UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

An introduction to LaTeX

during the Student Days of April 2024

(based on the introduction of Arno Trautmann & Robert Weis)

Tamasi Kar
Physikalisches Institut, Universität Heidelberg

08. - 12.04.2024

1. The Name of the Game
2. Introduction and Basic Usage of LaTeX
3. Changing the look: page dimensions and fonts
4. Cross references
5. Typesetting of Mathematics
6. Figures and tables
7. Writing a thesis
8. Typographic details
9. Presentations
10. Advanced topics: Lua code, defining macros, ...

1. The Name of the Game
2. Introduction and Basic Usage of LaTeX
   - A complete LaTeX system
   - LaTeX in a nutshell
   - How to use LaTeX
   - Auxiliary programs, the command line
   - Classes, packages and documentation
   - Basic document structures
   - Lists and enumerations
3. Changing the look: page dimensions and fonts
   - Page dimensions and type area
   - Headers and footers with `scrlayer-scrpage`
   - So far ...
   - Font switching
   - Font packages
   - Loading system fonts

TeX and LaTeX?

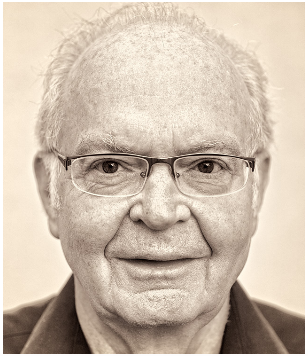
The Name of the Game

- `$\tau,\epsilon,\chi$`: τ, ϵ, χ
- `\TeX`: TeX
- `\LaTeX`: LaTeX

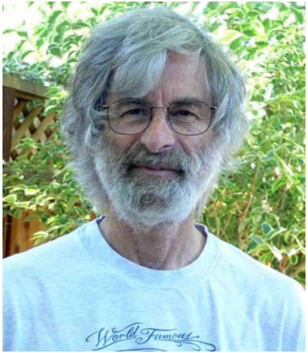
What is LaTeX?

- a typesetting system based on TeX (remember: $\tau\epsilon\chi$)
- TeX was developed by Don E. Knuth
- TeX is both the program and the macro language
- LaTeX is a large macro package for TeX (first developed by Leslie **La**mport)
- new variants of TeX evolved (encodings, output formats, ...)
- most widespread version is pdfTeX
- latest successor: LuaTeX, with many improvements
  (useable for everyday production, but still not very common)

Knuth / Lamport



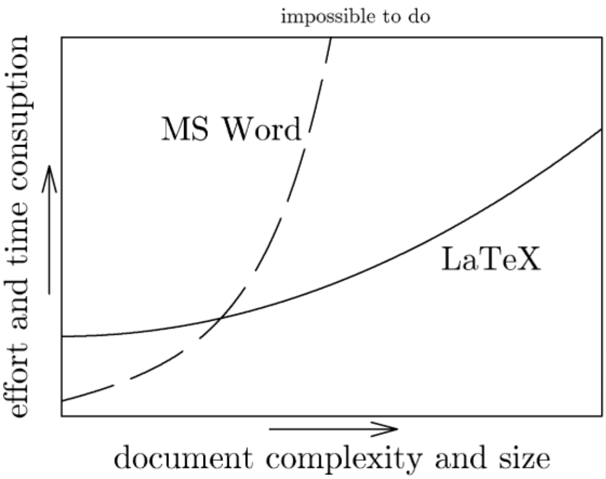
Donald Knuth



Leslie Lamport

1. The Name of the Game
2. Introduction and Basic Usage of LaTeX
   - A complete LaTeX system
   - LaTeX in a nutshell
   - How to use LaTeX
   - Auxiliary programs, the command line
   - Classes, packages and documentation
   - Basic document structures
   - Lists and enumerations
3. Changing the look: page dimensions and fonts
   - Page dimensions and type area
   - Headers and footers with `scrlayer-scrpage`
   - So far ...
   - Font switching
   - Font packages
   - Loading system fonts

Why LaTeX?



http://www.pinteric.com/miktex.html

Why LaTeX?

- math-Mode
- free availability on (nearly) all platforms
- portability
- no viruses
- high compatibility (over many years)
- no defects in source files
- extremely stable
- easy to edit with simple editors
- small files even on large projects
- separation of text, pictures, layout, …

Distributions and installation

- many programs and even much more files are needed to work with LaTeX
- packed together as distribution
- MiKTeX for windows (new versions also for linux)
- TeX Live for all platforms
- MacTeX for Mac OS X (based on TeX Live)
- all distributions take their files from CTAN (Comprehensive TeX Archive Network)
- TeX Live is strictly free software only $\Longrightarrow$ TLContrib as additional source (also for unstable packages)

Editors

- LaTeX source code is typically written in specialised editors
- TeX works: small editor with clear interface for easy TeXing
- emacs, vim: very powerfull editors with LaTeX-plugins
- kile, TeXmaker, TeXstudio, TeXnicCenter, ...
- important: the editor should use utf8 encoding to avoid encoding problems!

