The udiss bundle

TUGINDIA

Version 0.1—October 1, 2024

https://ctan.org/pkg/udiss
https://puszcza.gnu.org.ua/bugs/?group=udiss

Contents

---

1 Introduction

The udiss bundle is a LaTeX-class-file developed to assist students in typesetting their *u*niversity *diss*ertations. It is a collection of multiple support files. This is the first experimental version. We need feedback and comments from the community to make it better. Currently, nothing should be considered 'stable'. Soon after getting comments and reviews from the community, we will move towards a stable version.

Universities often have strict requirements regarding the formatting of the dissertations/theses submitted to them. This bundle pre-supplies a generic style (university-agnostic) for creating dissertations. It can be loaded simply with:

```
\documentclass{udiss}
```

But, apart from the generic style, it also supplies university or department specific files for some selected institutions. Each specialised class file comes with an identifier. E.g., if you want to use the template for the Kerala university, you may use the `ukerala` tag and load its template like in the following:

```
\documentclass{udiss}
\dissertationstyle{ukerala}
```

We are willing to expand. If you want your institution to be supported by us, please have a look at the instructions given in section 3. If you want to develop your own style, please have a look at the developer's instruction given in section 2.3. If you want to get your class published under this bundle, please have a look at the contribution guidelines given in section 2.4.

This document explains the user interface as well as the implementation of the class. It has three major parts. The first one contains the documentation of the class (section 2), the second one contains the developer's manual (section 2.3) and the third one contains the implementation of it (section 4).

Before getting started with the documentation, just a brief note about our approach. As of version 0.1, we are planning to have a radically all-modern and backwards-blind approach. This is a Unicode-only package, meaning, it won't work with the classic PDFLaTeX. We load the modern, well maintaned and inclusive implementation of the Latin Modern family, i.e., the New Computer Modern family (ctan.org/pkg/newcm). For using/testing the class, please use the latest TeX Live (all packages updated, as of 2024-09-27). Both XeLaTeX and LuaLaTeX supported, but the latter is recommended as it identifies scripts without markup.

2 Documentation

This part of the document describes everything that a normal user of this class should know. It enlists all the features of the class.

2.1 User manual

Since this is a class that typesets dissertations, it requires some basic information which is provided using the following macros.

We provide the usual LaTeX-like macros for the convenience, but these macros have *key-value*-equivalents as well.

`\title`	`\title{`⟨*the-title-of-the-dissertation*⟩`}` ⚠
`\author`	`\author{`⟨*name-of-the-author*⟩`}` ⚠
`\pronouns`	`\pronouns{`⟨*the-pronouns-of-the-author*⟩`}`

Like in most standard LaTeX classes, `\title` command saves the title of the dissertation from its argument. This is one of the commands which we mark as *strict*. In the manual, all such commands are marked with a ⚠ symbol at the right end. If such commands (or their parametric equivalents as shown in the following paragraph) are not provided, the compilation is interrupted with an error. The `\author` command doesn't return any error if the class option `blind` is loaded. Since the `blind` option is used for anonymous submissions, the author information is irrelevant anyways and hence, is ignored.

These macros are available as class options too. One just has to use `title={`⟨*title*⟩`}` in the optional argument of the `\documentclass` command. Similarly, `author` and `pronouns` also are available.

`\subtitle`	`\subtitle{`⟨*the-subtitle-of-the-dissertation*⟩`}`
`\shorttitle`	`\shorttitle{`⟨*a-short-title-for-headers*⟩`}`

Dissertations often have a subtitle too, but it is not an absolute requirement, so optionally, one may want to use `\subtitle` command. Sometimes the title and the subtitle are too long for the running header. One may provide a short title with `\shorttitle` for this purpose. Depending on the selected template and options, it will be printed.

These commands are also available as class options `subtitle` and `shorttitle`.

`\stream`	`\stream{`⟨*the-name-of-the-stream*⟩`}` ⚠
`\discipline`	`\discipline{`⟨*the-name-of-the-discipline*⟩`}` ⚠
`\degree`	`\degree{`⟨*the-name-of-the-degree*⟩`}` ⚠

These details are used at several places in the class. E.g., the declaration, the 'fulfilment' statement printed on the first page. Like most other macros, these too have straight forward *key-value*-equivalents `stream`, `discipline` and `degree`.

`\supervisor`	`\supervisor{`⟨*name-of-the-supervisor*⟩`}` ⚠
`\university`	`\university{`⟨*name-of-the-university*⟩`}` ⚠
`\department`	`\department{`⟨*name-of-the-department*⟩`}` ⚠

These commands are not seen in other standard LaTeX-classes, but `udiss` requires them obligatorily (only except in the `blind` mode).

`supervisor`, `university` and `department` are the *key-value*-equivalents.

`\logo`	`\logo{`⟨*file-name*⟩`}`
`\logo*`	`\logo*{`⟨*TEX-code*⟩`}`

Some departments mandate printing the university logos in the dissertation. If this command is used, the logo is printed. The location of the logo can be customised as per the requirement. The class provides options for that.

If you are mad about TeX and have a code for printing the logo of your institute, use this command with a star and in its argument, paste your code to get it printed at an appropriate place.

We have `logo` and `logo*` as the *key-value*-equivalents.

language	language={⟨*babel-language-names*⟩}	
ldfbabel	ldfbabel={⟨*babel-ldf-language-names*⟩}	
nofontwarning	nofontwarning={⟨*truth-value*⟩}	true | false

The babel package has a complex mechanism for loading languages. The complex design has its own advantages, especially with regards to backwards compatibility. For detailed explanation, it's best to go through the user manual of babel, or its official online documentation, especially the web-page at: https://latex3.github.io/babel/guides/which-method-for-which-language.html.

To help those who are completely at sea, we briefly will have a look at the two mechanisms babel provides. One is the original method of loading the language-name(s) as class option(s), so something like:

```
\usepackage[italian,hindi,english]{babel}
```

The last one from the list is understood as the 'main' language of the document. These languages are defined using the so-called `.ldf`-files. A lot of support for some languages is available using this mechanism. For some European languages, e.g., French, the difference between the two mechanisms is too sharp and it has been found that a lot of native speakers still prefer the old, stable way of using `.ldf`-files. If you are one of them and you are used to loading language-names as options to package babel, you should use the `ldfbabel` option of udiss. But we advise against it.

The second approach provided by babel for loading languages is as follows:

```
\babelprovide{⟨language-name⟩}
```

This method follows and uses the Unicode Common Locale Data Repository (CLDR) for creating the resources. This data is saved in `.ini` files. This is a more modern approach, in our opinion. But it may cost us a little bit of inconvenience since it doesn't support the common short-hands which are provided by the `.ldf` files. But short-hands make the input more and more TEX-dependent and thus less productive, so we don't endorse it. That is why the more intuitive name for language-options in this class is for the `.ini`-based approach. We load the `english` language by default if no other is loaded. For this too, we go with the `.ini` file developed for English.

The syntax of both these options is exactly like the one provided by babel. One may provide a comma-separated list and the last one of them will be set as the main language of the document. Have a look at section 2.3 for more details regarding this.

We also provide some auxiliary TEX files with the class for setting the font-families as well as the language-specifics as required by babel. If a file like that it is not found, we issue a warning. If the users want that warning to be silent, they can use the option `nofontwarning`.

license	license={⟨*all-rights-reserved/CC-BY-SA/GFDL*⟩}	all-rights-reserved
copyright-yrs	copyright-yrs={⟨*copyrightable-years*⟩}	\the\year
localcopyright	localcopyright={⟨*the-word-for-copyright*⟩}	Copyright

This is an option to set a license for the dissertation. We encourage users to use free culture licenses. Currently we support two popular licenses, i.e., 'CC-BY-SA 4.0+ International' and 'GFDL v1.3+'. We don't set any license by default as we believe making a work 'free' (as in freedom) should be the decision of the author and hence 'all rights are reserved' by default. If you are not using the `blind` package option, your output will have a copyright notice with your (author's) name and the current year as the copyrightable year in the metadata of the PDF. If you are wondering why, the `blind` option is for preserving the anonymity of the author and the copyright-notices require a copyright-holder by default, thus with `blind` option, we have disabled choosing licenses. If you want to change the copyrightable year(s), you may use the option `copyright-yrs`. Since the package is supposed to support multilingual typesetting, we have provided an option to change the word 'Copyright' to the one in the main language of your document.

placeholders	placeholders	

If you want to see how the things so far discussed come together, you may just load this option. It places placeholders for all the options discussed for creating a sample document.

lot	lot={⟨*truth-value*⟩}	true | false
lof	lof={⟨*truth-value*⟩}	true | false
declaration	declaration={⟨*truth-value*⟩}	true | false
declarationtxt		
decllocal		

By default, the class only prints the table of contents. If the author requires the list of figures (`lof`) or tables (`lot`), they may use the respective option. These are boolean keys which take either `true` or `false` as their values. Not passing any truth-value to a `key` is equivalent to using `key=true`. Similarly, many universities require a declaration from the candidate regarding the originality of the work. We have provided a sample draft of the declaration in a renewable macro, but it is not enabled by default. After getting comments from the community, we will decide what should be done with it. Note that the default value for all these keys is `false`[1]. Declaration text can be edited using `declarationtxt` option. The local word for declaration is stored in `decllocal` (defaulting to 'Declaration').

blind	blind={⟨*truth-value*⟩}	true | false

At times, anonymous submissions are required where the name of the author and supervisor is hidden.

framed-title	framed-title={⟨*truth-value*⟩}	true | false

By default the title-page is very minimal. If a user wants to have a frame around the title-page, they may use this option.

[1] Throughout the documentation, at the right corner, we enlist the possible values to a key and the default one is marked in red.

print
norefcolors
colorprint
udisslinkclr
udissurlclr
udissciteclr

`print={`⟨*truth-value*⟩`}` `true` | `false`
`norefcolors={`⟨*truth-value*⟩`}` `true` | `false`
`udisslinkclr={`⟨*link-colour*⟩`}` `udisslink`
`udissurlclr={`⟨*url-colour*⟩`}` `udissurl`
`udissciteclr={`⟨*cite-colour*⟩`}` `udisscite`

Many departments require the submissions to be made in the print form as well. For that, `print` is a boolean option. It creates outputs suitable for the print submissions. The class, by default, does not use `print` option. The default output of the class is suitable for digital submissions. The following settings are always turned on by the class in normal circumstances:

- The hyperref package is loaded.
- The xcolor package is loaded. Some colours for the internal cross-references are pre-selected.

If you don't want colours for the hyperlinks in your digital document, you may use the `norefcolors` option. If you don't like the pre-defined colour-scheme given by us for the digital documents, you may use the class options specifically developed for changing them. They are `udisslinkclr`, `udissurlclr` and `udisscitеclr`. The arguments of these options are either the pre-defined colours by package `xcolor` or user-defined colours with the `\colorlet` or `\definecolor` commands. The default colour-scheme used by us is as seen in the macro explanation above.

More importantly though, if a PDF with colours is given for printing and the printer is using the black and white mode, then the coloured words look faded in the print out. To avoid this, we load the xcolor package only when the `print` option is *not* loaded (basically always, by default). For performance reasons, we don't load the `hyperref` package when `print` is loaded. Hyper-references make sense only in a digital PDF, so to optimise PDF compilation, in `print` mode, we don't load `hyperref`.

Note that if you are using the `license` option with `CC-BY-SA`, the doclicense package `\Required` for printing the license automatically loads the hyperref package. We don't have any control over it and hence, you will see the hyperlink boxes around the CC-text. You can safely ignore them as the text is still in black colour and your objective is anyways just to *print* the output.

oldstylenumoff `oldstylenumoff={⟨truth-value⟩}` `true | false`
oldone `oldone={⟨truth-value⟩}` `true | false`

In text, the old style numbers look better. The numbers in the current document are old style. Lining numbers, on the other hand, don't have ascenders and descenders.

If you prefer the lining numbers, you may turn this option on and all the fonts will start showing lining numbers. Note that printing of old style numbers also depends upon whether the font has old style numbers or not. The relevant settings are added to the font automatically, but while selecting the font, first make sure whether the old style table is present or not.

Apart from that, the New Computer Modern font family provides an old-style shape for the number 1 (this exact shape!), but this shape is provided as a character variant and character variants cannot be loaded globally, since various fonts may have various variants at that slot, hence if someone wants this shape, they will have to use the `oldone` option. Without this option the number looks like as seen on the first line and that's the default style of udiss.

0123456789 `Old style with default 1`

0123456789 `Old style with the old 1`

0123456789 `Lining`

fulfilment `fulfilment={⟨the-titlepage-sentence-about-course-requirement⟩}`

The title-page prints a sentence which says:

> A dissertation submitted in the partial fulfilment of the requirements for the `\udiss@degree` program in `\udiss@discipline`.

In order to change it, one may use this package option. If you want to use the internal macros for univesity and department names, don't forget to use `\makeatletter` and `\makeatother`.

As described in the introduction (section 1), the users can choose to load the generic style or a specialised style. For that, we have the following macro:

`\dissertationstyle` `\dissertationstyle{⟨style-for-dissertation⟩}`

With this command, you may select the institute or the style. It will set certain defaults.

