itemization_beforeafter_space={i=<dimension>} sets the space for the top level only. itemization_beforeafter_space={0pt,i=10pt,ii=5pt} sets 10pt for level 1 itemization, 5pt for level 2, and 0pt for others. Levels are indicated by lowercase Roman numerals.
- itemization_itemsep=<dimension>: Specifies the space between items.

4.8 Front matter etc.

You can specify the behavior of \frontmatter / \mainmatter / \backmatter / \appendix with \jlreqsetup.

- frontmatter_pagebreak=[cleardoublepage/clearpage]: Specifies the command name to execute the page break at \frontmatter. If it is empty, nothing is done.
- frontmatter_counter={<counter name>={value=<value>, the=<code>, restore=[true/false]},⋯ }: Specifies the operation of the counter at \frontmatter. For example, chapter={value=0,the={[\arabic sets the value of the chapter counter to 0 and \thechapter to [\arabic{chapter}]. By default, the value and \the<counter name> definition are restored at \mainmatter, but this behavior is suppressed by specifying Prestore=false‘.
- frontmatter_heading={<heading command name>={<setting>}, ⋯ }: Changes the behavior of the heading command. In addition to the items that can be specified by \Delare***Heading, the following are accepted.
  - heading_type=[Tobira/Block/Runin/Cutin/Modify]: The type of heading1. If Modify is specified, it will be changed by \ModifyHeading.
  - heading_level=<number>: Specifies the level of the heading command. If not specified, the value at \frontmatter is used. This is ignored when heading_type=Modify.
  - restore=[true/false]: If true is specified, the original definition is restored at \mainmatter. The default is true.
- frontmatter_pagestyle={<page style name>[,restore=[true/false]]}: Switches to the specified page style at \frontmatter. By default, it returns to the original page style at \mainmatter, but this can be prevented by specifying restore=false.
- frontmatter_pagination={<page number specification>[,continuous,independent]}:

Specifies the output format of the page number, such as `frontmatter_pagination=roman`, with the LaTeX command name. In addition, continuous makes it a continuous number, and independent makes it a separate number.

- `frontmatter_precode=<code>`: The code that is executed first at `\frontmatter`.
- `frontmatter_postcode=<code>`: The code that is executed last at `\frontmatter`.

There are also settings that change `frontmatter` to `mainmatter`, `backmatter`, or `appendix`. However, there are some differences as follows.

- `restore=[true/false]` is an invalid setting.
- Pmainmatter_pagination‘ cannot specify continuous or independent.
- `appendix_pagebreak`, `appendix_pagestyle`, `appendix_pagination` do not exist.

Abstract

- `abstract_with_maketitle=[true/false]`: If the abstract environment is written before `\maketitle`, it delays its content and outputs it with `\maketitle`. Even in the case of two columns, it is output in one column. The default is false. This option is available only when `article` and `report` are specified.

5 Headings

You can create new headings with commands like `\New***Heading` (is a string that corresponds to the type of heading). The format is always `\New***Heading{<command name>}{<level>}{<settings>}`. Also, `\Renew***Heading`, `\Provide***Heading`, and `\Declare***Heading` are available at the same time. They are respectively

- `\Renew***Heading`: If the command with the specified name is not defined, it causes an error.
- `\Provide***Heading`: If the command with the specified name is not defined, it defines a new heading command.
- `\Declare***Heading`: It defines a new heading command regardless of whether the command with the specified name is defined or not.

5.1 Tobira headings

You can create it with \NewTobiraHeading. It creates a command with the same format as the usual class file’ s \section and so on. The settings are as follows. .

- `type=[han/naka]`: It creates a half-title heading if `han` is specified, and a middle-title heading if `naka` is specified.

