IFS LaTeX template documentation

Version 2.0
(28.09.2020)

Contents

Release notes

Date	Description
20.06.2017	`iflang` package has been added to make the language recognition more robust
17.07.2017	Minor fixes in the documentation as well as the template.
13.08.2017	Added the “IVK Header” to the task section of the document. If draft mode is enabled a bold rule placed at the page margin highlights an overfull text lines. Overfull text lines take more horizontal space then available.
05.09.2017	Minor fixes. The English version shows dates in the American format.
26.10.2017	Spelling and sentence corrections. The thesis’s example chapter about tables has been completed. Added the siunitx package.
19.11.2017	Added thesis type `researchproject`, the thesis declaration has been rewritten
28.09.2020	Template revised from scratch, added new functionalities, removed not used or outdated packages, bugfixes

Introduction

This LaTeX template enables you to typeset your seminar project, bachelor's or master's thesis according to the layout requirements of the IFS. The main idea of this template is to make writing your thesis as easy as possible by using LaTeX.

This document is a technical description of the template. It describes how you have to use the template and which guidelines you have to keep in mind.

Alongside this documentation you'll also receive an example thesis. It's prefilled with dummy data, to give you an example on how to use this template.

Most importantly: Stick with this template's structure, requirements and package suggestions you find in this document. Do not alter the template's structure, files, or visual appearance. That way you can focus on writing your thesis and make sure to get a visually appealing document.

This documentation is structured as follows:

- Chapter 2 provides a quick-start guide. You will learn what LaTeX is and how to install and use it. A local as well as a portable installation are described.
- Chapter 3 describes the IFS template in detail.
- Chapter 4 states important guidelines and requirements regarding the use of the template.
- Chapter 5 introduces some helpful LaTeX packages that might be helpful.

Quick-start guide

In the following chapter you'll learn what LaTeX is and how to use it on Windows. Please note that this guide focuses on how to setup your personal LaTeX installation. You're advised to look up your questions regarding LaTeX by yourself. Have a look at the following resources:

- Wikibooks: A good tutorial on nearly anything related to LaTeX. If you aren't familiar with LaTeX yet, always have a look on this tutorial first.
- TeX Exchange: A good questions and answers website. Ask your questions here or search for answers to questions already asked.
- CTAN: All LaTeX packages are hosted here. This is the right place to search for package documentations. Search on CTAN for the appropriate package documentations to get more insights on how to use these packages.

What is LaTeX?

Unlike Word, LaTeX is a typesetting system. That means that you can't directly alter the appearance of your document. Instead, you have to use suitable commands. Writing a LaTeX document actually equals “coding” your document like writing a program in Java or MATLAB. You can compile your LaTeX code into a PDF at any time with the so called `pdflatex` compiler. The compiled PDF doesn’t contain the commands you entered. Instead, it will show the visual results these commands produce in (like headers, paragraphs, tables, formulas and so on). That means that if you want to know how your document looks like you have to compile it first/every time.

The advantages of LaTeX are that you can easily write complex formulas and don't have to bother on how and where to place images and so on. LaTeX takes care of the difficult, time consuming stuff for you and results in good looking documents. Nearly every scientific paper uses LaTeX nowadays.

Installation guide

The following section describes how to use LaTeX on Windows. You have two possibilities:

1. Local installation: Install LaTeX on your hard drive. This requires administration privileges and isn't suitable for publicly available computers.
2. Portable installation: Install LaTeX on your USB drive or home directory so you can use it on every computer that runs Windows 7 or higher.

Local installation

MiKTeX Windows LaTeX distribution

For Windows, MiKTeX is the most popular LaTeX distribution. A LaTeX distribution contains all relevant programs required for creating LaTeX documents. A distribution comes with many if not most packages that add additional features to LaTeX. When writing your documents in LaTeX you’ll always rely on such a distribution.

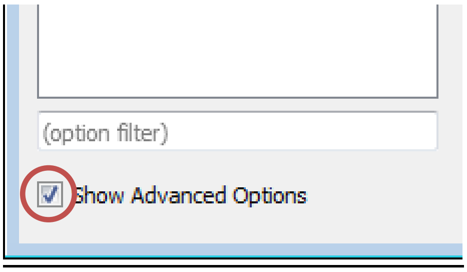
Open https://miktex.org and download “Basic MiKTeX Installer”. Make sure you check the option Install packages on the fly during the installation process.

Linux users can use TeX Live instead. Many popular Linux distributions include it in their default package repositories. For macOS you can use the MacTeX distribution.

TeXstudio editor

Next, you need a LaTeX editor. One popular choice is TeXstudio. You can get it here http://www.texstudio.org. It provides you with proper syntax highlighting and makes calling the `pdflatex` compiler simple. Once you've installed it, open the menu Options » Configure TeXstudio... Make the following changes:

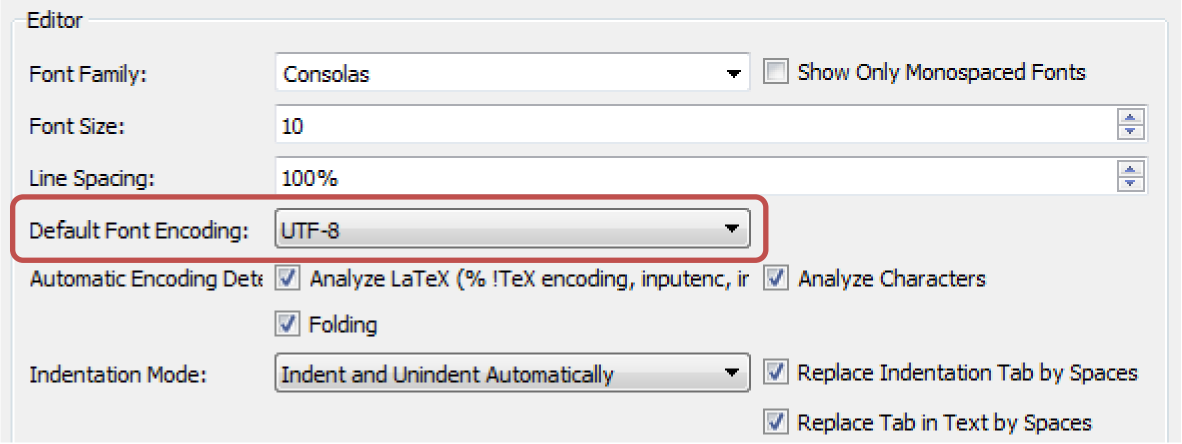
1. First, select the check box Show Advanced Options in the bottom left.



(option filter)

☑ Show Advanced Options

2. Next, open the Editor section and select UTF-8 as Default Font Encoding. This will ensure that special characters in your document (like umlauts) will be displayed correctly.



Editor

Font Family: Consolas ☐ Show Only Monospaced Fonts

Font Size: 10

Line Spacing: 100%

Default Font Encoding: UTF-8

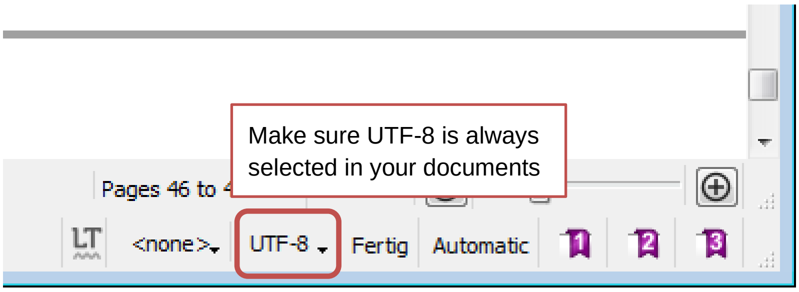
Automatic Encoding Det€ ☑ Analyze LaTeX (% !TeX encoding, inputenc, ir ☑ Analyze Characters

☑ Folding

Indentation Mode: Indent and Unindent Automatically ☑ Replace Indentation Tab by Spaces

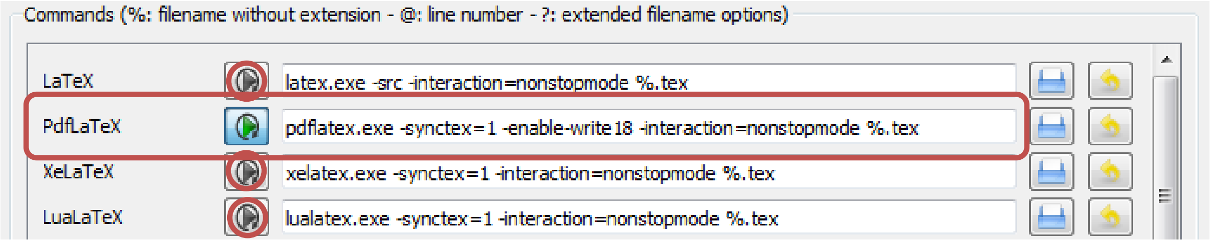
☑ Replace Tab in Text by Spaces

In the future make sure to always create new LaTeX source files within TeXstudio to ascertain that all your files are encoded using UTF-8. Otherwise you may get irreproducible errors. You can check the encoding of your files with any good text editor like Notepad++ or TeXstudio itself. TeXstudio shows the current encoding in the bottom right of the program's window and allows you to change it by double clicking in the current encoding.


Make sure UTF-8 is always selected in your documents

3. Next, select the Commands section. Make sure the button besides the label PdfLaTeX is toggled (if toggled the triangle will turn green). You can un-toggle every other button. Now, enter the following line of text into the text field:

```
pdflatex.exe -enable-write18 -synctex=1 -interaction=nonstopmode %.tex
```


Commands (%: filename without extension - @: line number - ?: extended filename options)

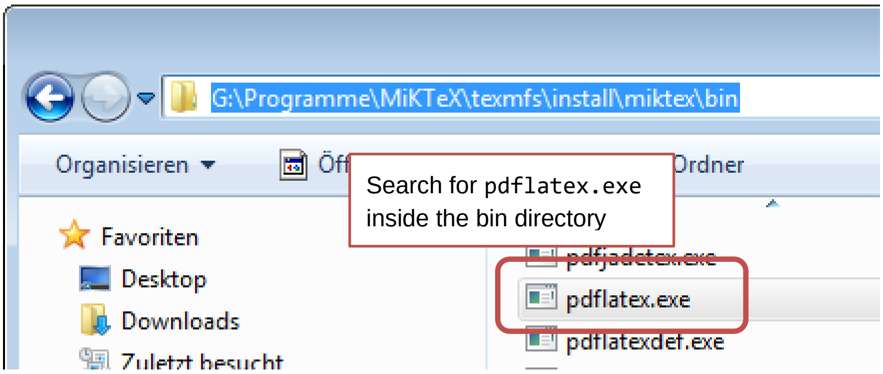
LaTeX: latex.exe -src -interaction=nonstopmode %.tex