We provide limited support as of v0.1 (only for university of Kerala). We wish to expand more. Send us support requests for adding your institute to the list.

2.2 Multilingual typesetting

The udiss class aims at supporting multilingual typesetting by default. We load the babel package for linguistic support, fontspec and unicode-math package for loading fonts. It's the best to not load them again. For loading languages, it's best to use the package internal mechanism. If you find something missing in the mechanism, you may load babel or fontspec commands directly. Please consider reporting a missing feature so that we can implement it in the future versions.

We will see two examples of loading languages.

2.3 Developer's manual

Only if you sure about what you are doing

As the section title and the double bend suggests, this area is for advanced customisation. Only enter it if you have sufficient confidence to avoid errors.

2.3.1 Fonts

We are a Unicode-only class, so we need to take care of three major font families, i.e., Roman, `Typewriter` and Sans (stylisation intentional, for demonstration). The fontspec package takes care of Unicode fonts in LaTeX, but we, additionally, load package unicode-math too, to set math-fonts. Even though, this much is practically enough for font-selection, we go with the babel's mechanism of loading language-specific fonts, so that the fonts don't apply globally. Our class already takes care of most of the requirements regarding font, but if you want to tinker with the defaults, you are advised to read this section. The following parameters are for you.

explicitext	explicitext={⟨*truth-value*⟩}	true | false
ignorefsoff	ignorefsoff={⟨*truth-value*⟩}	true | false

Without this option, one cannot use the `Extension` parameter of package fontspec/unicode-math. We have some options later which set the extension for `rm`, `sf` and `tt` families. They are also useless if this key is not used. A word of caution: loading explicit extensions for fonts is restrictive. Suppose you want to use KpRoman-Regular as your normal font, but, say, FreeSerifBold as your bold font (please don't, this is solely for exemplification :P), loading an explicit extension will cause errors as the former is an OpenType font and the latter is a TrueType one. So if you are 200% sure that all the fonts you need use one particular extension *only*, then use this option. Otherwise, you are better off without it.

The TeX-tree contains many files having the `.fontspec` extension. These files contain predefined settings for fonts which the users may not always want. Thus, we ignore these files by default. If you are very sure that you want to load `.fontspec` files, you may turn our conditional off and the `.fontspec`-files, will be loaded correctly.

rmfeatures	rmfeatures={⟨*font-features-for-serif-fonts*⟩}
sffeatures	sffeatures={⟨*font-features-for-sans-fonts*⟩}
ttfeatures	ttfeatures={⟨*font-features-for-mono-fonts*⟩}
mathfeatures	mathfeatures={⟨*font-features-for-math-fonts*⟩}

We have `udiss`-options for most commonly used optional parameters of fonts, still there are a lot of them available which we didn't consider important for the purpose of this class, e.g., `Scale`, `Color`. If you want to use these features with your fonts, you may use these four options as per your need. The value of this key is directly injected in the font-loading commands.

rmfntext	rmfntext={⟨*font-extension-for-serif-fonts*⟩}	ttf | otf
sffntext	sffntext={⟨*font-extension-for-sans-fonts*⟩}	ttf | otf
ttfntext	ttfntext={⟨*font-extension-for-mono-fonts*⟩}	ttf | otf
mathfntext	mathfntext={⟨*font-extension-for-math-fonts*⟩}	ttf | otf

With these four options, one may choose either `ttf` or `otf` extension for a particular font family. As mentioned before, it applies to all the fonts loaded in that family. Note that these options are only used when the `explicitext` option is active. This is why there is no default value set. By default this explicit extension method itself is not used.

`rmfont` NewCM10-Book	`sffont` NewCMSans10-Book	`ttfont` NewCMMono10-Book
`rmbffont` NewCM10-Bold	`sfbffont` NewCMSans10-Bold	`ttbffont` NewCMMono10-Bold
`rmitfont` NewCM10-BookItalic	`sfitfont` NewCMSans10-BookItalic	`ttitfont` NewCMMono10-BookItalic
`rmbfitfont` NewCM10-BoldItalic	`sfbfitfont` NewCMSans10-BoldOblique	`ttbfitfont` NewCMMono10-BoldOblique
`rmslfont` NewCM10-Book	`sfslfont` NewCMSans10-BookOblique	`ttslfont` NewCMMono10-Book
`rmbfslfont` NewCM10-Bold	`sfbfslfont` NewCMSans10-BoldOblique	`ttbfslfont` NewCMMono10-BoldOblique
`mathfont` NewCMMath-Book		
`mathbffont` NewCMMath-Bold		

Table 1: Default fonts for English language (all otfs)

If you want to know the default fonts we load always with udiss, you may refer to table 1.

2.3.2 Title-page customisation

The title-page prints several fields and very often authors are very picky about it. Most of the things we offer in it are customisable. One may change the following things for all the major fields on the title-page.

1. The font
2. The font-size
3. The font-color
4. The font-shape (e.g., `\itshape`, `\scshape` etc.)
5. Additional features to the font loaded

All the parameters (along with their default values) are listed in table 2. The table uses '-' when there is no change in the default, only the fields having a special value are marked.

`logowidth`	`logowidth={`⟨*logo-width-proportion*⟩`}`	0.3
`logoheight`	`logoheight={`⟨*logo-height-proportion*⟩`}`	0.075

By default the width of the logo is 0.3×`\linewidth` and 0.075×`\textheight`. If you want to change these default numbers, you may use these parameters.

Title	- `titlefont`	`\bfseries` `titleshape`	huge `titlesize`	black `titlecolor`	- `titleffeat`
Subtitle	- `subtitlefont`	`\normalfont` `subtitleshape`	large `subtitlesize`	black `subtitlecolor`	- `subtitleffeat`
Author	- `authorfont`	`\normalfont` `authorshape`	large `authorsize`	black `authorcolor`	- `authorffeat`
University	- `unifont`	`\normalfont` `unishape`	`\normalsize` `unisize`	black `unicolor`	- `uniffeat`
Department	- `deptfont`	`\normalfont` `deptshape`	`small` `deptsize`	black `deptcolor`	- `deptffeat`
Fulfilment	- `fffont`	`\normalfont` `ffshape`	small `ffsize`	black `ffcolor`	- `fffeat`

Table 2: Customising title-page

2.4 Contributing to the bundle

Only if you are ready to bear with us :P

We welcome contributions to our code. Our code is hosted at Puszcza, a free-software-based web-space. But for consistency, we request our supporters to write the code using 60 characters per line limit. Try to adhere to the coding conventions. If there is a slight violation here and there in your patches, please don't mind us editing your code first and then merging it in the main branch.

We release our code with GPLv3+ and the documentation with GFDLv1.3+. Your patches also will have to comply with the conditions of these licenses.

If you wish to document your patches, please avoid gendered language. Use *they/them* pronouns when referring to users.

Apart from university-styles, we also need support for language and fonts. If you know free/libre Unicode fonts for your language, please release them on CTAN or let us know about them, so that we will release using our templates. Provide appropriate translations for the fields provided in class.

The class comes with a test-suit. You are advised to run `l3build check` command before pushing a patch. This way we can be sure that the patch doesn't break any of the already existing features. If you think something was buggy in the original design and you got a fix for that, kindly run `l3build save` on the concerned test file and push the changed `.lvt` and `.tlg`s also.

3 Requests for new templates

We are open to your support-requests too. Just let us know the norms and conventions of your institution. Let us know which fields you would like to have pre-filled in the style file. If you require the logo to be printed on the title-page, send it to us in a decent resolution. We will include it in our bundle.

4 Implementation

In this section, we explain how this program was developed. We discuss the tools used by us for developing the features offered in this class. The features can be seen in the left margin hyperlinked to their respective place in the documentation section section 2.

We first declare the tag used by the `.ins` file for stripping the source-code of the generic class. We will have several tags for generating the other class-files, but for the convenience of maintenance, we will have all the code in this central `.dtx` file.

```
1 ⟨*class⟩
```

First we declare the class with some elementary information.

```
2 \ProvidesClass{udiss}[2024/10/01 v0.1
3   A LaTeX bundle for typesetting dissertations (TUGIndia)]
```

We develop the options for the class using `\ekvdefinekeys` command. This command is provided by the **expkv|DEF** package, whereas for creating class/package options, we use the **expkv|OPT** package. For performance reasons, we don't use the entire **expkv|BUNDLE**.

```
4 \RequirePackage{expkv-def,expkv-opt}
```

`\pronouns` `pronouns`

Mostly we have implemented the macros in the same order as they are documented, but the exception is of the first two, i.e., `\title` and `\author`. These commands are developed by the standard classes too and hence, after loading the class (memoir in our case) their definitions are rewritten. Thus these two macros are developed a little late in the code. Rest is still in line with the documentation. The method of saving the values of the macros is an indirect one. We use internal macros (e.g., `\udiss@title`) for saving them.

We start with the `pronouns` field. Since it is optional, we don't need any errors here. At each step, we set some if-conditionals. All of them follow the standard convention of prefixing the class/package name and the `@` symbol for internalising. Have a look at the following two conditionals which we use immediately while defining the `pronouns` field:

```
5 \newif\ifudiss@pronouns
6 \newif\ifudiss@blind
```

We use the xparse-syntax for developing user-side macros. First we check if the class option `blind` is loaded. If it is so, there is no point in saving `pronouns`, hence it is skipped. The following code does that:

```
7 \NewDocumentCommand{ \pronouns }{ m }{%
8   \ifudiss@blind
9   \else
10    \def\udiss@pronouns{#1}%
11  \fi
12 }
```

As mentioned before, we use **expkv|DEF** and **expkv|OPT** packages for developing the *key-values*. The former proives a command `\ekvdefinekeys` that defines all the *key*s. We could have called it just once in the code and developed all the *key*s at one go, but in order to keep the implementation in line with the documentation and for a generally better readable code, we segment it in several pieces and develop the respective options with different calls of `\ekvdefinekeys`. Note that the set used for all of them is still the same, i.e., `udiss`. Storing a value passed to a key in a specific macro for later use is quite straight-forward in **expkv|DEF**, but since we need to set some macros conditionally, we

will use the `code` *type* of expkv. The first non-space-token followed by the invocation of this type is the name of the *key*. Its argument is an arbitrary TEX-code where we can use have the usual TEX-argument-call, i.e., `#1`.

```
\ekvdefinekeys{udiss}{%
  code    pronouns       = {%
    \ifudiss@blind
    \else
      \def\udiss@pronouns{(#1)}%
    \fi
    \udiss@pronounstrue
  }%
}
```

(*End of definition for* `\pronouns` *and* `pronouns`*. These functions are documented on page 3.*)

The `\ifudiss@blind` conditional was developed in order to conditionally save (and print) some information which identifies the author. If a user loads the option `blind`, they don't need that information to be printed and since it is not supposed to be printed, why to even save it? Thus we save it only when needed.

`title` `author`

Since we don't allow blank arguments for *strict* commands/parameters, we check their existence there and there and generate an error if found blank. Since we need to error twice (once with the macro, once with the parameter, we save the code for generating the error in a handy internal macro. Let's have a look at one developed for the `title` field.

```
\newif\ifudiss@title
\def\udiss@blank@title@err{%
  \ClassError{udiss}{%
    The `title' field cannot be blank%
  }{%
    You have provided an empty argument to the `title'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
```

We use the same method for all the other *strict* commands too.

```
\newif\ifudiss@author
\def\udiss@blank@author@err{%
  \ClassError{udiss}{%
    The `author' field cannot be blank%
  }{%
    You have provided an empty argument to the `author'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
```

Now we start developing the code for the `title` and `author` parameters. In order to check whether a macro is empty or not, we need to use the `\ifx\macro`$_1$`\macro`$_2$ construct. But since we have an argument and not a ready macro for comparison, we first we develop an internal macro with the argument as its value.

```
\ekvdefinekeys{udiss}{%
  code    title          = {%
    \def\udiss@tmp@title{#1}%
```

The true branch of this conditional, i.e., everything till either `\else` or `\fi` is found, defines the actual macro to be used while typesetting, i.e., `\udiss@title`, but since the user has provided a blank macro, we give a cryptic value to the macro, e.g., `-NoValidTitle-`. Note that this gets printed in the document even though the user gets an error. Since we use this regularly, a user may search for `-NoValid` in the output to find out all the values they missed providing.

```
    \ifx\udiss@tmp@title\empty
      \def\udiss@title{-NoValidTitle-}%
```

After this, we print the error for the blank argument. In the `\else` branch (which means the user did *not* pass a blank argument, but provided a legit value), we define the command with the actual user-given-value. This is the most ideal condition.

```
      \udiss@blank@title@err
    \else
      \def\udiss@title{#1}%
    \fi
  },%
```

We repeat the same pattern for the *strict* commands now. Here is the code for `author`.

```
  code    author          = {%
    \def\udiss@tmp@author{#1}%
    \ifx\udiss@tmp@author\empty
      \def\udiss@author{-NoValidAuthor-}%
      \udiss@blank@author@err
    \else
      \def\udiss@author{#1}%
    \fi
  }%
}
```