- pagestyle=<page style name>: It specifies the page style of the heading area.
- label_format=<code>: It specifies the command to output the label. For example, specify it as label_format={Chapter \thechapter}.
- format=<code>: It specifies the format to actually output. #1 is replaced with the label, #2 with the heading text. In this case, the commands \jlreqHeadingLabel and \jlreqHeadingText are available internally. They are commands that take one argument each, and output the given argument itself if the label and heading text are not empty, respectively, and output nothing otherwise. For example, if you specify it as format={[\jlreqHeadingLabel{Label=#1}]}, it outputs [Label=<label>] if the label is not empty, and [] otherwise. Note that #1 is replaced not with the label itself, but with the code that has the adjustment of the space and so on. Therefore, you may get an unexpected result. #2 is the same.
- number=[true/false]: It specifies whether to number or not. However, even if number=false is specified, the corresponding counter is defined. Also, the definition of \the<counter name> is not changed, so you need to redefine it if necessary.

5.2 Block headings

You can create it with \NewBlockHeading. It creates a command with the format \<command name>*[running head]{heading text}[subtitle]. The settings are as follows.

Format related

- font=<font setting code>: It specifies the font of the heading.
- subtitle_font=<font setting code>: It specifies the font of the subtitle.
- label_format=<code>: It specifies the format of the label. Specify it as label_format={Chapter \thechapter}, for example.
- subtitle_format=<code>: It specifies the format of the subtitle. Specify it as subtitle_format={ “#1” }, for example. #1 is replaced with the subtitle itself.
- format=<code>: It specifies the format of the entire heading. #1 is replaced with the label, #2 with the heading text, #3 with the subtitle. Internally, the commands \jlreqHeadingLabel, \jlreqHeadingText, and \jlreqHeadingSubtitle are available. They are commands that take one argument each, and output the given argument itself if the label, heading text, and subtitle are not empty, respectively, and output nothing otherwise. For example, if you specify it as format={[\jlreqHeadingLabel{Label=#1}]}, it outputs [Label=<label>] if the label is not empty, and [] otherwise. Note that #1 is replaced not with the label itself, but with the code that has the adjustment of the space and so on. Therefore, you may get an unexpected result. #2 and #3 are the same.

Indent related

- `align=[left/center/right]`: It specifies the horizontal alignment of the heading position.
- `indent=<dimension>`: It specifies the amount of indentation of the entire heading.
- `end_indent=<dimension>`: It specifies the amount of back indentation of the entire heading.
- `after_label_space=<dimension>`: It specifies the amount of space between the label and the heading text.
- `second_heading_text_indent=[<dimension>/{<dimension>,<dimension>}]`: It specifies the indent of the second and subsequent lines of the heading text. It specifies from the beginning of the first line of the heading text, but if you put `*` at the beginning like `second_heading_text_indent=*1\zw`, it specifies from the beginning of the label. Also, if you specify it as `second_heading_text_indent={<when there is a label>,<when there is no label>}`, you can change the value depending on the presence or absence of the label. You can also use * in the specification of `<when there is a label>`.
- `subtitle_indent=<dimension>`: It specifies the amount of indentation of the subtitle. It specifies from the beginning of the first line of the heading text. However, if you put `*` at the beginning like `subtitle_indent=*1\zw`, it specifies from the beginning of the label. It is valid only when `subtitle_break=true`.

Others

- `subtitle_break=[true/false]`: It specifies whether to break the line between the heading text and the subtitle.
- `allowbreak_if_evenpage=[true/false]`: It allows a page break immediately after the heading if it is on an even page.
- `pagebreak=[clearpage/cleardoublepage/clearcolumn/nariyuki/begin_with_odd_page/begin_with_`
  It specifies the page break before the heading2. They are respectively, page break, `\cleardoublepage` execution, column break, as it is, start with odd page, start with even page.
- `pagestyle=<page style name>`: It specifies the page style of the heading area.
- `afterindent=[true/false]`: It specifies whether to indent the paragraph immediately after the heading.
- `column_spanning=[true/false]`: It makes the heading span columns. It is ignored when `pagebreak=nariyuki` or `pagebreak=clearcolumn`.
- `number=[true/false]`: It specifies whether to number or not. The same caution as `\NewTobiraHeading` is required.

Gyo-dori You can specify gyo-dori setting in one of the following ways.