PdfLaTeX: pdflatex.exe -synctex=1 -enable-write18 -interaction=nonstopmode %.tex

XeLaTeX: xelatex.exe -synctex=1 -interaction=nonstopmode %.tex

LuaLaTeX: lualatex.exe -synctex=1 -interaction=nonstopmode %.tex

4. Finally, open the Build section and let TeXstudio know where it can find the programs used during the build process. The path you have to enter here depends on where you have installed MiKTeX. Locate your MiKTeX installation directory in `C:\Program Files` or `C:\Program Files (x86)`. Inside this directory navigate through the following subdirectories `texmfs\install\miktex\bin`. If you can find `pdflatex.exe` inside `bin` you're done. Copy the current path


G:\Programme\MiKTeX\texmfs\install\miktex\bin

Search for `pdflatex.exe` inside the bin directory

and paste it into the text field besides the label that displays Commands (\$PATH).

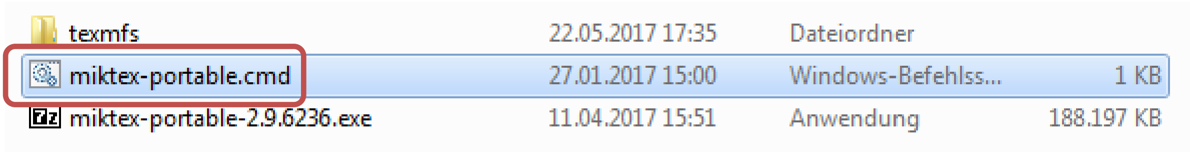
Finally, apply the changes by clicking on OK.

Portable installation

Portable MiKTeX distribution

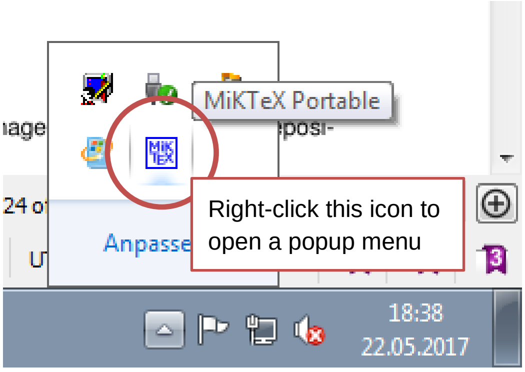
If you want to have a portable MiKTeX installation instead, go to https://miktex.org/portable and download MiKTeX portable. Once you have downloaded it, extract it onto your USB drive. This will take a while. (You should use an USB-3.0 drive or newer otherwise this may take very long.)

Once the installation process is done execute the following file



texmfs	22.05.2017 17:35	Dateiordner	
miktex-portable.cmd	27.01.2017 15:00	Windows-Befehlss...	1 KB
miktex-portable-2.9.6236.exe	11.04.2017 15:51	Anwendung	188.197 KB

inside your newly created MiKTeX installation directory. In the Windows task bar a new icon will show up

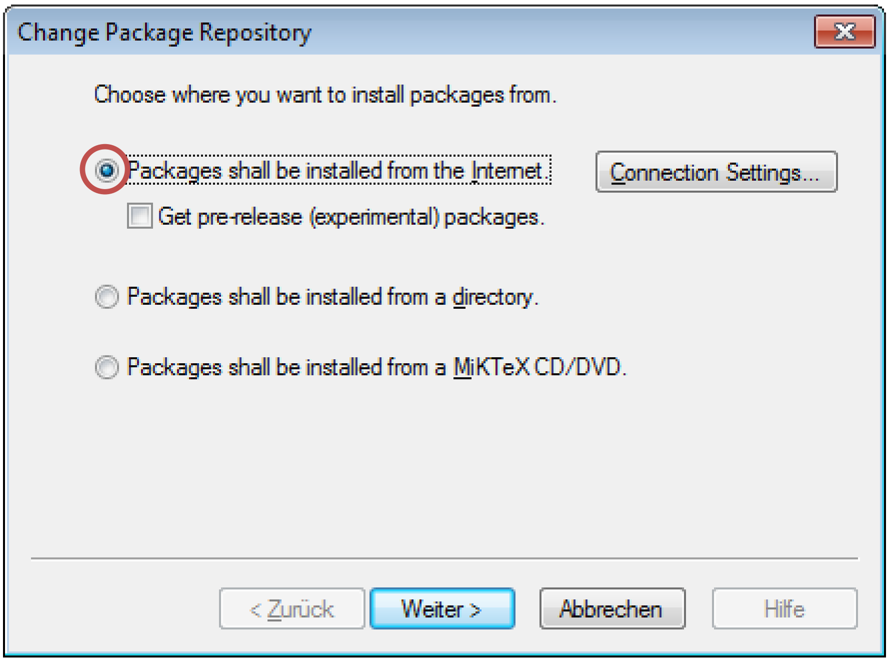


Right-click this icon to open a popup menu

Right-click this icon and select MiKTeX Options. Select the tab Packages and then select the button Change.

At this point you have to carefully choose between two options:

- Install new packages from the internet: This only works if you have free access to the internet. In this case select the first radio button



Change Package Repository

Choose where you want to install packages from.

● Packages shall be installed from the Internet.

□ Get pre-release (experimental) packages.

○ Packages shall be installed from a directory.