(*End of definition for* `title` *and* `author`*. These functions are documented on page 3.*)

subtitle
\subtitle
shorttitle
\shorttitle

Here is the code for the `subtitle` and `shorttitle` fields. As they aren't *strict*, no errors needed for them. We need a conditional though. We set it to true so that it is understood that this field was not forgotten by the author. Only if their value is true, we print them later.

```
\newif\ifudiss@subtitle
\newif\ifudiss@shorttitle
\ekvdefinekeys{udiss}{%
  code    subtitle        = {%
    \IfBlankTF{#1}{%
      \def\udiss@subtitle{-NoValidSubtitle-}%
      \udiss@subtitletrue
    }{%
      \def\udiss@subtitle{#1}%
      \udiss@subtitletrue
    }%
  },%
  code    shorttitle      = {%
    \IfBlankTF{#1}{%
      \def\udiss@shorttitle{-NoValidSubtitle-}%
      \udiss@shorttitletrue
```

```
    }{%
      \def\udiss@shorttitle{#1}%
      \udiss@shorttitletrue
    }%
  }%
}
\NewDocumentCommand{ \subtitle }{ m }{%
  \def\udiss@subtitle{#1}%
  \udiss@subtitletrue
}
\NewDocumentCommand{ \shorttitle }{ m }{%
  \def\udiss@shorttitle{#1}%
  \udiss@shorttitletrue
}
```

(*End of definition for* `subtitle` *and others. These functions are documented on page* 3.)

stream
\stream
discipline
\discipline
degree
\degree

Again some compulsory fields, so we develop the errors, then the *key*s and then the macros.

```
\newif\ifudiss@stream
\def\udiss@blank@stream@err{%
  \ClassError{udiss}{%
    The `stream' field cannot be blank%
  }{%
    You have provided an empty argument to the `stream'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\newif\ifudiss@discipline
\def\udiss@blank@discipline@err{%
  \ClassError{udiss}{%
    The `discipline' field cannot be blank%
  }{%
    You have provided an empty argument to the `discipline'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\newif\ifudiss@degree
\def\udiss@blank@degree@err{%
  \ClassError{udiss}{%
    The `degree' field cannot be blank%
  }{%
    You have provided an empty argument to the `degree'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\ekvdefinekeys{udiss}{%
  code    stream         = {%
    \def\udiss@tmp@stream{#1}%
    \ifx\udiss@tmp@stream\empty
      \def\udiss@stream{-NoValidStream-}%
      \udiss@blank@stream@err
    \else
      \def\udiss@stream{#1}%
```

```
    \fi
  },%
  code    discipline     = {%
    \def\udiss@tmp@discipline{#1}%
    \ifx\udiss@tmp@discipline\empty
      \def\udiss@discipline{-NoValidDiscipline-}%
      \udiss@blank@discipline@err
    \else
      \def\udiss@discipline{#1}%
    \fi
  },%
  code    degree         = {%
    \def\udiss@tmp@degree{#1}%
    \ifx\udiss@tmp@degree\empty
      \def\udiss@degree{-NoValidDegree-}%
      \udiss@blank@degree@err
    \else
      \def\udiss@degree{#1}%
    \fi
  }%
}
\NewDocumentCommand{ \stream }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@stream@err
    \def\udiss@stream{-NoValidStream-}%
  }{%
    \def\udiss@stream{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@streamtrue
  }{%
    \def\udiss@stream{#1}%
    \udiss@streamtrue
  }%
}
\NewDocumentCommand{ \discipline }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@discipline@err
    \def\udiss@discipline{-NoValidDiscipline-}%
  }{%
    \def\udiss@discipline{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@disciplinetrue
  }{%
    \def\udiss@discipline{#1}%
    \udiss@disciplinetrue
  }%
}
\NewDocumentCommand{ \degree }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@degree@err
    \def\udiss@degree{-NoValidDegree-}%
  }{%
```

```
    \def\udiss@degree{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@degreetrue
  }{%
    \def\udiss@degree{#1}%
    \udiss@degreetrue
  }%
}
```

(*End of definition for* `stream` *and others. These functions are documented on page 3.*)

We then develop the last set of *strict* macros which are used in the declaration and some on the title-page too. The method is no different.

supervisor
\supervisor
university
\university
department
\department

```
\newif\ifudiss@supervisor
\newif\ifudiss@university
\newif\ifudiss@department
\def\udiss@blank@supervisor@err{%
  \ClassError{udiss}{%
    The `supervisor' field cannot be blank%
  }{%
    You have provided an empty argument to the `supervisor'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\def\udiss@blank@university@err{%
  \ClassError{udiss}{%
    The `university' field cannot be blank%
  }{%
    You have provided an empty argument to the `university'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\def\udiss@blank@department@err{%
  \ClassError{udiss}{%
    The `department' field cannot be blank%
  }{%
    You have provided an empty argument to the `department'
    field. This is not\MessageBreak permitted by the class.%
  }%
}
\ekvdefinekeys{udiss}{%
  code    supervisor         = {%
    \def\udiss@tmp@supervisor{#1}%
    \ifx\udiss@tmp@supervisor\empty
      \def\udiss@supervisor{-NoValidSupervisor-}%
      \udiss@blank@supervisor@err
    \else
      \def\udiss@supervisor{#1}%
    \fi
  },%
  code    university         = {%
    \def\udiss@tmp@university{#1}%
```

```
    \ifx\udiss@tmp@university\empty
      \def\udiss@university{-NoValidUniversity-}%
      \udiss@blank@university@err
    \else
      \def\udiss@university{#1}%
    \fi
  },%
  code    department         = {%
    \def\udiss@tmp@department{#1}%
    \ifx\udiss@tmp@department\empty
      \def\udiss@department{-NoValidDepartment-}%
      \udiss@blank@department@err
    \else
      \def\udiss@department{#1}%
    \fi
  }%
}
\NewDocumentCommand{ \supervisor }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@supervisor@err
    \def\udiss@supervisor{-NoValidSupervisor-}%
  }{%
    \def\udiss@supervisor{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@supervisortrue
  }{%
    \def\udiss@supervisor{#1}%
    \udiss@supervisortrue
  }%
}
\NewDocumentCommand{ \university }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@university@err
    \def\udiss@author{-NoValidUniversity-}%
  }{%
    \def\udiss@university{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@universitytrue
  }{%
    \def\udiss@university{#1}%
    \udiss@universitytrue
  }%
}
\NewDocumentCommand{ \department }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@department@err
    \def\udiss@author{-NoValidDepartment-}%
  }{%
    \def\udiss@department{#1}%
  }%
  \IfBlankTF{#1}{%
    \udiss@departmenttrue
```

```
  }{%
    \def\udiss@department{#1}%
    \udiss@departmenttrue
  }%
}
```

(*End of definition for* `supervisor` *and others. These functions are documented on page 3.*)

`\logo` `\logo*` We develop the `\logo` command with a starred variant using the intuitive xparse trickery. Notice the use of two different conditionals `\ifudiss@logo` and `\ifudiss@texlogo`.

```
\newif\ifudiss@logo
\newif\ifudiss@texlogo
\NewDocumentCommand{ \logo }{ s +m }{%
  \IfBooleanTF{#1}{%
    \def\udiss@texlogo{#2}%
    \udiss@texlogotrue
  }{%
    \def\udiss@logo{#2}%
  }%
  \udiss@logotrue
}
```

(*End of definition for* `\logo` *and* `\logo*`*. These functions are documented on page 3.*)

`logo` `logo*` expkv doesn't have a way to have starred variants of the keys, but we can always create a different key with an additional star. That's what we do here.

```
\ekvdefinekeys{udiss}{%
  code    logo           = {%
    \def\udiss@logo{#1}%
    \udiss@logotrue
  },%
  code    logo*          = {%
    \def\udiss@texlogo{#1}%
    \udiss@logotrue
    \udiss@texlogotrue
  }%
}
```

(*End of definition for* `logo` *and* `logo*`*. These functions are documented on page 3.*)

Now we move to the complicated task of developing the mechanism for loading languages. In order to imitate the syntax of `babel`, we need a parser which inteprets comma-separated language-names. We develop a new counter called `udiss@langbabel` and use it with `\alph` inside `\csname` in order to produce an ordered sequence of macros that store the different language-names. After parsing, the language macros should look like `\udiss@langbabel@a`, `\udiss@langbabel@a` etc. The following code does that. The `\udiss@lgbbl` macro is the internal parser for the language-names.

```
\newcounter{udiss@langbabel}
\stepcounter{udiss@langbabel}
\def\udiss@int@langbabel#1{%
  \expandafter\edef
    \csname
      udiss@langbabel@\alph{udiss@langbabel}%
    \endcsname{#1}%
```

```
  \stepcounter{udiss@langbabel}%
}
\NewDocumentCommand\udiss@lgbbl{ >{ \SplitList{,} } m }{%
  \ProcessList{#1}{\udiss@int@langbabel}%
}
```

language
ldfbabel
nofontwarning

We first create the if-conditionals required for this job.

```
\newif\ifudiss@lg@used
\newif\ifudiss@ldfbabel
```

The `language` key takes the comma-separated list as its argument and passes the list to the `\udiss@lgbbl` macro which we just saw. So whenever we pass multiple languages to the option `language`, it is parsed by `\udiss@lgbbl`. We then turn the `\ifudiss@lg@used` conditional on, so that when neither of these options are used by the user, we can do something.

```
\ekvdefinekeys{udiss}{%
  code    language       = {%
    \udiss@lgbbl{#1}%
    \udiss@lg@usedtrue
  },%
```

On the other hand though, the process is quite simple and straight forward. We need to store the list of the `.ldf`-languages in a macro and later pass it to the babel package as babel already understands how to parse it. We store the list in the `\udiss@ldfbabel@list` macro. With this option, we have to turn an additional conditional on, i.e., `\ifudiss@ldfbabel`. Usually we don't use the `.ldf` mechanism at all as mentioned in section 2.

```
  code    ldfbabel       = {%
    \def\udiss@ldfbabel@list{#1}%
    \udiss@lg@usedtrue
    \udiss@ldfbabeltrue
  },%
```

The `nofontwarning` option is developed using the `invbool` *type* which creates (or uses, if present) a conditional and sets it to true. In this case, it is `\ifudiss@font@warning`. Unless the option `nofontwarning` is used, the aforementioned condition is true.

```
  invbool nofontwarning  = {\ifudiss@font@warning}%
}
```

(*End of definition for* `language`*,* `ldfbabel`*, and* `nofontwarning`*. These functions are documented on page* 4.)

license
copyright-yrs
localcopyright

Now we develop the options for setting a license with the legit copyrightable years. As usual, we first develop the required conditionals.

```
\newif\ifudiss@free@culture
\newif\ifudiss@ccbysa
\newif\ifudiss@gfdl
```

We use a new expkv *type*s for license which is `choice`. It takes a definite list of `choice`s that run some arbitrary code. We define an internal macro `\udiss@license@notice` which holds the notice text of the selected licenses. We set the `\ifudiss@free@culture` conditional to true if the user's `choice` is `CC-BY-SA` or `GFDL`. Why we do this is because if the choice is `all-rights-reserved`, it need not be explicitly mentioned. It is always assumed. Also, it is seen in PDF's metadata. For this, we enable the conditional. Using

the `initial` *prefix* of expkv, we set the `all-rights-reserved` as the default choice of users.

```
\ekvdefinekeys{udiss}{%
  choice  license        = {%
    all-rights-reserved    = {%
      \def\udiss@license@notice{%
        All rights reserved.\textLF
      }%
    },%
    CC-BY-SA               = {%
      \def\udiss@license@notice{%
        This work is available under the "Creative Commons
        Attribution ShareAlike 4.0 International" license.%
      }%
      \def\udiss@free@license@url{%
        https://creativecommons.org/licenses/%
        by-sa/4.0/legalcode.txt%
      }%
      \udiss@free@culturetrue
      \udiss@ccbysatrue
    },%
    GFDL                   = {%
      \def\udiss@license@notice{%
        Permission is granted to copy, distribute and/or
        modify this document under the terms of the GNU Free
        Documentation License, Version 1.3 or any later
        version published by the Free Software Foundation;
        with no Invariant Sections, no Front-Cover Texts,
        and no Back-Cover Texts. A copy of the license is
        included in the section entitled "GNU Free
        Documentation License".%
      }%
      \def\udiss@free@license@url{%
        https://www.gnu.org/licenses/fdl-1.3.txt%
      }%
      \udiss@free@culturetrue
      \udiss@gfdltrue
    },%
  },%
  initial license        = {all-rights-reserved},%
```

Then we use the `store` *type* which, as the name suggests, `store`s the value given by the user in a macro. Again, with the `initial` *prefix*, we set the initial values for the two options.

```
  store   copyright-yrs  = {\udiss@copyrightable@years},%
  initial copyright-yrs  = {\the\year},%
  store   localcopyright = {\udiss@local@copyright},%
  initial localcopyright = {Copyright},%
```

(*End of definition for* `license`, `copyright-yrs`, *and* `localcopyright`. *These functions are documented on page* 5.)