- Specify the number of lines and place it in the center. Specify the number of lines with `lines=<natural number>`. You can also specify the number of lines to add before and after with `before_lines=<natural number>` and `after_lines=<natural number>`. For example, if you specify `lines=3,after_lines=1`, it will be placed in four lines, and the space after will be one line larger than the space before. The space specified by `before_lines` does not enter at the top of the page, but if you put `*` at the beginning like `before_lines=*1`, it always enters.
- Specify the number of lines and either the space before or after. Specify the number of lines with `lines=<natural number>`, and the space before or after with either `before_space=<dimension>` or `after_space=<dimension>`.
- Specify the space before and after. Specify it with `before_space=<dimension>` and `after_space=<dimension>`.

5.3 Gyo-dori for consecutive headings

You can set the gyo-dori settings for block headings that are placed consecutively with `\SetBlockHeadingSpaces`. `\SetBlockHeadingSpaces` is used as follows:

```
\SetBlockHeadingSpaces {
    {_part{lines=3,before_lines=1},_section{lines=2},_subsection{lines=2}}
    [lines=5]{_section,23pt,_subsection,16pt}
}
```

This means the following.

- If the headings are placed in the order of `\part`, `\section`, `\subsection`, and the front and back are not headings, then `\part` is three-line spacing + one line space before, and `\section` and `\subsection` are two-line spacing.
- If the headings are placed in the order of `\section`, `\subsection`, and the front and back are not headings, then the whole is five-line spacing, and there is `23pt` space between `\section` and `\subsection`, and `16pt` space after `\subsection`.

The individual settings are as follows.

- Each `{}` contains `_<heading command name>` or `<dimension>` separated by commas.
- You can add settings enclosed in `[]` at the beginning. This is a setting for the entire heading that is placed consecutively. `lines / before_lines / after_lines / before_space / after_space` are available. Each meaning is the same as the line spacing specification described above.

- The dimension is the amount of space as it is.
- You can add settings enclosed in `{}` after `_<heading command name>` to set the amount of space for that heading. If not set, there is no space before and after.
- In the settings enclosed in `{}` for the heading, `lines / before_lines / after_lines / before_space / after_space` are available. Each meaning is the same as the line spacing specification described above.
- If you make the part enclosed in `{}` only `*`, (for example, `_section{*}`), it uses the same setting as when it is placed alone.

Note that whether the headings are consecutive or not is determined simply by whether the block heading commands are written in succession. Therefore, even if there are commands that are not related to the output between the heading commands, they are not considered to be consecutive headings. However, if only spaces, line breaks, or `\label[<option>]{<argument>}`⋯`{<argument>}` are inserted between the heading commands, they are considered to be consecutive headings.

5.4 Run-in headings

Run-in headings are created with `\NewRuninHeading`. The command creates a heading with the same format as the standard document class's `\section`, i.e. `\<command name>*[running head]{heading text}`. The settings are as follows:

- `font=<font setting command>`: Specifies the font of the heading.
- `indent=<dimension>`: Specifies the indentation of the entire heading text.
- `after_label_space=<dimension>`: Specifies the space between the label and the heading text.
- `label_format=<code>`: Specifies the format of the label. For example, `label_format={\theparagraph}`.
- `after_space=<dimension>`: Specifies the space between the heading and the main text.
- `number=[true/false]`: Specifies whether to number the heading or not. The same caution as `\NewTobiraHeading` applies.

5.5 Cut-in headings

You can create cut-in headings with `\NewCutinHeading`. The command has the format `\<command name>{heading text}`. The settings are as follows:

- `font=<font setting command>`: Specifies the font of the heading.
- `indent=<dimension>`: Specifies the indentation of the entire heading.
- `after_space=<dimension>`: Specifies the space between the heading and the text.
- `onelinemax=<dimension>`, `twolinemax=<dimension>`: If the length of the heading text is

less than or equal to onelinemax, the window heading is output in one line. If it is less than or equal to twolinemax, it is output in two lines. Otherwise, it is output in three lines. The default values are 6 characters and 20 characters, respectively.