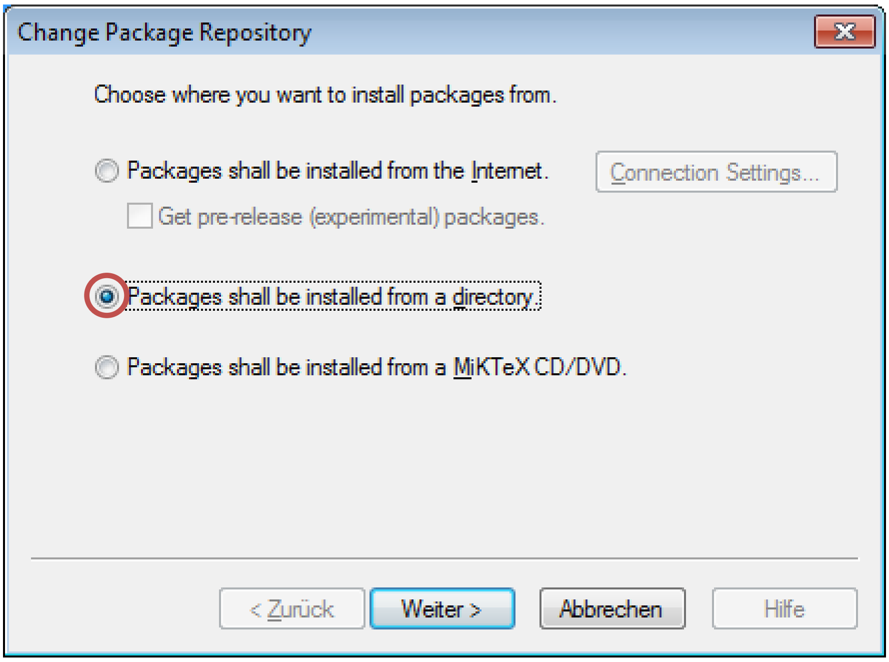
○ Packages shall be installed from a MiKTeX CD/DVD.

and chose a suitable host in the list in the next window (preferably one that is located in Germany).

However, some companies may block certain network traffic. In this case installing new packages over the Internet will fail and you have to stick with the second option below.

- Install new packages from a local repository: If you do not have free access to the Internet or you want to be able to edit your LaTeX documents without being online all the time, you need to download the proTeXt distribution that comes with nearly every LaTeX package.

  Go to https://www.tug.org/protext/ and click the link that shows “download the self-extracting protext.exe file”. Note that this file requires a lot of free space on your USB drive (4 GB of free space is recommended). Extract the proTeXt distribution into an arbitrary directory on your USB drive. Again, this will take a while. Once you're done go back to the MiKTeX options dialog you opened previously and select the second radio button



Change Package Repository

Choose where you want to install packages from.

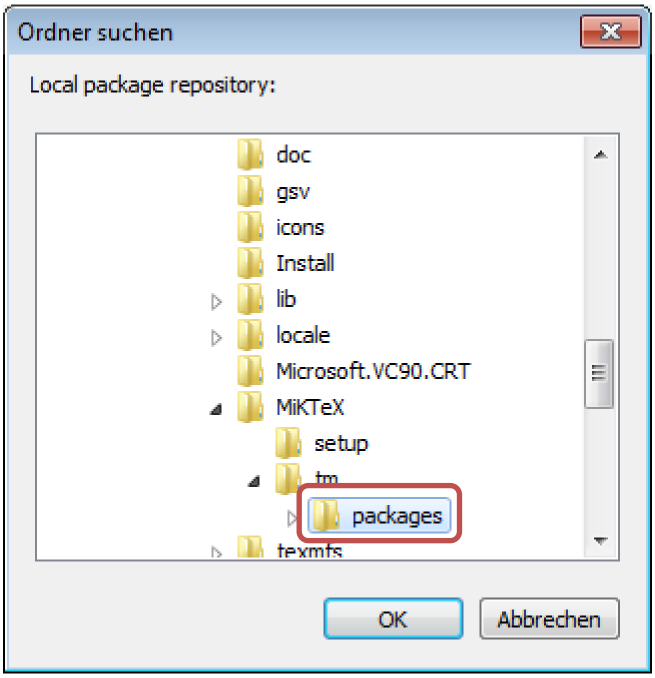
○ Packages shall be installed from the Internet.

□ Get pre-release (experimental) packages.

● Packages shall be installed from a directory.

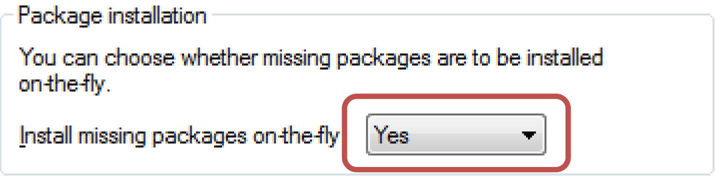
○ Packages shall be installed from a MiKTeX CD/DVD.

In the next window, you to have to browse to the directory inside the proTeXt distribution that contains the LaTeX packages. For example, if you have installed proTeXt in `E:\ProTeXt` the path is `E:\ProTeXt\MiKTeX\tm\packages`:


Ordner suchen

Local package repository:

After you've configured your desired source of packages, select the tab General of the MiKTeX Options dialog. Choose to install packages on the fly


Package installation

You can choose whether missing packages are to be installed on-the-fly.