`placeholders` This option pre-fills all the necessary fields with some random (and rather delusional :P) values. We use the `nmeta` *type* of expkv for this. This type uses the already developed keys

with some values. When the key developed with `nmeta` is used by the user, it behaves as if the user has passed the keys inside it with the corresponding values.

```
  nmeta   placeholders   = {%
            title          = {%
              A groundbreaking dissertation%
            },%
            author         = {%
              Jane Doe%
            },%
            pronouns       = {She/her},%
            subtitle       = {%
              A milestone study on a longstanding question
              in a discipline%
            },%
            shorttitle     = {A Milestone study},%
            stream         = {intriguing stream},%
            discipline     = {rigorous discipline},
            degree         = {enriching degree},%
            supervisor     = {A kind supervisor},%
            university     = {An ideal university},%
            department     = {A homely department},%
            license        = {CC-BY-SA},%
            copyright-yrs  = {2022, 2023, 2024}%
          }%
}
```

(*End of definition for* `placeholders`. *This function is documented on page* 5.)

lot
lof
declaration
declarationtxt
decllocal

The list of tables and figures are conditionals developed with the `bool` *type* of expkv which creates (or uses, if already present) a boolean. It is by default set to the false value. The declaration also is a `bool`ean as it is turned off by default.

```
\ekvdefinekeys{udiss}{%
  bool    lot            = {\ifudiss@lot},%
  bool    lof            = {\ifudiss@lof},%
  bool    declaration    = {\ifudiss@declaration},%
```

The `declarationtxt` *key* contains text which has paragraphs inside it and the default *key-value*s of expkv are `\def` type (meaning, they don't accept `\par` tokens). So in order to have the effect of `\long\def`, we need to use `long` *prefix* of expkv. We set an initial value for this too. Then we store the value for the local word for declaration.

```
  long
  store   declarationtxt = {\udiss@declaration},%
  initial declarationtxt = {%
    \noindent As required by the university regulations, I
    wish to state that the work embodied in this thesis
    titled \enquote{\udiss@title: \udiss@subtitle} forms
    my own contribution to the research work carried out
    under the guidance of \udiss@supervisor\ at the
    \udiss@university, \udiss@department.

    This work has not been submitted for any other degree
    of this or any other university. Whenever references
    have been made to previous works of others, it has
```

```
    been clearly indicated as such and included in the
    bibliography.

    \bigskip
    \flushright{%
      Date: \today

      \vspace{4em}%

      Name: \udiss@author
    }%

    \bigskip\bigskip
    \flushleft{%
      Certified by:%

      \vspace{4em}%

      \udiss@supervisor
    }%
  },%
  store   decllocal      = {\udiss@decllocal},%
  initial decllocal      = {Declaration}%
}
```

(*End of definition for* `lot` *and others. These functions are documented on page 5.*)

`print` `norefcolors` `udisslinkclr` `udissurlclr` `udissciteclr`

The `print` key is created using the `bool` *type*. The `norefcolors` option, on the other hand, uses the `invbool` *type*. For `udisslinkclr`, `udissurlclr` and `udissciteclr` we use the `store` *type* along with some `initial` values.

```
\ekvdefinekeys{udiss}{%
  bool    print          = {\ifudiss@print},%
  invbool norefcolors    = {\ifudiss@colors},%
  store   udisslinkclr   = {\udiss@int@link@color},%
  initial udisslinkclr   = {udisslink},%
  store   udissurlclr    = {\udiss@int@url@color},%
  initial udissurlclr    = {udissurl},%
  store   udissciteclr   = {\udiss@int@cite@color},%
  initial udissciteclr   = {udisscite}%
}
```

(*End of definition for* `print` *and others. These functions are documented on page 6.*)

`blind`

The next boolean is for printing blind documents.

```
\ekvdefinekeys{udiss}{%
  bool    blind          = {\ifudiss@blind}%
}
```

(*End of definition for* `blind`*. This function is documented on page 5.*)

`framed-title`

We provide a conditional for producing a framed title-page. The conditional will be used later while typesetting.

```
\ekvdefinekeys{udiss}{%
  bool    framed-title   = {\ifudiss@ftitle}%
}
```

(*End of definition for* `framed-title`. *This function is documented on page 5.*)

`oldone` `oldstylenumoff`

We set the old style numerals as the default for typesetting. We provide an inverse boolean for setting the old style numbers off. The special shape for number one is enabled only with the boolean key `oldone`. We develop it here.

```
\ekvdefinekeys{udiss}{%
  invbool oldstylenumoff = {\ifudiss@oldstylenum},%
  bool    oldone         = {\ifudiss@oldone}%
}
```

(*End of definition for* `oldone` *and* `oldstylenumoff`. *These functions are documented on page 7.*)

`explicitext` `ignorefsoff`

To turn on the explicit extension mechanism of the font-loading packages, we provide a conditional. It is disabled by default and hence is of the `bool` type.

```
\ekvdefinekeys{udiss}{%
  bool    explicitext    = {\ifudiss@explicitext},%
  invbool ignorefsoff    = {\ifudiss@ignorefontspec}%
}
```

(*End of definition for* `explicitext` *and* `ignorefsoff`. *These functions are documented on page 8.*)

`fulfilment`

In order to store the sentence written on the title-page, we again use the common `store` and `initial store` pair.

```
\ekvdefinekeys{udiss}{%
  store   fulfilment     = {\udiss@fulfilment},%
  initial fulfilment     = {%
    A dissertation submitted in the partial fulfilment of
    the requirements for the \udiss@degree\ program in
    \udiss@discipline.%
  }%
}
```

(*End of definition for* `fulfilment`. *This function is documented on page 7.*)

All of the upcoming *key*s are font-selection keys. We set their defaults as explained in table 1.

`rmfont` `rmbffont` `rmitfont` `rmbfitfont` `rmslfont` `rmbfslfont`

The following sets the default text font for the document. We are using NewCM family as the default font-family. But as of now, the NewCM fonts don't have all the variants. E.g., it lacks slanted/oblique shapes in the natural weight, but we provide the regular ones in place of them.

```
\ekvdefinekeys{udiss}{%
  store   rmfont         = {\udiss@rmfont},%
  initial rmfont         = {NewCM10-Book.otf},%
  store   rmbffont       = {\udiss@rmbffont},%
  initial rmbffont       = {NewCM10-Bold.otf},%
  store   rmitfont       = {\udiss@rmitfont},%
  initial rmitfont       = {NewCM10-BookItalic.otf},%
  store   rmbfitfont     = {\udiss@rmbfitfont},%
  initial rmbfitfont     = {NewCM10-BoldItalic.otf},%
  store   rmslfont       = {\udiss@rmslfont},%
  initial rmslfont       = {NewCM10-Book.otf},%
  store   rmbfslfont     = {\udiss@rmbfslfont},%
  initial rmbfslfont     = {NewCM10-Bold.otf},%
```

(*End of definition for* `rmfont` *and others. These functions are documented on page 9.*)

sffont
sfbffont
sfitfont
sfbfitfont
sfslfont
sfbfslfont

Similarly, we store values for the sans fonts too. NewCM lacks certain variants in sans, but again, the options are set font-agnostically and hence they are more in number than the actually available variants. The sans fonts in NewCM, don't have an Italic design. We have used the oblique version of the fonts there instead. Note that the same designs are used for the slanted fonts too.

```
store   sffont          = {\udiss@sffont},%
initial sffont          = {NewCMSans10-Book.otf},%
store   sfbffont        = {\udiss@sfbffont},%
initial sfbffont        = {NewCMSans10-Bold.otf},%
store   sfitfont        = {\udiss@sfitfont},
initial sfitfont        = {NewCMSans10-BookOblique.otf},%
store   sfbfitfont      = {\udiss@sfbfitfont},%
initial sfbfitfont      = {NewCMSans10-BoldOblique.otf},%
store   sfslfont        = {\udiss@sfslfont},%
initial sfslfont        = {NewCMSans10-BookOblique.otf},%
store   sfbfslfont      = {\udiss@sfbfslfont},%
initial sfbfslfont      = {NewCMSans10-BoldOblique.otf},%
```

(*End of definition for* `sffont` *and others. These functions are documented on page 9.*)

ttfont
ttbffont
ttitfont
ttbfitfont
ttslfont
ttbfslfont

The settings for typewriter fonts are loaded in the below code. We have a strange situation here. NewCM provides `BookItalic`, but doesn't have a `BoldItalic`. We instead have `BoldOblique`. Thus we set the `BookItalic` as the `ttitfont` and `BoldOblique` as the `ttbfit` font. The same pair for `ttslfont` and `ttbfslfont`.

```
store   ttfont          = {\udiss@ttfont},%
initial ttfont          = {NewCMMono10-Book.otf},%
store   ttbffont        = {\udiss@ttbffont},
initial ttbffont        = {NewCMMono10-Bold.otf},%
store   ttitfont        = {\udiss@ttitfont},%
initial ttitfont        = {NewCMMono10-BookItalic.otf},%
store   ttbfitfont      = {\udiss@ttbfitfont},%
initial ttbfitfont      = {NewCMMono10-BoldOblique.otf},%
store   ttslfont        = {\udiss@ttslfont},%
initial ttslfont        = {NewCM10-Book.otf},%
store   ttbfslfont      = {\udiss@ttbfslfont},%
initial ttbfslfont      = {NewCMMono10-BoldOblique.otf},%
```

(*End of definition for* `ttfont` *and others. These functions are documented on page 9.*)

mathfont
mathbffont

Maths fonts have very little variation, so we just have two parameters for selecting the fonts and two of the usual parameters for features and extensions.

```
store   mathfont        = {\udiss@mathfont},%
initial mathfont        = {NewCMMath-Book.otf},%
store   mathbffont      = {\udiss@mathbffont},%
initial mathbffont      = {NewCMMath-Bold.otf},%
```

(*End of definition for* `mathfont` *and* `mathbffont`*. These functions are documented on page 9.*)

rmfeatures
rmfntext
sffeatures
sffntext
ttfeatures
ttfntext
mathfeatures
mathfntext

If the user wants to add features to the selected fonts, they are allowed to use these commands and if they use the `explicitext` option, they can choose the font extensions