5.6 \ModifyHeading

\ModifyHeading is a command that changes the settings of a heading command that has already been defined (using one of the above commands). For example,

```
\ModifyHeading{section}{lines=10}
```

changes only the line spacing of \section to 10 lines, while keeping the font and other settings the same. You cannot change the type of heading with this command.

5.7 \SaveHeading

\SaveHeading saves the definition of a heading command. For example, you can use it like this:

```
\SaveHeading{section}{\restoresection} % Save the contents of \section to \restoresection
\RenewBlockHeading{section}{1}{font=…} % Redefine \section with a new font.
…
\restoresection % Restore the contents of \section.
```

6 Page Style

You can define a page style using

```
\NewPageStyle{<page style name>}{<settings>}
```

<settings> is in keyval format. You can apply the defined page style with \pagestyle. The settings are as follows:

- yoko: Prints horizontally at the top and bottom. Default.
- tate: Prints vertically on the fore-edge side.
- running_head_font=<font setting command>: Specifies the font for the running head.
- nombre_font=<font setting command>: Specifies the font for the page number.
- running_head_position, nombre_position: Specifies the position of the running head and the page number. The specification method changes depending on whether yoko or tate is specified.
  - yoko specified: You can specify it like top-left. You can use top / bottom / center / left / right / gutter / fore-edge. left and right are for odd pages. If

twoside is specified, even pages are the opposite.

  - tate specified: You can specify <dimension>. running_head_position specifies the drop amount from the top of the running head, and nombre_position specifies the raise amount from the bottom of the page number.

- nombre=<format>: Specifies the page number to output. The default is \thepage.
- odd_running_head=<format>, even_running_head=<format>: Specifies the running head for odd and even pages, respectively. If you specify a name starting with _ like _section, it outputs the corresponding heading. (_section outputs the current \section.)
- mark_format={[odd=<format>/even=<format>/_<heading command name>=<format>],⋯ }: Specifies the format for outputting headings to the running head. Specify it like mark_format={_section={Section \thesection: #1},_chapter={Chapter \thechapter \quad #1}}. You can also specify odd or even instead of the heading command name, which will be the format for the running head on odd/even pages. It is implemented by defining \sectionmark etc. when executing \pagestyle.
- nombre_ii=<format>: Specifies the second page number. You can also specify the location with nombre_ii_position and the font with nombre_ii_font. The specification method is the same as nombre and nombre_position.
- odd_running_head_ii, even_running_head_ii, running_head_ii_position, running_head_ii_font are also available. If nombre_ii_position or running_head_ii_position is not specified, it will be set to the same position as nombre_position or running_head_position when yoko is specified. When tate is specified, it will be displayed in the place following the first page number or running head.
- odd_head_format=<format>, odd_foot_format=<format>, even_head_format=<format>, even_foot_format=<format>P: Specifies the format for the header and footer. #1will be replaced with the entire header or footer. However,#1may contain code for position adjustment, so it may not work as expected, especially whentateis specified in\NewPageStyle. For example, to draw a rule under the header on odd pages, you can do odd_head_format={\underline{\makebox[\jlreqyokoheadlength]{#1}}}. \jlreqyokoheadlengthis a macro defined in this class file and gives the horizontal length of the header. (The footer length is the same.) The vertical length, i.e., the length whentateis specified in\NewPageStyle, can be obtained with \jlreqtateheadlength‘.

\RenewPageStyle, \ProvidePageStyle, \DeclarePageStyle are also available1.
You can modify an existing page style with \ModifyPageStyle.

7 JFM

We use the following custom JFMs. Some packages may reset the settings to use their own JFMs or the standard JFMs. In that case, you need to set the appropriate package options to use the JFMs of this class file.

7.1 For pLaTeX/upLaTeX

The names of the JFMs are as follows. The characters enclosed in `[]` may or may not be included depending on the settings.

```
[u][b][z]jlreq[g][-v]
```

Each character is included in the following cases.

- `u`: When using upLaTeX
- `b`: When using hanging punctuation. (When the class option hanging_punctuation is specified.)
- `z`: When the space before the opening brackets at the beginning of a line is two-bun for the beginning of a paragraph and one-bun for the line break. (When the class option `open_bracket_pos=zenkakunibu_nibu` is specified.)
- `g`: For gothic fonts.
- `-v`: For vertical writing.