Install missing packages on-the-fly: Yes

and click once on the button Refresh FNDB and the button Update Formats.

Portable TeXstudio editor

You can find a portable version of TeXstudio here http://www.texstudio.org/#download. After you're done installing MiKTeX portable follow the same steps as described above to properly configure TeXstudio, see the section TeXstudio editor. The only difference is that you have to locate your MiKTeX installation on your USB drive.

Troubleshooting

If you get unexpected compilation errors check that your packages and your MiKTeX distribution is up to date. Updating MiKTeX depends on your type of installation:

- If you have chosen to install MiKTeX you can run the program "Update MiKTeX" from the Windows start menu.
- If you have chosen to install MiKTeX on your USB drive you can start MiKTeX portable, right-click the MiKTeX icon in the Windows task bar and select Update MiKTeX.

Sometime you have to run the updater twice to get all updates.

How to use the IFS template

This chapter describes the usage of the IFS template in detail. The template's structure and functionality are introduced as well as some required and recommended packages. Required packages means that this template is built around some specific packages. The template relies on you to use them.

What's the purpose of this template?

The template implements the IFS design guidelines and comes with a lot of predefined commands that support you in filling in your personal and thesis-related data.

The template has a predefined structure of specific files and directories. This structure is fixed. Hence, you have to stick with it. Your first step is to understand this structure so that you know where to enter, place, or find specific parts of your thesis.

Please keep in mind to not change or remove any files of the template. Otherwise you may break its functionality. **To avoid confusions, always ask your advisor first.**

Structure of the template

This section describes the files as well as the directories that are part of the IFS template. In the root directory of the template you'll find the following predefined files 🗏:

- 🗏 **thesis.tex**: This file is the starting point of your thesis. Inside this file you enter your personal data and include your chapters.
- 🗏 **references.bib**: This file is going to contains all your references. Because you just started writing your thesis it's still empty. You can open it with any text editor or with a reference management application like JabRef. The template uses BibTeX. See https://en.wikibooks.org/wiki/LaTeX/Bibliography_Management#BibTeX on how to write proper BibTeX entries. Make sure this file retains its initial UTF-8 encoding when editing with a third-party text editor.
- 🗏 **sbmthesis.cls**: This file contains the IFS template. You shouldn't edit this file otherwise you may alter the appearance of your thesis or break its functionality.
- 🗏 **plaindin.bst**: This file controls how your bibliography looks. According to the IFS layout guidelines the look is given by the DIN 1505. You shouldn't edit this file at all.

Next, you'll also find the following predefined directories 🗀:

- 🗀 **chapters**: Place your chapter files here (one file per chapter).
- 🗀 **data**: Place all your data (like images, source code, plots, …) here. The following subdirectories have already been created for you and may support you organize your data:
    - 🗀 **img**: Place your raster graphics here (like JPG- or PNG-files).
    - 🗀 **scalable**: Place your scalable graphics here (like SVG- or PDF-files).
    - 🗀 **datasheets**: Place your data sheets here. Data sheets are files that describe a technical tool, a rig, … They're typically a PDF.
    - 🗀 **plot**: Place files that show plots here.

    - 🗀 **code**: Place source code you want to display in your document here. You can also use links to your real code files.

  If you stick with this structure, it is easier for you to find certain files at a later stage during your project. You can of course create other directories, but you should always place them in 🗀 data. However, avoid placing your files in the root directory.

- 🗀 **mandatory**: Inside this directory you find the following predefined empty source files:
    - 🗎 **abstract_de.tex**: Enter your abstract in German language here.
    - 🗎 **abstract_en.tex**: You have to enter your abstract in English language here.
    - 🗎 **acronyms.tex**: Put your acronyms here.
    - 🗎 **symbols.tex**: Put your symbols here.
    - 🗎 **titlepage.pdf**: Replace this file by the pages given to you by your advisor.

  The abstracts and the thesis's task are mandatory inputs. Except for the thesis's task, do not use any commands that alter your text's style (bold or italic letters, section headers, …).

- 🗀 **optional**: Inside this directory you'll find the following predefined optional empty source file:
    - 🗎 preamble.tex: If you want to add a preamble to your thesis enter it here.

  Optional inputs aren't included by default (i.e. you may enter your preamble, but you have to call the template to actually show your input).
  See the comments inside 🗎 preamble.tex for more details.

- 🗀 **template**: This directory contains files used by the IFS template.

How to configure the template

The template comes with a collection of different commands that allow you to configure your thesis. This includes adding titlepages, preamble and so on. This section describes every available command.

Note that you’ll notice that many of the commands that will be introduced below are already part of your 🗎 thesis.tex document. You only have to fill in the command’s arguments.

Class options

The IFS template LaTeX class requires the input of the options given in Table 1. In case the value of an option is invalid an error is thrown and the compilation process will be stopped.

Option	**Value**	**Description**
`language`	`de, en`	Sets the main language of the document. This option controls the language of the predefined structures.
`layout`	`oneside, twoside`	Default is twoside and must not be changed.
`titlepage`	`true`, `false`	Switches between including titlepage or not.
`preamble`	`true`, `false`	Switches between including preamble or not.
`draftmode`	`true`, `false`	Switches between draft mode and final mode

Table 1: Overview of the sbmthesis class options.

Setting the option `layout=oneside` switches to one-sided layout. Please do not change without permission of your advisor. Per default the layout is defined as two-sided.