too using the *key*s specifically developed for that. We use `store` and `choice` *type*s for this.

```
store   rmfeatures     = {\udiss@rmfeatures},%
choice  rmfntext       = {%
          ttf            = {%
            \ifudiss@explicitext
              \def\udiss@rmfntext{ttf}%
            \else
              \ClassError{udiss}{%
                Option `explicitext' is needed for adding
                font extensions%
              }{%
                You have loaded font-option `rmfntext' to
                add extension to font-names.
                This\MessageBreak option is supposed to be
                used only with the `explicitext' option.
                E.g., for\MessageBreak loading, say,
                FreeSerif, you may choose one of the
                following two methods:%
                \MessageBreak\MessageBreak
                1. The recommended method:%
                \MessageBreak
                \space\space\space
                ----8<---------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  rmfont = {FreeSerif.ttf}\@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}\MessageBreak
                \space\space\space
                ----8<---------------8<----%
                \MessageBreak\MessageBreak
                2. The alternative:%
                \MessageBreak
                \space\space\space
                ----8<------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  explicitext,\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  rmfont = {FreeSerif},\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  rmfntext = {ttf}\@percentchar
                  \MessageBreak
                  \space\space\space
```

```
      ]{udiss}%
      \MessageBreak
      \space\space\space
      ----8<------------8<----%
    }%
  \fi
},%
otf            = {%
  \ifudiss@explicitext
    \def\udiss@rmfntext{otf}%
  \else
    \ClassError{udiss}{%
      Option `explicitext' is needed for adding
      font extensions%
    }{%
      You have loaded font-option `rmfntext' to
      add extension to font-names.
      This\MessageBreak option is supposed to be
      used only with the `explicitext' option.
      E.g., for\MessageBreak loading, say,
      KpRoman-Regular.otf, you may choose one of
      the following two\MessageBreak
      methods:%
      \MessageBreak\MessageBreak
      1. The recommended method:%
      \MessageBreak
      \space\space\space
      ----8<---------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        rmfont = {KpRoman-Regular.otf}%
        \@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}\MessageBreak
      \space\space\space
      ----8<---------------------8<----%
      \MessageBreak
      \MessageBreak
      2. The alternative:%
      \MessageBreak
      \space\space\space
      ----8<------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        explicitext,\@percentchar
        \MessageBreak
        \space\space\space\space\space
```

```
                  rmfont = {KpRoman-Regular},%
                  \@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  rmfntext = {otf}\@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}%
                \MessageBreak
                \space\space\space
                ----8<------------------8<----%
              }%
            \fi
          }%
        },%
store   sffeatures     = {\udiss@sffeatures},%
choice  sffntext       = {%
          ttf            = {%
            \ifudiss@explicitext
              \def\udiss@sffntext{ttf}%
            \else
              \ClassError{udiss}{%
                Option `explicitext' is needed for adding
                font extensions%
              }{%
                You have loaded font-option `sffntext' to
                add extension to font-names.
                This\MessageBreak option is supposed to be
                used only with the `explicitext' option.
                E.g., for\MessageBreak loading, say,
                FreeSans, you may choose one of the
                following two methods:%
                \MessageBreak\MessageBreak
                1. The recommended method:%
                \MessageBreak
                \space\space\space
                ----8<---------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  sffont = {FreeSans.ttf}\@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}\MessageBreak
                \space\space\space
                ----8<---------------8<----%
                \MessageBreak\MessageBreak
                2. The alternative:%
                \MessageBreak
                \space\space\space
                ----8<------------8<----%
                \MessageBreak
```

```
        \space\space\space
        \string\documentclass[\@percentchar
          \MessageBreak
          \space\space\space\space\space
          explicitext,\@percentchar
          \MessageBreak
          \space\space\space\space\space
          sffont = {FreeSans},\@percentchar
          \MessageBreak
          \space\space\space\space\space
          sffntext = {ttf}\@percentchar
          \MessageBreak
          \space\space\space
        ]{udiss}%
        \MessageBreak
        \space\space\space
        ----8<------------8<----%
      }%
    \fi
  },%
  otf            = {%
    \ifudiss@explicitext
      \def\udiss@sffntext{otf}%
    \else
      \ClassError{udiss}{%
        Option `explicitext' is needed for adding
        font extensions%
      }{%
        You have loaded font-option `sffntext' to
        add extension to font-names.
        This\MessageBreak option is supposed to be
        used only with the `explicitext' option.
        E.g., for\MessageBreak loading, say,
        KpSans-Regular.otf, you may choose one of
        the following two\MessageBreak
        methods:%
        \MessageBreak\MessageBreak
        1. The recommended method:%
        \MessageBreak
        \space\space\space
        ----8<--------------------8<----%
        \MessageBreak
        \space\space\space
        \string\documentclass[\@percentchar
          \MessageBreak
          \space\space\space\space\space
          sffont = {KpSans-Regular.otf}%
          \@percentchar
          \MessageBreak
          \space\space\space
        ]{udiss}\MessageBreak
        \space\space\space
        ----8<--------------------8<----%
        \MessageBreak
```

```
                \MessageBreak
                2. The alternative:%
                \MessageBreak
                \space\space\space
                ----8<------------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  explicitext,\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  sffont = {KpSans-Regular},%
                  \@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  sffntext = {otf}\@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}%
                \MessageBreak
                \space\space\space
                ----8<------------------8<----%
              }%
            \fi
          }%
        },%
store   ttfeatures     = {\udiss@ttfeatures},%
choice  ttfntext       = {%
          ttf            = {%
            \ifudiss@explicitext
              \def\udiss@ttfntext{ttf}%
            \else
              \ClassError{udiss}{%
                Option `explicitext' is needed for adding
                font extensions%
              }{%
                You have loaded font-option `ttfntext' to
                add extension to font-names.
                This\MessageBreak option is supposed to be
                used only with the `explicitext' option.
                E.g., for\MessageBreak loading, say,
                FreeMono, you may choose one of the
                following two methods:%
                \MessageBreak\MessageBreak
                1. The recommended method:%
                \MessageBreak
                \space\space\space
                ----8<---------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
```

```
        \space\space\space\space\space
        ttfont = {FreeMono.ttf}\@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}\MessageBreak
      \space\space\space
      ----8<---------------8<----%
      \MessageBreak\MessageBreak
      2. The alternative:%
      \MessageBreak
      \space\space\space
      ----8<------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        explicitext,\@percentchar
        \MessageBreak
        \space\space\space\space\space
        ttfont = {FreeMono},\@percentchar
        \MessageBreak
        \space\space\space\space\space
        ttfntext = {ttf}\@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}%
      \MessageBreak
      \space\space\space
      ----8<------------8<----%
    }%
  \fi
},%
otf            = {%
  \ifudiss@explicitext
    \def\udiss@ttfntext{otf}%
  \else
    \ClassError{udiss}{%
      Option `explicitext' is needed for adding
      font extensions%
    }{%
      You have loaded font-option `ttfntext' to
      add extension to font-names.
      This\MessageBreak option is supposed to be
      used only with the `explicitext' option.
      E.g., for\MessageBreak loading, say,
      KpMono-Regular.otf, you may choose one of
      the following two\MessageBreak
      methods:%
      \MessageBreak\MessageBreak
      1. The recommended method:%
      \MessageBreak
      \space\space\space
      ----8<--------------------8<----%
```

```
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  ttfont = {KpMono-Regular.otf}%
                  \@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}\MessageBreak
                \space\space\space
                ----8<--------------------8<----%
                \MessageBreak
                \MessageBreak
                2. The alternative:%
                \MessageBreak
                \space\space\space
                ----8<------------------8<----%
                \MessageBreak
                \space\space\space
                \string\documentclass[\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  explicitext,\@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  ttfont = {KpMono-Regular},%
                  \@percentchar
                  \MessageBreak
                  \space\space\space\space\space
                  ttfntext = {otf}\@percentchar
                  \MessageBreak
                  \space\space\space
                ]{udiss}%
                \MessageBreak
                \space\space\space
                ----8<------------------8<----%
              }%
            \fi
          }%
        },%
store   mathfeatures   = {\udiss@mathfeatures},%
choice  mathfntext     = {%
          ttf            = {%
            \ifudiss@explicitext
              \def\udiss@mathfntext{ttf}%
            \else
              \ClassError{udiss}{%
                Option `explicitext' is needed for adding
                font extensions%
              }{%
                You have loaded font-option `mathfntext'
                to add extension to font-names.
                This\MessageBreak option is supposed to be
```

```
      used only with the `explicitext' option.
      E.g., for\MessageBreak loading, say,
      NotoSansMath-Regular, you may choose one
      of the following two\MessageBreak
      methods:%
      \MessageBreak\MessageBreak
      1. The recommended method:%
      \MessageBreak
      \space\space\space
      ----8<----------------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfont = {NotoSansMath-Regular.ttf}%
        \@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}\MessageBreak
      \space\space\space
      ----8<----------------------------8<----%
      \MessageBreak\MessageBreak
      2. The alternative:%
      \MessageBreak
      \space\space\space
      ----8<-------------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        explicitext,\@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfont = {NotoSansMath-Regular},%
        \@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfntext = {ttf}\@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}%
      \MessageBreak
      \space\space\space
      ----8<-------------------------8<----%
    }%
  \fi
},%
otf            = {%
  \ifudiss@explicitext
    \def\udiss@mathfntext{otf}%
  \else
    \ClassError{udiss}{%
```

```
      Option `explicitext' is needed for adding
      font extensions%
    }{%
      You have loaded font-option `mathfntext'
      to add extension to font-names.
      This\MessageBreak option is supposed to be
      used only with the `explicitext' option.
      E.g., for\MessageBreak loading, say,
      XITSMath-Regular, you may choose one of
      the following two\MessageBreak methods:%
      \MessageBreak\MessageBreak
      1. The recommended method:%
      \MessageBreak
      \space\space\space
      ----8<----------------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfont = {XITSMath-Regular.otf}%
        \@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}\MessageBreak
      \space\space\space
      ----8<----------------------------8<----%
      \MessageBreak\MessageBreak
      2. The alternative:%
      \MessageBreak
      \space\space\space
      ----8<-------------------------8<----%
      \MessageBreak
      \space\space\space
      \string\documentclass[\@percentchar
        \MessageBreak
        \space\space\space\space\space
        explicitext,\@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfont = {XITSMath-Regular},%
        \@percentchar
        \MessageBreak
        \space\space\space\space\space
        mathfntext = {otf}\@percentchar
        \MessageBreak
        \space\space\space
      ]{udiss}%
      \MessageBreak
      \space\space\space
      ----8<-------------------------8<----%
    }%
  \fi
}%
```

```
            },%
}
```

(*End of definition for* `rmfeatures` *and others. These functions are documented on page 8.*)

titleshape
titlefont
titleffeat
titlesize
titlecolor

We store various variables in these keys for customising the layout of the title-page. The following are the keys specifically developed for customising the title. The sizing is done with internal macros specifically designed for consistent size-changing. We use fontspec-scaling for that.

```
\ekvdefinekeys{udiss}{%
  store   titleshape     = {\udiss@title@shape},%
  initial titleshape     = {\bfseries},%
  store   titlefont      = {\udiss@title@font},%
  store   titlesize      = {\udiss@title@size},%
  initial titlesize      = {\udiss@int@huge},%
  store   titlecolor     = {\udiss@title@color},%
  initial titlecolor     = {black},%
  store   titleffeat     = {\udiss@title@fontfeats},%
  initial titleffeat     = {}%
}
```

(*End of definition for* `titleshape` *and others. These functions are documented on page 10.*)

subtitleshape
subtitlefont
subtitleffeat
subtitlesize
subtitlecolor

The same logic is applied for subtitles and these variables are created.

```
\ekvdefinekeys{udiss}{%
  store   subtitleshape  = {\udiss@subtitle@shape},%
  initial subtitleshape  = {\normalfont},%
  store   subtitlefont   = {\udiss@subtitle@font},%
  store   subtitlesize   = {\udiss@subtitle@size},%
  initial subtitlesize   = {\udiss@int@large},%
  store   subtitlecolor  = {\udiss@subtitle@color},%
  initial subtitlecolor  = {black},%
  store   subtitleffeat  = {\udiss@subtitle@fontfeats},%
  initial subtitleffeat  = {}%
}
```

(*End of definition for* `subtitleshape` *and others. These functions are documented on page 10.*)

authorshape
authorfont
authorsize
authorcolor
authorffeat

Similarly, we have the following keys for author.

```
\ekvdefinekeys{udiss}{%
  store   authorshape    = {\udiss@author@shape},%
  initial authorshape    = {\normalfont},%
  store   authorfont     = {\udiss@author@font},%
  store   authorsize     = {\udiss@author@size},%
  initial authorsize     = {\udiss@int@large},%
  store   authorcolor    = {\udiss@author@color},%
  initial authorcolor    = {black},%
  store   authorffeat    = {\udiss@author@fontfeats},%
  initial authorffeat    = {}%
}
```

(*End of definition for* `authorshape` *and others. These functions are documented on page 10.*)

unishape
unifont
unisize
unicolor
uniffeat

The same pattern for printing the university name.

```
\ekvdefinekeys{udiss}{%
  store   unishape       = {\udiss@uni@shape},%
  initial unishape       = {\itshape},%
  store   unifont        = {\udiss@uni@font},%
  store   unisize        = {\udiss@uni@size},%
  initial unisize        = {\normalsize},%
  store   unicolor       = {\udiss@uni@color},%
  initial unicolor       = {black},%
  store   uniffeat       = {\udiss@uni@fontfeats}%
}
```

(*End of definition for* `unishape` *and others. These functions are documented on page* 10.)

deptshape
deptfont
deptsize
deptcolor
deptffeat

The following keys for the department.

```
\ekvdefinekeys{udiss}{%
  store   deptshape      = {\udiss@dept@shape},%
  initial deptshape      = {\normalfont},%
  store   deptfont       = {\udiss@dept@font},%
  store   deptsize       = {\udiss@dept@size},%
  initial deptsize       = {\udiss@int@small},%
  store   deptcolor      = {\udiss@dept@color},%
  initial deptcolor      = {black},%
  store   deptffeat      = {\udiss@dept@fontfeats},%
  initial deptffeat      = {}%
}
```

(*End of definition for* `deptshape` *and others. These functions are documented on page* 10.)

ffshape
fffont
ffsize
ffcolor
ffffeat

These are for the fulfilment-text.

```
\ekvdefinekeys{udiss}{%
  store   ffshape        = {\udiss@fulfilment@shape},%
  initial ffshape        = {\normalfont},%
  store   fffont         = {\udiss@fulfilment@font},%
  store   ffsize         = {\udiss@fulfilment@size},%
  initial ffsize         = {\udiss@int@small},%
  store   ffcolor        = {\udiss@fulfilment@color},%
  initial ffcolor        = {black},%
  store   fffeat         = {\udiss@fulfilment@fontfeats},%
  initial fffeat         = {}%
}
```

(*End of definition for* `ffshape` *and others. These functions are documented on page* 10.)

logowidth
logoheight

The following parameters save the values for the logo-height and the logo-width. These are the last options after which we `\Process` our options.

```
\ekvdefinekeys{udiss}{%
  store   logowidth      = {\udiss@logo@width},%
  initial logowidth      = {0.3},%
  store   logoheight     = {\udiss@logo@hght},%
  initial logoheight     = {0.075}%
}
\ekvoProcessGlobalOptions{udiss}
```

(*End of definition for* `logowidth` *and* `logoheight`. *These functions are documented on page 9.*)

`\dissertationstyle` We develop the command for loading custom styles. It just inputs files starting with `udiss-style-` followed by the designated tag, e.g., `ukerala` for the university of Kerala.

```
\NewDocumentCommand{ \dissertationstyle }{ m }{%
  \AddToHook{begindocument/before}{%
    \input{udiss-style-#1}%
  }%
}
```

(*End of definition for* `\dissertationstyle`. *This function is documented on page 7.*)