Character sets

- UTF-8, ISO-8859-1, latin, ANSI, ASCCI
- it all began with ASCII
- nowadays standard: UTF-8
- LaTeX-Support for UTF-8 steadily growing (UTF-8 Input, Labels, etc)

A small LaTeX document

Exercise 1

- open texmaker / texstudio / TeXworks
- type in the following document
- choose pdfLaTeX as tool and compile
- ☺

```
\documentclass{minimal}
\begin{document}
The most simple possible document.
\end{document}
```

LuaLaTeX

Exercise 2

Add LuaLaTeX as a compilation tool, get it working and compile the previous example with it.

Exercise 3

Add the package `lmodern` and use some special characters, for example: ¥ š ť é ä ą ď ľ ạ â à ā ã ȧ. Try this with lualatex and pdflatex. What's the difference?

LaTeX workflow

- source code is prepared in a specialized editor
- the program (pdflatex or lualatex) is actually pdftex or luatex with a lot of precompiled macros
- different auxiliary files are produced during the run
- several compilation runs may be necessary
- auxiliary programs may be used to sort indices, convert graphics etc.
- in the end, a pdf file is ready to be viewed

LaTeX workflow

- source code is prepared in a specialized editor
- the program (pdflatex or lualatex) is actually pdftex or luatex with a lot of precompiled macros
- different auxiliary files are produced during the run
- several compilation runs may be necessary
- auxiliary programs may be used to sort indices, convert graphics etc.
- in the end, a pdf file is ready to be viewed
- advantages:
  – separation of content, formatting information and output
  – improves stability and almost eliminates the possibility of data loss
  – final document cannot be changed
  – edition history is not (unintentionally) available
  – output is much smaller than e. g. a comparable word document

How to interact with LaTeX

- LaTeX knows “commands” and “environments”
- a command starts with a backslash `\`
- environments start with `\begin{env}` and end with `\end{env}`
- both commands and environments may have:
  – mandatory arguments, given in curly braces `{}`, or
  – optional arguments given in square brackets `[]`

How to interact with LaTeX

- LaTeX knows “commands” and “environments”
- a command starts with a backslash `\`
- environments start with `\begin{env}` and end with `\end{env}`
- both commands and environments may have:
  – mandatory arguments, given in curly braces `{}`, or
  – optional arguments given in square brackets `[]`
- in principle, a command or environment is an abbreviation for a lot of low-level instructions:
- e. g., `\section{}` means: “Typeset the given text in a sans serif font, bold face, 16 point, add the section number in front, increase the section counter, add an entry in the table of contents, add vertical space before and after the text, and suppress the indent of the next paragraph.”

LaTeX - commands

- a command starts with a backslash \
- a command may read arguments
- white-spaces collapse to one space or are ignored
- text right of % is treated as comment (use \% to get a percent sign)

```
\command a
\command   {a}
    additional    whitespaces like
  spaces,   tabs           or
       newlines   are
 ignored.
% and comments are totally ignored
```

results in

additional whitespaces like spaces, tabs or newlines are ignored.

Structure of a LaTeX source file

Every typical document contains the following elements:

- first: the class `\documentclass{}`

small example document:

```
\documentclass{scrartcl}
\usepackage{fontspec}
\author{A. U. Thor}
\title{My First Document}
\begin{document}
<document's content>
\end{document}
```

Structure of a LaTeX source file

Every typical document contains the following elements:

- first: the class `\documentclass{}`
- then: the header (`\usepackage{}`, `\author{}`, `\title{}`, ...)

small example document:

```
\documentclass{scrartcl}
\usepackage{fontspec}
\author{A. U. Thor}
\title{My First Document}
\begin{document}
<document's content>
\end{document}
```

Structure of a LaTeX source file

Every typical document contains the following elements:

- first: the class `\documentclass{}`
- then: the header (`\usepackage{}`, `\author{}`, `\title{}`, …)
- finally: the main part `\begin{document}` … `\end{document}`

small example document:

```
\documentclass{scrartcl}
\usepackage{fontspec}
\author{A. U. Thor}
\title{My First Document}
\begin{document}
<document's content>
\end{document}
```

Structure of a LaTeX source file

Every typical document contains the following elements:

- first: the class `\documentclass{}`
- then: the header (`\usepackage{}`, `\author{}`, `\title{}`, …)
- finally: the main part `\begin{document}` … `\end{document}`

small example document:

```
\documentclass{scrartcl}
\usepackage{fontspec}
\author{A. U. Thor}
\title{My First Document}
\begin{document}
<document's content>
\end{document}
```

Exercise 4

Type, compile and view this small example document. Add `\maketitle` and see what happens!