The Booleans for `titlepage` and `preamble` allow to specify whether you want to include the titlepage or a preamble. The options are set to `false` by default.

In draft mode, bold markers placed at the page margin highlight overfull text lines. In case LaTeX can't hyphenate a long/technical term create a hyphenation rule for it. See the chapter Custom hyphenation rules for more details. You should always fix overfull text lines because the text overflow isn't visually appealing. Furthermore, when draft mode is enabled, instead of including images only frames are drawn. This might be useful, especially if you have a lot of images. When you're done writing, set `draftmode=false`. With draft mode disabled, an error will be thrown in case something crucial needs to be fixed. In this case, see the error message for more details.

How to use the IFS template

The following section gives you an overview on how to use the IFS template.

Enter your personal data

First, fill in your personal and thesis-related data. You have to do this inside 🗎 thesis.tex. You'll notice that this file already contains all relevant command calls. However, the command's arguments are missing and have to replace the placeholders with your data. See the additional comments in 🗎 thesis.tex for more details.

Mandatory inputs

Next, you have to fill in your abstract in German and English language and your thesis's task. The abstracts and the thesis's task are mandatory inputs. Enter your German abstract in 🗎 mandatory/abstract_de.tex and your English abstract in 🗎 mandatory/abstract_en.tex. If you're not yet done with your abstracts, skip this section for now and come back later.

If you already have a finished task description in the PDF format head over to the next section to see how you can use this file instead. Otherwise you have to rewrite your description using LaTeX.

Optional inputs

Preamble

If you want to add a preamble set the option `preamble=true`. This will show the content of 🗎 optional/preamble.tex before the first chapter. Keep your preamble short and only use paragraphs to arrange your content (do not enter a chapter or sections).

How to add new chapters

In order to add new chapters to your thesis create an LaTeX source file and place it in the directory 🗀 chapters. In the examples below the new file is called 🗎 01_introduction.tex. Next, open the 🗎 thesis.tex file that comes with the IFS template and add your chapter like it is shown below:

```
% +----------------------------+
% | Include your chapters here |
% +----------------------------+
% Insert \cleardoublepage after every include
\include{chapters/01_introduction}
\cleardoublepage
```

The first line with the `\include` command call adds your chapter to your document. Note that you have to omit the file extension. The second line makes sure that if you add another chapter to your document, the new chapter will always appear on an odd/right page. Do not break with this common typographic rule.

If you add more and more chapters your code may look like:

```
% +----------------------------+
% | Include your chapters here |
% +----------------------------+
% Insert \cleardoublepage after every include
\include{chapters/01_introduction}
```

```
\cleardoublepage
\include{chapters/02_modelling}
\cleardoublepage
\include{chapters/03_control}
\cleardoublepage
```

You may also want to add some chapters to your thesis's appendix. Create a new LaTeX source file and place it in 🗀 chapters. Next, inside 🗎 thesis.tex, look out for the line that says “Include your appendix's chapters here” and add your chapter like described above. An example is given in the code listing below. In this example 🗎 chapters/a_formulas.tex and 🗎 chapters/a_formulas.tex are added to the thesis's appendix.

```
% +--------------------------------------+
% | Include your appendix's chapters here |
% +--------------------------------------+
% Insert \cleardoublepage after every include
\include{chapters/a_formulas}
\cleardoublepage
\include{chapters/b_proofs}
\cleardoublepage
```

Guidelines and requirements of the template

In the following chapter guidelines regarding

- the creation of custom hyphenation rules,
- bibliography,
- acronyms or abbreviations,
- quotation marks
- and source code listings.

will be introduced.

Read this section with care: You have to stick with these guidelines because the template is designed around the use of some specific LaTeX packages. Hence, you have to use the commands these packages or the template provide.

Custom hyphenation rules

In some special cases LaTeX is unable to correctly hyphenate a term (you may notice this when a term is “to long” and overlaps with the margins of the page). In this case you have to provide a custom hyphenation rule by your own. Look out for the part in your document's 🗎 thesis.tex preamble that says

```
% +----------------------------+
% | Add hyphenation rules here |
% +----------------------------+
\hyphenation{
   ...
}
```

and add your hyphenation rules here. An example: Assume that LaTeX does not know how to hyphenate “Doppelkupplungsgetriebe”. You define a hyphenation rule by entering minus signs in-between the letters of a term everywhere the hyphenation is allowed. Applied to the example a hyphenation rule could look like `Doppel-kupplungs-ge-triebe`. Insert a blank or new line to separate different hyphenation rules. See the attached sample thesis for some examples.

Bibliography

The IFS template uses BibTeX to create the bibliography. Your bibliography will be generated based on your entries in 🗎 references.bib. You can open this file with any text editor. The structure of this file is as follows:

- A new entry starts with the `@`-sign followed by a template specifier. See this Wikibooks article for an overview of predefined templates. Typically, you have to use `@article` (for articles published in a journal or magazine), `@book` (for books),

`@conference` (for conference papers), or `@misc` (for other kinds of publications like web resources).

- Every template has some required fields (key/value pairs) that you have to enter.
- The first and most important field has no key since it's the key that you'll be using to reference an individual entry in your LaTeX code.

See the attached sample thesis for examples on how to define new bibliography entries in 🗎 references.bib and how to reference an entry in your LaTeX document.

Note that if you try to reference newly added entries you may get `[?]`. In this case delete the file 🗎 thesis.bbl in the root directory of your template and compile your document twice. This will most likely fix the problem. If the question mark persists, check if the key is correct. Make sure that you understand how to generate a bibliography (e.g. see the mentioned Wikibooks article) and that you run bibtex.