To minimise loading a lot of packages, we use the memoir class which provides many features useful for document-creation. But we also need to set print and digital versions of the dissertations. For that we use our conditional.

```
\ifudiss@print
\else
  \PassOptionsToClass{oneside}{memoir}%
\fi
\LoadClass{memoir}
```

`\title` `\author` We defined most other commands as per the flow of the documentation, but as mentioned before, we had kept `\title` and `\author` aside. As mentioned before, we did so because these commands are defined by the memoir class. If we had defined it before loading the memoir class (which we just loaded), the commands would have got renewed after loading the class and thus our internal macros would have failed. On the other hand, we could not have loaded the memoir class in the beginning, because we use the `\ifudiss@print` conditional, i.e., developed with the help of expkv|DEF package. Even if we had separated the code of `print` key, it would have been necessary to process the keys using `\ekvoProcessGlobalOptions`, but we could not have done that since there were a lot of other options pending. This is why, we develop the `\title` and `\author` commands now.

```
\RenewDocumentCommand{ \title }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@title@err
    \def\udiss@title{-NoValidTitle-}%
  }{%
    \def\udiss@title{#1}%
  }%
}
\RenewDocumentCommand{ \author }{ m }{%
  \IfBlankTF{#1}{%
    \udiss@blank@author@err
    \def\udiss@author{-NoValidAuthor-}%
  }{%
    \def\udiss@author{#1}%
  }%
}
```

(*End of definition for* `\title` *and* `\author`. *These functions are documented on page 3.*)

We are using OpenType fonts for better accessibility. They are incompatible with the classic PDFLaTeX and require either LuaLaTeX or XeLaTeX for compilation. To ensure

that one of these two is loaded, we load the iftex package and mandate a Unicode engine with the `\RequireTUTeX` command.

```
\RequirePackage{iftex}
\RequireTUTeX
```

We used custom udiss size commands. They are developed with fontspec-scaling as follows:

```
\def\udiss@int@HUGE{%
  \addfontfeature{Scale={2.5}}%
}
\def\udiss@int@Huge{%
  \addfontfeature{Scale={2}}%
}
\def\udiss@int@huge{%
  \addfontfeature{Scale={1.7}}%
}
\def\udiss@int@LARGE{%
  \addfontfeature{Scale={1.4}}%
}
\def\udiss@int@Large{%
  \addfontfeature{Scale={1.2}}%
}
\def\udiss@int@large{%
  \addfontfeature{Scale={1.1}}%
}
\def\udiss@int@small{%
  \addfontfeature{Scale={0.9}}%
}
\def\udiss@int@footnotesize{%
  \addfontfeature{Scale={0.8}}%
}
\def\udiss@int@scriptsize{%
  \addfontfeature{Scale={0.7}}%
}
\def\udiss@int@tiny{%
  \addfontfeature{Scale={0.6}}%
}
\def\udiss@int@miniscule{%
  \addfontfeature{Scale={0.5}}%
}
```

We require the babel package for loading the language settings. We have the following three situations with loading languages in udiss:

1. Users don't load any of the language options at all.
2. Users load languages with option `ldfbabel`.
3. Users load languages with option `language`.

If a user has used `ldfbabel`, then the macro which stores the language-list is directly passed to the package using `\PassOptionsToPackage`.

```
\ifudiss@lg@used
  \ifudiss@ldfbabel
```

```
    \PassOptionsToPackage{%
      \udiss@ldfbabel@list
    }{babel}%
  \fi
```

If user hasn't loaded any language options, we assume that they have no intent to load languages that require `.ldf` files and hence we mandate loading `.ini` files for all the languages. In addition to that, we load `english` as the main language. This takes care of the first two situations.

```
\else
  \PassOptionsToPackage{english,provide*=*}{babel}%
\fi
\RequirePackage{babel}
```

We also load package iflang for testing language-names.

```
\RequirePackage{iflang}
```

First we check if the user has loaded `ldf` languages. We ignore that branch completely. If it is not loaded and they have loaded exactly one language, then we set that language as the main language. Note that when user uses one language, the value of the counter is 2 as it is increased by the counter each time.

```
\newcounter{udiss@tmp@langcnt}
\ifudiss@ldfbabel
\else
  \ifudiss@lg@used
    \ifnum\theudiss@langbabel=2\relax
      \addtocounter{udiss@langbabel}{-1}%
      \ifLuaTeX
        \babelprovide[%
          main,%
          import,%
          onchar                  ={ids fonts}%
        ]{%
          \csname
            udiss@langbabel@\alph{udiss@langbabel}%
          \endcsname
        }%
      \else
        \babelprovide[main,import]{%
          \csname
            udiss@langbabel@\alph{udiss@langbabel}%
          \endcsname
        }%
      \fi
    \fi
```

When users use multiple languages, the situation is a bit more complicated. Because the parser parses and increases the counter each time. Suppose, the user provided 5 languages, the parser will create 5 macros *and* increase the counter for one last time resulting in the value 6. In order to stop processing when the number of actually used languages is reached, we need to store the current number *minus one* because of this. We have to start a loop from value 1 till it reaches the last value. For this, we reset the counter. In order to stop when all the languages are parsed, we need to remember the actual value of the counter somewhere. Hence we create a temporary counter for this and equate it

with the value of our actual counter minus 1. Note that we escape `\ifudiss@ldfbabel` too. Now we start the calculation when the counter's value is 3 or more, basically the author has to use at least two languages in order to activate the loop ahead.

```
    \ifnum\theudiss@langbabel>2\relax
      \setcounter{udiss@tmp@langcnt}{%
        \numexpr\theudiss@langbabel -1\relax
      }%
```

We now can safely manipulate our actual counter. Starting from the first language, we load all of them with the `\babelprovide` command. We reset the counter to 1:

```
      \setcounter{udiss@langbabel}{1}%
```

Now we start a loop.

```
      \loop
```

Only LuaLaTeX supports the `ids fonts` parameter with `\babelprovide` for applying fonts to the input text with respect to their Unicode slots. That is why we start the `\ifLuaTeX` conditional and load the `onchar` option. Loading a conditional inside `\babelprovide` wasn't permitted because of expansion, hence we load the entire command in the else branch. Notice that there is no space after the `=` sign of the `onchar` option. To know why, you may want to read this wonderful answer by Jonathan P. Spratte: https://topanswers.xyz/tex?q=8039#a7652.

```
        \ifLuaTeX
          \babelprovide[%
            import,%
            onchar                  ={ids fonts}%
          ]{%
```

Now we pass `\udiss@langbabel@a` as an argument to `\babelprovide` command (the value of the counter is as of now 1).

```
            \csname
              udiss@langbabel@\alph{udiss@langbabel}%
            \endcsname
          }%
```

We start the `\else` branch and use the same thing again, just without the `onchar` option .

```
        \else
          \babelprovide[import]{%
            \csname
              udiss@langbabel@\alph{udiss@langbabel}%
            \endcsname
          }%
        \fi
```

Woohoo! We loaded the first language successfully. Now, we increment the counter, but there is another situation which we have to take into account. What if the user just loaded a single language with the `language` option? The loop will run into trouble, that's why we only run the loop if the value of the total number of languages is more than one. We had saved it in a temporary counter.

```
        \ifnum\theudiss@tmp@langcnt=1\relax
        \else
          \addtocounter{udiss@langbabel}{1}%
        \fi
```

Now the value of the counter is 2 which means the second language. We need to again load the same settings for it, but we can't keep doing this manually since 1) we are lazy (the most important) 2) we have no idea how many languages the user has loaded, so now we actually activate the `\loop` that we had started with the `\repeat` command. It runs all the commands seen after the `\loop` command (this includes the one used for increasing the counter too) till the value of the counter is less than the value of `udiss@tmp@langcnt`. Remember we had already set it to the value of the original counter (before setting it to 1) *minus one*. Now the next command will stop at one number even lesser than it as we have used the `<` relation while comparing. Also, here we will close the if-conditional that checked if more than one languages were used or not.

```
    \ifnum\theudiss@langbabel<\theudiss@tmp@langcnt
    \relax
  \repeat
\fi
```

Are you wondering why we did this? Because we want to set the *exact* last language as the `main` language! The current value of our counter at this point has that language. But remember we are out of the loop as well as the if-conditional. Thus we have to again consider the situation where user just passed one language. So far what we have done is load all the languages except the last one with `\babelprovide`. This includes the single-language case. Thus we need no further processing in that case. So we only parse the last pending language if the user had passed multiple languages. We start an if-conditional and keep its `if`-branch empty and write the rest in the `else`-branch.

```
\ifnum\theudiss@langbabel=1\relax
```

We again use `\babelprovide`, but this time with `main` option. Load the last language successfully.

```
\else
  \ifLuaTeX
    \babelprovide[%
      main,%
      import,%
      onchar                  ={ids fonts}%
    ]{%
      \csname
        udiss@langbabel@\alph{udiss@langbabel}%
      \endcsname
    }%
  \else
    \babelprovide[main,import]{%
      \csname
        udiss@langbabel@\alph{udiss@langbabel}%
      \endcsname
    }%
  \fi
\fi
```

Now it's the time to load language support files. We need to load all the language support files and here we don't have the constraint to treat the last language specially. Therefore we add 1 to the temporary counter (the one that was decreased before).

```
\setcounter{udiss@tmp@langcnt}{%
  \numexpr\theudiss@langbabel + 1\relax
}%
```

We again start from the first language. If the support file exists for that language, we `\input` it.

```
    \setcounter{udiss@langbabel}{1}%
    \loop
      \IfFileExists{%
        udiss-fonts-\csname
          udiss@langbabel@\alph{udiss@langbabel}%
        \endcsname
      }{%
        \input{%
          udiss-fonts-\csname
            udiss@langbabel@\alph{udiss@langbabel}%
          \endcsname
        }%
```

Otherwise, we check if the warning option is true. By default it is always true, but if the user has used `nofontwarning` option to turn them off, this conditional should do its job.

```
      }{%
        \ifudiss@font@warning
          \ClassWarningNoLine{udiss}{%
            We don't support \csname
              udiss@langbabel@\alph{udiss@langbabel}%
            \endcsname\space yet.%
            \MessageBreak
            It requires a font-setup. You may add the fonts
            directly\MessageBreak to your preamble using:%
            \MessageBreak\MessageBreak
            \space\space\string\babelfont[%
              \csname
                udiss@langbabel@\alph{udiss@langbabel}%
              \endcsname
            ]{rm}{main font}%
            \MessageBreak
            \space\space\string\babelfont[%
              \csname
                udiss@langbabel@\alph{udiss@langbabel}%
              \endcsname
            ]{sf}{sans font}%
            \MessageBreak
            \space\space\string\babelfont[%
              \csname
                udiss@langbabel@\alph{udiss@langbabel}%
              \endcsname
            ]{tt}{mono font}%
            \MessageBreak\MessageBreak
            Read more about this in the documentation of
            `udiss' in a\MessageBreak section called
            `multilingual typesetting'. Consider%
            \MessageBreak contacting us with a support
            request and let us know some\MessageBreak
            suitable free (libre) fonts for this
            language. If you\MessageBreak don't wish to see
            this warning again, use the option%
```

```
            \MessageBreak `nofontwarning' to suppress it%
          }%
        \fi
      }%
```

Like before, we increment the counter and start the loop till this is repeated for all the languages.

```
      \addtocounter{udiss@langbabel}{1}%
      \ifnum\theudiss@langbabel<\theudiss@tmp@langcnt\relax
    \repeat
  \fi
\fi
```

After all this processing, we load the Roman, `Typewriter` and Sans fonts stored in the variables set by *key*s.

```
\babelfont{rm}[%
  \ifudiss@ignorefontspec
    IgnoreFontspecFile,%
  \fi
  \ifudiss@explicitext
    Extension            = {.\udiss@rmfntext},%
  \fi
  BoldFont               = {\udiss@rmbffont},%
  ItalicFont             = {\udiss@rmitfont},%
  BoldItalicFont         = {\udiss@rmbfitfont},%
  SlantedFont            = {\udiss@rmslfont},%
  BoldSlantedFont        = {\udiss@rmbfslfont},%
  \ifudiss@oldstylenum
    Numbers              = {OldStyle},%
    \ifudiss@oldone
      CharacterVariant   = {6},%
    \fi
  \fi
  \ifLuaTeX
    Renderer             = {HarfBuzz},%
  \fi
  \udiss@rmfeatures
]{\udiss@rmfont}
\babelfont{sf}[%
  \ifudiss@ignorefontspec
    IgnoreFontspecFile,%
  \fi
  \ifudiss@explicitext
    Extension            = {.\udiss@sffntext},%
  \fi
  BoldFont               = {\udiss@sfbffont},%
  ItalicFont             = {\udiss@sfitfont},%
  BoldItalicFont         = {\udiss@sfbfitfont},%
  SlantedFont            = {\udiss@sfslfont},%
  BoldSlantedFont        = {\udiss@sfbfslfont},%
  \ifudiss@oldstylenum
    Numbers              = {OldStyle},%
    \ifudiss@oldone
      CharacterVariant   = {6},%
```

```
    \fi
  \fi
  \ifLuaTeX
    Renderer              = {HarfBuzz},%
  \fi
  \udiss@sffeatures
]{\udiss@sffont}
\babelfont{tt}[%
  \ifudiss@ignorefontspec
    IgnoreFontspecFile,%
  \fi
  \ifudiss@explicitext
    Extension             = {.\udiss@ttfntext},%
  \fi
  BoldFont                = {\udiss@ttbffont},%
  ItalicFont              = {\udiss@ttitfont},%
  BoldItalicFont          = {\udiss@ttbfitfont},%
  SlantedFont             = {\udiss@ttslfont},%
  BoldSlantedFont         = {\udiss@ttbfslfont},%
  \ifLuaTeX
    Renderer              = {HarfBuzz},%
  \fi
  \udiss@ttfeatures
]{\udiss@ttfont}
```