For example, if you process a source without using hanging punctuation and specifying `open_bracket_pos=zenkakunibu_nibu` as a class option with pLaTeX, the JFM named zjlreq will be used for horizontal writing mincho font.
For LuaLaTeX

- The JFM for horizontal writing is jlreq
- The JFM for vertical writing is jlreqv

The same JFM is used for gothic fonts. This class file changes the JFM of LuaTeX-ja standard to these.

8 Others

If the class option `jlreq_notes` is passed, a notification will be given when a setting that contradicts the description of Japanese typesetting processing is made.

9 jlreq-complements

`jlreq-complements` is a package that customizes the environments that are typically provided by LaTeX document classes. You can use it as follows. It accepts the following options:

- `platex`, `uplatex`, `lulalatex`: Specify the engine.
- `setupname=<name>`: Specify the name of the command for customization. The default is `jlreqcomplementssetup`, and you can set it by writing `\jlreqcomplementssetup{<settings>}` in the preamble.

In `jlreq`, it is loaded as `\usepackage[<engine recognized in jlreq>,setupname=jlreqsetup]{jlreq-complements}`, so you can customize the environments with the usual `\jlreqsetup`. To make this work well with an existing name, the original command and the new one need to be compatible. It is usually better to avoid this.

9.1 `thebibliography` environment

- `thebibliography_heading=<code>`: Specify the command to output the heading of the `thebibliography` environment. Use it like `thebibliography_heading={\section*{\refname}}`.
- `thebibliography_after_label_space=<dimension>`: Specify the space after the label of each item in the `thebibliography` environment.
- `thebibliography_indent=<dimension>`: Specify the indentation of the entire `thebibliography` environment.
- `thebibliography_mark=<code>`: Specify the code to register the heading of the `thebibliography` environment in the column.
- `thebibliography_precode=<code>`, `thebibliography_postcode=<code>`: Specify the code that is executed before and after the `thebibliography` environment, respectively.

9.2 `theindex` environment

- `theindex_heading=<code>`: Specify the command to output the heading of the `theindex` environment.
- `theindex_mark=<code>`: Specify the code to register the heading of the `theindex` environment in the column.
- `theindex_twocolumn=[true/false]`: Specify whether to output the `theindex` environment in two columns.
- `theindex_column_gap=<dimension>`: Specify the column gap in the `theindex` environment when `theindex_twocolumn=true`.
- `theindex_column_rule_width=<dimension>`: Specify the value of `\columnseprule` in the

theindex environment when theindex_twocolumn=true.

- theindex_pagestyle=<page style name>: Specify the page style for the theindex environment.
- theindex_postcode=<code>, theindex_precode=<code>: Specify the code that is executed before and after the theindex environment, respectively.

9.3 Theorem environment

- theorem_beforeafter_space=<dimension>: Specify the space before and after the theorem environment.
- theorem_label_font=<font setting code>: Specify the font for the label part of the theorem environment.
- theorem_font=<font setting code>: Specify the font for the body of the theorem environment.
- theorem_indent=<dimension>: Specify the indentation of the body of the theorem environment.
- proof_label_font=<font setting code>: This setting is only valid when the amsthm package is loaded. Specify the font for the label of the proof environment.

When the amsthm package is loaded, a new theorem style jlreq is defined and the current style is changed to jlreq. In this case, the above settings function as settings for this jlreq style.

10 LICENSE

This package is distributed under the BSD 2-Clause License. See LICENSE.

11 CHANGELOG

- 2017-02-08
  - First release.
- 2017-02-17
  - Fixed bugs.
  - Implement abstract environment.
  - Changed/Added some keys to class option/\jlreqsetup
  - Stopped to load pxrubirica, luatexja-ruby and nidanfloat.
- 2017-03-14
  - Fixed bugs.
  - \sffamily etc. also change the Japanese font family.
  - Added many options to \DeclareBlockHeading.
  - Some options related to quote environment etc.