The appearance of your references is part of the IFS design guidelines and cannot be changed.

Acronyms or abbreviations

Add your acronyms or abbreviations to 🗎 mandatory/acronyms.tex using the required acro package. One big advantage of this package over the more common acronym package is that the list of acronyms is automatically sorted.

The package's documentation is well written and the usage straight forward. Thus, only a few commands are presented at this point. See the attached sample thesis for even more examples.

How to define acronyms

The basic command for defining a new acronym is `\DeclareAcronym{<id>}{<options>}`. The options are key-value pairs separated with a =-sign. The most important options are given in Table 2.

Option	**Description**
`short`	The short form of the acronym (mandatory).
`long`	The long form of the acronym (mandatory).
`short-plural`	The plural ending appended to the short form (default: s).
`short-plural-form`	If defined replaces `short-plural`.
`long-plural`	The plural ending appended to the long form (default: s).
`long-plural-form`	If defined replaces `long-plural`.
`foreign`	Defines a description in a foreign language.
`foreign-lang`	Defines the language of the description given with `foreign`.

Table 2: Overview of some options of the acro package.

The code listing below gives some examples on how to define new acronyms.

```
\DeclareAcronym{cd}{
    short        = CD,
    long         = Compact Disc
}

\DeclareAcronym{sw}{
    short        = SW,
    long         = Seitenwelle,
    long-plural  = n
}

\DeclareAcronym{ecu}{
    short        = ECU,
    long         = Steuergerät,
    foreign      = Electronic Control Unit,
    foreign-lang = English
}
```

How to reference acronyms

After an acronym has been defined, you can use it with the `\ac{<id>}` command. If you want to control the appearance (i.e. short or long version) by yourself use `\acs{<id>}` to force the display of the short version or `\acl{<id>}` for the long version.

If you want to display the plural form instead use `\acp{<id>}`. Use `\acsp{<id>}` to force the display of the short plural form or `\aclp{<id>}` for the long version. The code listing below shows some examples.

```
\ac{cd}     % displays Compact Disc (CD)
\ac{cd}     % on the second call displays CD
\acsp{sw}   % displays SWs
\acp{sw}    % displays Seitenwellen (SWs)
\ac{ECU}    % displays Steuergerät (Electronic Control Unit, ECU)
\ac{ECU}    % on the second call displays ECU
```

Note that you have to compile your document twice before a new acronym appears in the text. Also note that only acronyms that have been referenced at least once will appear in the list of acronyms.

Quotation marks

Use the csquotes package for entering quotation marks. To put your text in quotation marks use the `\enquote{<text>}` command. The marks will be set depending on the document's main language. Do not explicitly write quotation marks by yourself.

Example: The code `\enquote{in a nutshell}` results in “in a nutshell”.

Source code listings

The template defines a default style for your source code listings based on the listings package. This includes color definitions for comments or keywords.

The listings package allows for adding inline source code, i.e. source code that appears in your running text, by using the `\lstinline|<your code>|` command. This command is pre-configured to automatically break long lines. If a line break occurs, the symbol ↳ will be inserted so that the reader knows the line of codes spans over multiple lines of text. The two vertical line characters `|<your code>|` help `\lstinline` determine the start and end of your inline code and are not allowed in-between. Otherwise the source code recognition will fail.

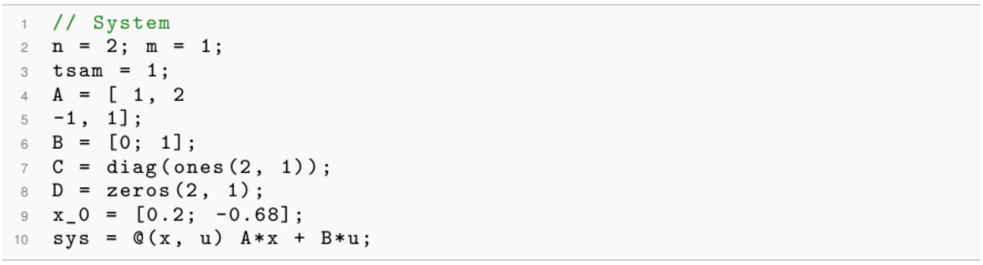
Besides displaying inline code you can display larger pieces of source code with the `lstlisting` environment:

```
\begin{lstlisting}[<options>]
<your code here>
\end{lstlisting}
```

The IFS template already includes a default style. In order to have a working syntax highlighting the only thing you have to do is define the language of the source code. Use the `language=<language>` option to do that. Supported languages are `Matlab`, `C`, `C++`, `Java`, `Python`, `R`, `XML` and many more.

To add a caption to your listing use the option `caption=<your caption>`. To add a label so you can reference your listing, use the `label=<your label>` option.

The code listing below shows some Matlab code.



```
// System
n = 2; m = 1;
tsam = 1;
A = [ 1, 2
-1, 1];
B = [0; 1];
C = diag(ones(2, 1));
D = zeros(2, 1);
x_0 = [0.2; -0.68];
sys = @(x, u) A*x + B*u;
```

Listing 1.1: Some Matlab sample code.