We load the unicode-math here for loading the math fonts. After that, inside `\expanded`, we load the `\setmathfont` command.

```
\RequirePackage{unicode-math}
\expanded{%
  \setmathfont[%
    \ifudiss@ignorefontspec%
      IgnoreFontspecFile,%
    \fi%
    \ifudiss@explicitext
      Extension             = {.\udiss@mathfntext},%
    \fi
    BoldFont                = {\udiss@mathbffont},%
    SlantedFont             = {\udiss@mathfont},%
    BoldSlantedFont         = {\udiss@mathbffont},%
    \udiss@mathfeatures%
  ]{\udiss@mathfont}%
}
```

Now we load a few last packages.

```
\RequirePackage{graphicx}
\RequirePackage{xcolor}
\RequirePackage{csquotes}
```

For Creative Commons license, we use the doclicense package with appropriate settings.

```
\ifudiss@ccbysa
  \RequirePackage[%
    hyperxmp            = {false},%
    type                = {CC},%
```

```
    modifier           = {by-sa},%
    version            = {4.0}%
  ]{doclicense}
\fi
```

We set the udiss-specific-colours with the following commands, but we set them conditionally. If the colours are disabled, we set the same colours to `black` value.

```
\ifudiss@colors
  \colorlet{udisslink}{red!60!black}
  \colorlet{udissurl}{blue!60!black}
  \colorlet{udisscite}{green!60!black}
\else
  \colorlet{udisslink}{black}
  \colorlet{udissurl}{black}
  \colorlet{udisscite}{black}
\fi
```

We start the hook `begindocument/before` and check if author did not provide any of the *strict* fields and if they have done so, we generate errors.

```
\AddToHook{begindocument/before}{%
  \ifdefined\udiss@title
  \else
    \def\udiss@title{\textbf{-NoValidTitle-}}%
    \ClassError{udiss}{%
      Title not given%
    }{%
      `title' is a compulsory field. Provide it in the
      preamble like in the following:%
      \MessageBreak
      \MessageBreak
      ----8<----------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\title{title-of-the-dissertation}%
      \MessageBreak\MessageBreak
      ----8<----------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@author
  \else
    \def\udiss@author{\textbf{-NoValidAuthor-}}%
    \ClassError{udiss}{%
      No author given in a normal (non-blind) document%
    }{%
      Do you want a blind document? If yes, then use the
      package option `blind' like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<-----------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\documentclass[blind]{udiss}%
      \MessageBreak\MessageBreak
      ----8<-----------------8<----%
      \MessageBreak
```

```
      Otherwise, `author' is a compulsory field. Provide it
      in the preamble like in\MessageBreak the following:%
      \MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\author{author-name}%
      \MessageBreak\MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@stream
  \else
    \def\udiss@stream{\textbf{-NoValidStream-}}%
    \ClassError{udiss}{%
      Stream not given%
    }{%
      `stream' is a compulsory field. Provide it in the
      preamble like in the following:%
      \MessageBreak
      \MessageBreak
      ----8<-----------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\stream{stream-of-the-candidate}%
      \MessageBreak\MessageBreak
      ----8<-----------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@discipline
  \else
    \def\udiss@discipline{\textbf{-NoValidDiscipline-}}%
    \ClassError{udiss}{%
      Discipline not given%
    }{%
      `discipline' is a compulsory field. Provide it in the
      preamble like in the following:%
      \MessageBreak
      \MessageBreak
      ----8<-----------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\discipline%
      {discipline-of-the-candidate}%
      \MessageBreak\MessageBreak
      ----8<-----------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@degree
  \else
    \def\udiss@degree{\textbf{-NoValidDegree-}}%
    \ClassError{udiss}{%
      Degree not given%
    }{%
```

```
      `degree' is a compulsory field. Provide it in the
      preamble like in the following:%
      \MessageBreak
      \MessageBreak
      ----8<----------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\degree{degree-of-the-candidate}%
      \MessageBreak\MessageBreak
      ----8<----------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@supervisor
  \else
    \def\udiss@title{\textbf{-NoValidTitle-}}%
    \ClassError{udiss}{%
      No supervisor given in a normal (non-blind) document%
    }{%
      Do you want a blind document? If yes, then use the
      package option `blind' like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<-----------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\documentclass[blind]{udiss}%
      \MessageBreak\MessageBreak
      ----8<-----------------8<----%
      \MessageBreak
      Otherwise, `supervisor' is a compulsory field. Provide
      it in the preamble like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\supervisor{supervisor's-name}%
      \MessageBreak\MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@university
  \else
    \def\udiss@university{\textbf{-NoValidUniversity-}}%
    \ClassError{udiss}{%
      No university given in a normal (non-blind) document%
    }{%
      Do you want a blind document? If yes, then use the
      package option `blind' like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<-----------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\documentclass[blind]{udiss}%
      \MessageBreak\MessageBreak
```

```
      ----8<------------------8<----%
      \MessageBreak
      Otherwise, `university' is a compulsory field. Provide
      it in the preamble like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\university{university-name}%
      \MessageBreak\MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak
    }%
  \fi
  \ifdefined\udiss@department
  \else
    \def\udiss@department{\textbf{-NoValidDepartment-}}%
    \ClassError{udiss}{%
      No department given in a normal (non-blind) document%
    }{%
      Do you want a blind document? If yes, then use the
      package option `blind' like\MessageBreak in the
      following:%
      \MessageBreak
      ----8<------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\documentclass[blind]{udiss}%
      \MessageBreak\MessageBreak
      ----8<------------------8<----%
      \MessageBreak
      Otherwise, `department'\space is a compulsory field.
      Provide it in the preamble like in\MessageBreak the
      following:%
      \MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak\MessageBreak
      \space\space\string\department{department-name}%
      \MessageBreak\MessageBreak
      ----8<-------------------------8<----%
      \MessageBreak
    }%
  \fi
```

For printing, hyperref and hyperxmp are irrelevant, so we load them only when the documents are digital.

```
  \ifudiss@print
  \else
    \RequirePackage{hyperref}%
    \RequirePackage{hyperxmp}%
```

We set the metadata here using **\hypersetup** command.

```
    \hypersetup{%
      unicode,%
      colorlinks,%
```

```
      pdftitle           = {\udiss@title}%
    }%
    \ifudiss@subtitle
      \hypersetup{%
        pdfsubject         = {\udiss@subtitle}%
      }%
    \fi
```

We don't set author's name if the user has used the `blind` option as it defeats the whole purpose. If it is not used, we set the copyright information and the license-url. All-rights-reserved doesn't have a specific license-url, so it is used only if the user uses a free culture license.

```
    \ifudiss@blind
    \else
      \hypersetup{%
        pdfauthor          = {\udiss@author},%
        pdfcopyright       = {%
          \udiss@title\textLF
          \udiss@local@copyright\
          ©
          \udiss@copyrightable@years\
          \udiss@author\textLF
          \udiss@license@notice
        },%
      }%
      \ifudiss@free@culture
        \hypersetup{%
          pdflicenseurl      = {%
            \udiss@free@license@url
          }%
        }%
      \fi
    \fi
```

If the color option is not disabled, we load the color variables with `\hypersetup`.

```
    \ifudiss@colors
      \hypersetup{%
        linkcolor        = {\udiss@int@link@color},%
        urlcolor         = {\udiss@int@url@color},%
        citecolor        = {\udiss@int@cite@color}%
      }%
```

Otherwise, we use the `hidelinks` option of hyperref which creates links, but without colors.

```
    \else
      \hypersetup{%
        hidelinks%
      }%
    \fi
  \fi
}
```

Now, after setting everything, we typeset the front matter including title-page, table of contents, optionally list of figures/tables and declaration with the following code.

```
\AddToHook{begindocument/end}{%
  \begingroup
  \thispagestyle{titlingpage}%
  \ifudiss@ftitle
    \setlength{\fboxsep}{25pt}%
    \noindent\fbox{%
      \begin{minipage}%
        [c][\dimexpr\textheight-2\fboxsep-2\fboxrule]%
        [c]{\dimexpr\linewidth-2\fboxsep-2\fboxrule}%
  \fi
        \hspace{0pt}%
        \vspace{3em}%
        \begin{center}
          \begingroup
          \udiss@title@shape
          \ifx\udiss@title@font\empty
          \else
            \fontspec[\udiss@title@fontfeats]%
                     {\udiss@title@font}%
          \fi
          \udiss@title@size
          \color{\udiss@title@color}%
          \udiss@title
          \endgroup

          \medskip

          \ifudiss@subtitle
            \begingroup
            \udiss@subtitle@shape
            \ifx\udiss@subtitle@font\empty
            \else
              \fontspec[\udiss@subtitle@fontfeats]%
                       {\udiss@subtitle@font}%
            \fi
            \udiss@subtitle@size
            \color{\udiss@subtitle@color}%
            \udiss@subtitle
            \endgroup
          \fi

          \vfill

          \ifudiss@blind
          \else
            \begingroup
            \udiss@author@shape
            \ifx\udiss@author@font\empty
            \else
              \fontspec[\udiss@author@fontfeats]%
                       {\udiss@author@font}%
            \fi
            \udiss@author@size
            \color{\udiss@author@color}%
```

```
    \udiss@author
    \ifudiss@pronouns
      \begingroup
      \space
      \addfontfeature{Scale={0.6}}%
      \udiss@pronouns
      \endgroup
    \fi
    \endgroup

  \vspace{2em}%
  \fi

  \begingroup
  \udiss@fulfilment@shape
  \ifx\udiss@fulfilment@font\empty
  \else
    \fontspec[\udiss@fulfilment@fontfeats]%
             {\udiss@fulfilment@font}%
  \fi
  \udiss@fulfilment@size
  \color{\udiss@fulfilment@color}%
  \udiss@fulfilment
  \endgroup

  \vfill

  \begingroup
  \udiss@uni@shape
  \ifx\udiss@uni@font\empty
  \else
    \fontspec[\udiss@university@fontfeats]%
             {\udiss@uni@font}%
  \fi
  \udiss@uni@size
  \color{\udiss@uni@color}%
  \udiss@university
  \endgroup

  \vspace{0.5em}%

  \begingroup
  \udiss@dept@shape
  \ifx\udiss@dept@font\empty
  \else
    \fontspec[\udiss@dept@fontfeats]%
             {\udiss@dept@font}%
  \fi
  \udiss@dept@size
  \color{\udiss@dept@color}%
  \udiss@department
  \endgroup

  \ifudiss@logo
```

```
            \bigskip
            \ifudiss@texlogo
              \udiss@texlogo
            \fi
            \includegraphics[%
              width        = {%
                \udiss@logo@width\linewidth
              },%
              height       = {%
                \udiss@logo@hght\textheight
              }%
            ]{%
              \udiss@logo
            }%
          \fi
        \end{center}
  \ifudiss@ftitle
      \end{minipage}%
    }%
  \fi

  \newpage

  \ifudiss@blind
  \else
    \ifudiss@free@culture
      \thispagestyle{titlingpage}%
      \vspace*{\stretch{1}}
      \begin{quote}
        \setlength{\parindent}{0pt}%
        \begingroup
        \begin{minipage}{\linewidth}
          \bfseries
          \udiss@title

          \udiss@local@copyright\
          ©
          \udiss@copyrightable@years\
          \udiss@author

          \medskip
          \noindent\hrule \linewidth 0.4pt\relax
        \end{minipage}
        \endgroup

        \ifudiss@gfdl
          Permission is granted to copy, distribute and/or
          modify this document under the terms of the GNU
          Free Documentation License, Version 1.3 or any
          later version published by the Free Software
          Foundation; with no Invariant Sections, no
          Front-Cover Texts, and no Back-Cover Texts. A copy
          of the license is included in the section entitled
          \enquote{GNU Free Documentation License}.%
```

```
        \fi
        \ifudiss@ccbysa
          \begin{minipage}{\linewidth}
            \doclicenseThis
          \end{minipage}
        \fi
        \vspace{\stretch{3}}
      \end{quote}
      \newpage
    \fi
  \fi

  \tableofcontents*

  \newpage

  \ifudiss@blind
  \else
    \ifudiss@declaration
      \thispagestyle{empty}%
      \begin{center}%
        {%
          \LARGE\bfseries
          \udiss@decllocal
        }%
      \end{center}%

      \udiss@declaration
    \fi
  \fi

  \newpage

  \ifudiss@lot
    \listoftables
  \fi

  \newpage

  \ifudiss@lof
    \listoffigures
  \fi
  \endgroup
}%
```

The following code adds the GFDL v1.3+ notice at the end of the document if that option was used.

```
\ifudiss@blind
\else
  \ifudiss@gfdl
    \AddToHook{enddocument}{%
      \newpage
      \begingroup
      \pagestyle{plain}%
```

```
      \addcontentsline{toc}%
                      {chapter}%
                      {GNU Free Documentation License}%
      \input{gfdl-tex-1p3}%
      \clearpage
      \endgroup
    }%
  \fi
\fi
```
⟨/class⟩

Our class ended here. Now we start creating the language-support files. The first is the Malayalam language. We use fonts provided at: https://ctan.org/pkg/rit-fonts. Notice that this is done conditionally. If the language is loaded as the main language, the fonts are set as the default fonts using class options, else `malayalam` is passed as the optional argument to `\babelfont`.