- 2017-03-20
  - Fixed bugs.
  - Insert some spaces around `\footnote / \sidenote / \endnote`.
- 2017-04-04
  - Fixed a bug.
  - Added options `tate` and `font` to `\DeclarePageStyle`.
- 2017-04-29
  - Fixed bugs.
  - Added `jafontsize` and `jafontscale` options and `\jafontsize`.
  - Added `\tatechuyoko`.
  - `jlreq_warnings` -> `jlreq_notes` (class option).
  - Moved some class options to `\jlreqsetup`.
  - Added some options to `\jlreqsetup`.
  - `paper={<height>,<width>}` -> `paper={<width>,<height>}`.
- 2017-06-11
  - Stopped to load `plext` and `lltjext`.
  - Added `align` to `\DeclareBlockHeading` and delete `indent=center`, `end_indent=center`.
  - Changed `\kcatcode` for some characters (upLaTeX).
- 2017-08-13
  - Added `column_spanning` to `\DeclareBlockHeading`.
  - Sidenotes are a part of the main text now.
  - Changed the default length of sidenotes to 0.
  - jlreq does not define `\sidenote` if the length for sidenotes is zero.
  - Added a command for the full-width ideographic space.
- 2017-08-29
  - Fixed a bug.
- 2017-11-23
  - Fixed bugs.
  - Added `\SetBlockHeadingSpaces`.
  - Removed a space from `\contentsname` and `\indexname`.
- 2017-12-02
  - Fixed bugs.
- 2017-12-22
  - Improved JFM.
  - Change the way to detect `\label` between block headings.
  - Added chapter number to `\theequation`, `\thefigure`, `\thetable`.
- 2018-02-01

  - Sidenotes appears only odd pages in `tate` mode.
  - Added `\fnfixbottomtrue` for LuaLaTeX.
  - Added some options related to captions.
  - Extended `itemization_beforeafter_space`.
  - Fixed bugs.
- 2018-04-11
  - Sidenotes (`\footnote`) appears in the second column in `tate` mode.
  - Added options `begin_widh_(odd|even)_page` to `\DeclareBlockHeading`.
  - Changed `\labelenumi` as in `jarticle` etc.
  - Fix a bug on `column_gap` class option.
  - Added `mark_format` to `\DeclarePageStyle`.
- 2018-05-19
  - Made the width of the label in the table of contents longer.
  - Moved some macros to jlreq-helpers.sty
  - Fixed bugs.
- 2018-06-17
  - Gothic font is attached to font shape 'b'.
  - Fixed bugs.
- 2018-08-08
  - Added `nombre_ii` etc. to `\DeclarePageStyle`.
  - Fixed bugs.
  - Added `footnote_second_indent` and `endnote_second_indent` to `\jlreqsetup`.
- 2018-08-15
  - Fixed bugs.
- 2018-09-01
  - jlreq works with unusual `\mag`.
  - Fixed bugs.
- 2018-12-10
  - Added `number=[true/false]` to `\New***Heading`.
  - Added options for `\frontmatter` etc in `\jlreqsetup`.
  - Made `\jlreqHeadingLabel` etc available in `format` in `\NewTobiraHeading` and `\NewBlockHeading`.
  - Fixed bugs.
- 2019-01-15
  - Added `nombre_font` etc to `\NewPageStyle`. `font` is deprecated.
  - `format` without `#1` is allowed in `\NewBlockHeading`.
  - Extended `caption_label_format` etc. in `\jlreqsetup`.