The Matlab sample code above uses the following options: `language=Matlab`, `caption=Some Matlab sample code.`, and `label=lst:SampleMatlabCode.` The full source code is

```
\begin{lstlisting}[language=Matlab,caption=Some Matlab sample code.,label=lst:SampleMatlabCode]
% System
n = 2; m = 1;
...
\end{lstlisting}
```

The label is used to reference the source code listing using `\ref{<label>}`.

Other valid language key values are: Java, Python, C, C++, R, XML. See the listings package documentation for more detals.

Additional commands of the IFS template

The template can be extended with some more commands that are probably very useful to you. However, you don't have to use these commands.

Math related commands

The template includes some useful commands that may be useful when you're writing mathematical expressions. The commands only work in math mode (inline or environment).

Command	Description
`\I{<dim>}`	Displays the unit/identity matrix. If the dimension `dim` is 1, displays a 1 instead. Example: `\I{2}` results in $\boldsymbol{I}_2$, `\I{1}` results in 1.
`\itrafo`, `\trafo`	Displays the transformation (Fourier, Laplace, …) symbols $\circ\!\!-\!\!\bullet$ and $\bullet\!\!-\!\!\circ$.
`\Re{<num>}`, `\Im{<num>}`	Real and imaginary part of an imaginary number.
`\Rang{<num>}`, `\rank{<num>}`, `\diag{<num>}`, `\sign{<num>}`, `\Spur{<num>}`, `\Kern{<num>}`, `\trace{<num>}`	Some common math functions. No further explanation needed.
`\T`	Displays the transpose symbol. Example: `\bm{A}^\T` results in $\boldsymbol{A}^\top$.

If you want to use these commands, add the following lines of code in the file sbmthesis.cls:

```
% --------------------------------
%%% Load additional packages here

% Custom math operators

\newcommand*{\Rang}[1]{\operatorname{Rang}\del{#1}}

\newcommand*{\rank}[1]{\operatorname{rank}\del{#1}}

\newcommand*{\diag}[1]{\operatorname{diag}\del{#1}}

\newcommand*{\sign}[1]{\operatorname{sign}\del{#1}}

\newcommand*{\Spur}[1]{\operatorname{Spur}\del{#1}}

\newcommand*{\Kern}[1]{\operatorname{Kern}\del{#1}}

\newcommand*{\trace}[1]{\operatorname{trace}\del{#1}}

\let\Re\relax\newcommand*{\Re}[1]{\operatorname{Re}\del{#1}}

\let\Im\relax\newcommand*{\Im}[1]{\operatorname{Im}\del{#1}}

\newcommand{\T}{\top}
```

```
% Math symbols
\DeclareSymbolFont{symbolsC}{U}{ntxsyc}{m}{n}
\SetSymbolFont{symbolsC}{bold}{U}{ntxsyc}{b}{n}
\DeclareMathSymbol{\itrafo}{\mathrel}{symbolsC}{23}
\DeclareMathSymbol{\trafo}{\mathrel}{symbolsC}{24}

% Identity matrix
\newcommand{\I}[1]{%
    \def\variable{1}%
    \ifnum #1 > \variable%
    \ifmmode%
    \bm{I}_{#1}%
    \else%
    $\bm{I}_{#1}$%
    \fi%
    \else%
    1%
    \fi%
}
```

Useful LaTeX packages

This chapter presents some useful packages. The packages are **not** included in the template and have to be loaded if you want to use the commands.

Special symbols, menu keys and file paths

If you want to include a special symbol like 🔌 or 👥 have a look at the fontawesome package and the amssymb package. The latter one primarily contains math related symbols whereas the first one contains common web icons. Please note that these packages have to be loaded if needed.

The menukeys package is pretty useful for printing keystrokes, menu sequences or file paths. If you want to use it add the following lines of code in the file 📄 sbmthesis.cls:

```
% key strokes and menu paths
\RequirePackage[os=win]{menukeys}
\renewmenumacro{\directory}{hyphenatepaths}
```

Then you can use:

- **Keystrokes**: If you describe a program and want to highlight a key combination use the `\keys` command. Example: `\keys{\ctrl+C}` results in Ctrl+C. The package also includes some additional symbols like `\return` which results in ↲.
- **Menu sequences**: If you want to describe a sequence of menus use the `\menu` command. Example: `\menu[,]{Extras,Settings,{Units, rulers and origin}}` results in Extras › Settings › Units, rulers and origin. Note that the optional parameter `,` in square brackets defines the separator token between different entries.
- **File paths**: If you want to link to a certain file path use the `\directory` command. Example: `\directory[/]{data/resources/F91-W.pdf}` results in `data▸resources▸F91-W`. Note that the optional parameter `/` in square brackets defines the separator token.

Hyperlinks

For hyperlinks and URLs use the hyperref package (already loaded). Amongst others, it includes the following useful commands:

- `\url{<my_url>}`: Adds a clickable URL to your text using a typewriter font.
- `\href{<my_url>}{<description>}`: Adds a clickable URL to your text and hides it behind a custom description. If you want to use a typewriter font for the description you have to wrap your text using `\texttt{<description>}`.

How to add additional packages

You can add additional packages in the document's preamble below the line that says

```
% +--------------------------------------------------------+
% |               Include other packages here              |

% |                          or                            |

% | at the intended position in the cls file (prefered)    |
% +--------------------------------------------------------+
🗎 thesis.tex: \usepackage{<package name>}

🗎 sbmthesis.cls: \RequirePackage{<package name>}
...
```

Keep in mind that your packages may interfere with packages that have already been loaded. Also make sure that you do not load a package which has been already loaded in the document class file.