⟨*babel-ml⟩

```
\IfLanguageName{malayalam}{%
  \ekvset{udiss}{%
    rmfont               = {RIT-Rachana-Regular.ttf},%
    rmitfont             = {RIT-Rachana-Italic.ttf},%
    rmbffont             = {RIT-Rachana-Bold.ttf},%
    rmbfitfont           = {RIT-Rachana-BoldItalic.ttf},%
    rmfeatures           = {%
      Script               = {Malayalam}%
    },%
    sffont               = {RIT-MeeraNew.ttf},%
    sfitfont             = {RIT-MeeraNew.ttf},%
    sfbffont             = {RIT-tnjoy-bold.ttf},%
    sfbfitfont           = {RIT-tnjoy-bold.ttf},%
    sffeatures           = {%
      Script               = {Malayalam}%
    },%
    ttfont               = {RIT-MeeraNew.ttf},%
    ttitfont             = {RIT-MeeraNew.ttf},%
    ttbffont             = {RIT-tnjoy-bold.ttf},%
    ttbfitfont           = {RIT-tnjoy-bold.ttf},%
    sffeatures           = {%
      Script               = {Malayalam}%
    }%
  }%
}{%
  \babelfont[malayalam]{rm}[%
    IgnoreFontspecFile,%
    Script               = {Malayalam},%
    ItalicFont           = {RIT-Rachana-Italic.ttf},%
    BoldFont             = {RIT-Rachana-Bold.ttf},%
    BoldItalicFont       = {RIT-Rachana-BoldItalic.ttf},%
    \ifLuaTeX
      Renderer             = {HarfBuzz}%
    \fi
  ]{RIT-Rachana-Regular.ttf}
  \babelfont[malayalam]{sf}[%
    IgnoreFontspecFile,%
```

```
    Script                  = {Malayalam},%
    ItalicFont              = {RIT-tnjoy-regular.ttf},%
    BoldFont                = {RIT-tnjoy-bold.ttf},%
    BoldItalicFont          = {RIT-tnjoy-bold.ttf},%
    \ifLuaTeX
      Renderer                = {HarfBuzz}%
    \fi
  ]{RIT-tnjoy-regular.ttf}
  \babelfont[malayalam]{tt}[%
    IgnoreFontspecFile,%
    Script                = {Malayalam},%
    ItalicFont            = {RIT-tnjoy-regular.ttf},%
    BoldFont              = {RIT-tnjoy-bold.ttf},%
    BoldItalicFont        = {RIT-tnjoy-bold.ttf},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
  ]{RIT-tnjoy-regular.ttf}%
}
⟨/babel-ml⟩
```

Marathi requires local numerals and local counters. We provide that here and set Mukta as the default font provided in: https://ctan.org/pkg/EkType-Tanka.

```
⟨*babel-mr⟩
\babelprovide[%
  mapdigits,%
  counters/स्वर = अ आ इ ई उ ऊ ए ऐ ओ औ अं अः ॲ ऋ ऌ ऑ,%
  counters/अंक = एक दोन तीन चार पाच सहा सात आठ नऊ दहा अकरा
  बारा तेरा चौदा पंधरा सोळा सतरा अठरा एकोणीस वीस एकवीस बावीस
  तेवीस चोवीस पंचवीस सव्वीस सत्तावीस अठ्ठावीस एकोणतीस तीस एकतीस
  बत्तीस तेहतीस चौतीस पस्तीस छत्तीस सदतीस अडतीस एकोणचाळीस चाळीस
  एकेचाळीस बेचाळीस त्रेचाळीस चव्वेचाळीस पंचेचाळीस शेहेचाळीस सत्तेचाळीस
  अठ्ठेचाळीस एकोणपन्नास पन्नास एकावन्न बावन्न त्रेपन्न चौपन्न पंचावन्न
  छप्पन्न सत्तावन्न अठ्ठावन्न एकोणसाठ साठ एकसष्ट बासष्ट त्रेसष्ट चौसष्ट
  पासष्ट सहासष्ट सदुष्ट अडुसष्ट एकोणसत्तर सत्तर एकाहत्तर बाहत्तर
  त्र्याहत्तर चौऱ्याहत्तर पंचाहत्तर शाहत्तर सत्त्याहत्तर अठ्ठ्याहत्तर
  एकोणऐंशी ऐंशी एक्याऐंशी ब्याऐंशी त्र्याऐंशी चौऱ्याऐंशी पंच्याऐंशी श्याऐंशी
  सत्त्याऐंशी अठ्ठ्याऐंशी एकोणनव्वद नव्वद एक्याण्णव ब्याण्णव त्र्याण्णव
  चौऱ्याण्णव पंचाण्णव शहाण्णव सत्त्याण्णव अठ्ठ्याण्णव नव्याण्णव शंभर,%
  counters/व्यंजन = क ख ग घ ङ%
  च छ ज झ ञ%
  ट ठ ड ढ ण%
  त थ द ध न%
  प फ ब भ म%
  य र ल व श ष स ह ळ,%
  Alph = अंक,%
  alph = स्वर%
]{marathi}
\def\theenumiii{\localecounter{व्यंजन}{enumiii}}
\def\theenumiv{\localecounter{अंक}{enumiii}}
\IfLanguageName{marathi}{%
  \ekvset{udiss}{%
    rmfont                = {Mukta-Regular.ttf},%
    rmitfont              = {Mukta-Regular.ttf},%
```

```
    rmbffont              = {Mukta-Bold.ttf},%
    rmbfitfont            = {Mukta-Bold.ttf},%
    rmfeatures            = {%
      Script                = {Devanagari}%
    },%
    sffont                = {Mukta-Regular.ttf},%
    sfitfont              = {Mukta-Regular.ttf},%
    sfbffont              = {Mukta-Bold.ttf},%
    sfbfitfont            = {Mukta-Bold.ttf},%
    sffeatures            = {%
      Script                = {Devanagari}%
    },%
    ttfont                = {Mukta-Regular.ttf},%
    ttitfont              = {Mukta-Regular.ttf},%
    ttbffont              = {Mukta-Bold.ttf},%
    ttbfitfont            = {Mukta-Bold.ttf},%
    sffeatures            = {%
      Script                = {Devanagari}%
    }%
  }%
}{%
  \babelfont[marathi]{rm}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Marathi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
  ]{Mukta-Regular.ttf}
  \babelfont[marathi]{sf}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Marathi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
  ]{Mukta-Regular.ttf}
  \babelfont[marathi]{tt}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Marathi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
```

```
      Renderer                = {HarfBuzz}%
    \fi
  ]{Mukta-Regular.ttf}%
}
⟨/babel-mr⟩
⟨*babel-en⟩
\babelfont[english]{rm}[%
  IgnoreFontspecFile,%
  Extension               = {.otf},%
  ItalicFont              = {NewCM10-BookItalic},%
  BoldFont                = {NewCM10-Bold},%
  BoldItalicFont          = {NewCM10-BoldItalic},%
  SlantedFont             = {NewCM10-Book},%
  BoldSlantedFont         = {NewCM10-Bold},%
  Numbers                 = {OldStyle},%
  CharacterVariant        = {6},%
  \ifLuaTeX
    Renderer                = {HarfBuzz}%
  \fi
]{NewCM10-Book}
\babelfont[english]{sf}[%
  IgnoreFontspecFile,%
  Extension               = {.otf},%
  ItalicFont              = {NewCMSans10-BookOblique},%
  BoldFont                = {NewCMSans10-Bold},%
  BoldItalicFont          = {NewCMSans10-BoldOblique},%
  SlantedFont             = {NewCMSans10-Book},%
  BoldSlantedFont         = {NewCMSans10-Bold},%
  Numbers                 = {OldStyle},%
  CharacterVariant        = {6},%
  \ifLuaTeX
    Renderer                = {HarfBuzz}%
  \fi
]{NewCMSans10-Book}
\babelfont[english]{tt}[%
  IgnoreFontspecFile,%
  Extension               = {.otf},%
  BoldFont                = {NewCMMono10-Bold},%
  ItalicFont              = {NewCMMono10-Italic},%
  \ifLuaTeX
    Renderer                = {HarfBuzz}%
  \fi
]{NewCMMono10-Book}
⟨/babel-en⟩
```

Hindi and Marathi share the script, but the language variants Marathi needs are not required by Hindi. So we use the same font, but with a little different settings for Hindi.

```
⟨*babel-hi⟩
\IfLanguageName{hindi}{%
  \ekvset{udiss}{%
    rmfont                = {Mukta-Regular.ttf},%
    rmitfont              = {Mukta-Regular.ttf},%
    rmbffont              = {Mukta-Bold.ttf},%
    rmbfitfont            = {Mukta-Bold.ttf},%
```

```
    rmfeatures            = {%
      Script                = {Devanagari}%
    },%
    sffont                = {Mukta-Regular.ttf},%
    sfitfont              = {Mukta-Regular.ttf},%
    sfbffont              = {Mukta-Bold.ttf},%
    sfbfitfont            = {Mukta-Bold.ttf},%
    sffeatures            = {%
      Script                = {Devanagari}%
    },%
    ttfont                = {Mukta-Regular.ttf},%
    ttitfont              = {Mukta-Regular.ttf},%
    ttbffont              = {Mukta-Bold.ttf},%
    ttbfitfont            = {Mukta-Bold.ttf},%
    sffeatures            = {%
      Script                = {Devanagari}%
    }%
  }%
}{%
  \babelfont[hindi]{rm}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Hindi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
  ]{Mukta-Regular.ttf}
  \babelfont[hindi]{sf}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Hindi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
  ]{Mukta-Regular.ttf}
  \babelfont[hindi]{tt}[%
    IgnoreFontspecFile,%
    ItalicFont            = {Mukta-Regular.ttf},%
    BoldFont              = {Mukta-Bold.ttf},%
    BoldItalicFont        = {Mukta-Bold.ttf},%
    Language              = {Hindi},%
    Script                = {Devanagari},%
    Numbers               = {OldStyle},%
    \ifLuaTeX
      Renderer              = {HarfBuzz}%
    \fi
```

```
  ]{Mukta-Regular.ttf}
}
⟨/babel-hi⟩
```

In this tag, we set three main variables for the `ukerala` dissertation-style. This is just a sample. These style files can contain much more than this. In the experimental release, we start with a small sample. We will be expanding this more.

```
⟨*ukerala⟩
\ekvset{udiss}{%
  university                = {University of Kerala},%
  logo                      = {udiss-logo-ukerala.png},%
  logoheight                = {0.11}%
}
⟨/ukerala⟩
```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

U

GNU Free Documentation License

Version 1.3, 3 November 2008

https://fsf.org/

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document 'free' in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of 'copyleft', which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The '**Document**', below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as '**you**'. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A '**Modified Version**' of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A '**Secondary Section**' is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The '**Invariant Sections**' are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is

not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The '**Cover Texts**' are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A '**Transparent**' copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not 'Transparent' is called '**Opaque**'.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The '**Title Page**' means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, 'Title Page' means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The '**publisher**' means any person or entity that distributes copies of the Document to the public.

A section '**Entitled XYZ**' means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as '**Acknowledgements**', '**Dedications**', '**Endorsements**', or '**History**'.) To '**Preserve the Title**' of such a section when you modify the Document means that it remains a section '**Entitled XYZ**' according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical

measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled 'History', Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled 'History' in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the 'History' section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled 'Acknowledgements' or 'Dedications', Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled 'Endorsements'. Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled 'Endorsements' or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled 'Endorsements', provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of

peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled 'History' in the various original documents, forming one section Entitled 'History'; likewise combine any sections Entitled 'Acknowledgements', and any sections Entitled 'Dedications'. You must delete all sections Entitled 'Endorsements'.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an 'aggregate' if the copyright resulting from the compilation is not used to limit the legal

rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled 'Acknowledgements', 'Dedications', or 'History', the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to

the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License 'or any later version' applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

'Massive Multiauthor Collaboration Site' (or 'MMC Site') means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A 'Massive Multiauthor Collaboration' (or 'MMC') contained in the site means any set of copyrightable works thus published on the MMC site.

'CC-BY-SA' means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

'Incorporate' means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is 'eligible for relicensing' if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME.
>
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled 'GNU Free Documentation License'.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the 'with … Texts.' line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.