  - Fixed bugs.
- 2019-04-01
  - Added `use_reverse_pagination` to the class option.
  - Stopped to use `zref` package.
  - New regnal year.
  - Fixed bugs.
- 2019-05-07
  - Added a small length to `\textwidth` and `\textheight`.
  - Changed the implementation of `running_head_ii` etc. in `\DeclarePageStyle`.
  - Fixed bugs.
- 2019-09-24
  - Deleted the (re-)definitions of `\@cite` and `\@biblabel`.
  - Added `\allowbreak` before block headings.
  - Fixed bugs.
- 2020-02-07
  - Changed the default value of `itemization_label_length` to `\leftmargini` etc.
  - Removed the redefinitions `\rmfamily` etc and added a code to `\@rmfamilyhook`.
  - Changed `\parskip` to `0pt`.
  - Fixed bugs.
- 2020-05-01
  - Added `theorem_label_font` and `theorem_font` to `\jlreqsetup`.
  - Fixed bugs.
- 2020-09-27
  - Added `*`-version of `\tatechuyoko`.
  - Fixed bugs.
- 2020-12-29
  - `fontsize` etc. with LuaLaTeX accept `H`.
  - Added `\jidori`.
  - Fixed bugs.
- 2021-03-17
  - Use the pagestyle `plain` at `\maketitle` if the current one is not `empty`
  - Removed JFM glue after `\item`.
  - Removed JFM glue after block headings.
  - Fixed bugs.
- 2021-05-28
  - Extended `caption_align` in `\jlreqsetup`.
  - Removed some `\ifthenelse`.

- 2021-07-22
  - Stopped to use `\IfHookExistsTF`.
  - Added `\akigumi`.
  - Stopped to load packages `xkeyval` and `ifthen`.
  - It has more compatibility with expl3.
  - Added `pagestyle` to `\DeclareBlockHeading`.
  - Fixed bugs.
- 2021-07-25
  - Load `ifthen` again. (Only for Re:VIEW, will be removed in future.)
  - Fixed bugs.
- 2021-08-12
  - Removed the direct dependence on `etoolbox` package.
  - Fixed a bug.
- 2021-10-09
  - Fixed bugs.
- 2021-11-05
  - `paper=b*` is regarded as a ISO series.
  - Removed many codes relating with LaTeX hooks mechanism (because it seems not stable.)
  - Removed `\RequirePackage{ifthen}`.
  - Removed `\kcatcode`settings with upLaTeX.
- 2022-04-05
  - Added `warichu_opening` and `warichu_closing` to `\jlreqsetup`.
  - Change a little bit penalties around block heading.
  - Fixed a bug: `\selectfont` after `\DeclareFontShape` raised an error.
  - Fixed a bug: `use_reverse_pagination` did not work.
  - Fixed a bug: A second running head disappeared sometimes.
  - Rewrote `\DeclarePageStyle`.
  - Deleted `\@makefntext`, define `\@makefntext` directly.
  - Fixed other bugs.
- 2022-04-11
  - Fixed a bug.
- 2022-07-13
  - Fixed a bug: The position of running heads were not correct.
- 2022-11-28
  - Fixed a bug: did not register to running head when `\SetBlockHeadingSpaces` is used.
  - Fixed a bug: wrong papersize for ISO C4.

  - Added a package `jlreq-complements`
  - Fixed some other bugs and adjust with some other packages.
- 2023-03-05
  - Fixed a bug on cutin headings.
- 2023-06-19
  - Stopped to load ‘everyhook' package LuaLaTeX (it was not compatible with the document).
  - Fixed a bug: a space before ‘enumerate' environment was not inserted sometimes.
  - Fixed a bug on `use_reverse_pagination`.
- 2024-02-13
  - Added some `\par` (for hook system in LaTeX kernel)
  - Added `tableofcontents_twocolumn` and `abstract_with_maketitle` to `\jlreqsetup`.
  - Deleted `\PushPostHook`.
  - Some modifications of jfm.
  - Fixed a bug: heading command may have an infinite loop.
  - Load `stfloats` with LuaLaTeX.
  - Improved position adjustment in pagestyle.
  - Some other improvements etc.
- 2024-02-16
  - Fixed a bug.
- 2024-08-23
  - Fixed bugs.
  - Changed the name of some internal variables.
  - Added the usage of this class to English document.
  - Use `\DeclareKeys` etc.
  - Changed the internal processing of block headings a little.
- 2024-09-26
  - Removed pattern matchings from Makefile.
- 2024-10-04
  - Fixed: tfm files are mixed into the list of vf files in Makefile.

Noriyuki Abe https://github.com/abenori/jlreq