Some small examples

Exercise 5

To get a first touch with TeX: Tell the lecturer what you want to do and try out the solutions he/she suggests (or takes from the magical book, documentation, ...)

Auxiliary programs

- most auxiliary programs run automatically or can be accessed via the GUI
- but sometimes the command line can be very useful
- especially for finding documentation!

Auxiliary programs

- most auxiliary programs run automatically or can be accessed via the GUI
- but sometimes the command line can be very useful
- especially for finding documentation!
- for windows: `dir`, `cd`, `del`
- for linux: `ls`, `cd`, `rm`

Exercise 6

Open the command line, navigate to your document, and compile it using the command lualatex <dateiname>.tex

Auxiliary files

.tex	TeX-file with source text
	output formats
.dvi	‘normal’ TeX-output
.xdv	XeTeX-output
.pdf	pdfTeX-output or result of conversion of (x)dvi

Auxiliary files

.tex	TeX-file with source text	
	output formats	
.dvi	‘normal’ TeX-output	
.xdv	XeTeX-output	
.pdf	pdfTeX-output or result of conversion of (x)dvi	
	auxiliary files	
.log	Log-file with informations, warnings, errors	**(write only)**

Auxiliary files

.tex	TeX-file with source text	
	output formats	
.dvi	‘normal’ TeX-output	
.xdv	XeTeX-output	
.pdf	pdfTeX-output or result of conversion of (x)dvi	
	auxiliary files	
.log	Log-file with informations, warnings, errors	**(write only)**
.aux	auxiliary file with temporary informations	**(read and write)**
.toc	table of contents	
.lof	list of figures	
.synctex.gz	nessecary for SyncTeX-functions	
⋮	⋮	

Document classes

- document classes determine the look of a document (page layout, fonts, ...)
- LaTeXs standard classes: `book` , `report` , `article` , `letter`

Document classes

- document classes determine the look of a document (page layout, fonts, ...)
- LaTeXs standard classes: `book` , `report` , `article` , `letter` , `minimal`
- much more features and adapted for german style: classes of the KOMA script: `scrbook` , `scrreprt` , `scrartcl` , `scrlttr2`
- specialized classes for presentations (`beamer`), CVs, ...
- use *one* document class per document, with `\documentclass{scrartcl}` at the very beginning of the code
- `\documentclass[]{}` takes an optional argument, depending on the class' implementation

Packages

- packages are (small or large) collections of commands
- and loaded with `\usepackage{packagename}`
- most packages have options (`\usepackage[opt. list]{}`) to adjust the behaviour and/or commands
- packages might depend on others (this is taken care of automatically)
- packages may (very few cases) be incompatible with each other

⇒ you're lost ...

Packages

- packages are (small or large) collections of commands
- and loaded with `\usepackage{packagename}`
- most packages have options (`\usepackage[opt. list]{}`) to adjust the behaviour and/or commands
- packages might depend on others (this is taken care of automatically)
- packages may (very few cases) be incompatible with each other

⇒ ask an expert or consult documentation

Documentation

- every class and package has a documentation
- accesible via command line:
  `texdoc <packagename>`
- in the best case, a pdf file opens
- if nothing is found, try the `-l` or `-s` versions of `texdoc`

Documentation

- every class and package has a documentation
- accesible via command line:
  `texdoc <packagename>`
- in the best case, a pdf file opens
- if nothing is found, try the `-l` or `-s` versions of `texdoc`
- if that does not help, search the internet

Documentation

- every class and package has a documentation
- accesible via command line:
  `texdoc <packagename>`
- in the best case, a pdf file opens
- if nothing is found, try the `-l` or `-s` versions of `texdoc`
- if that does not help, search the internet
- if you search the net first, consult the documentation for more accurate/up-to-date information

Sectioning commands

Most document classes offer the following predefined sectioning commands:

- `\part{}`, `\chapter{}` (not in all classes)
- `\section{}`, `\subsection{}`, `\subsubsection{}`
- `\paragraph{}`, `\subparagraph{}`
- `\minisec{}` (only in the KOMA classes)

Sectioning commands

Most document classes offer the following predefined sectioning commands:

- `\part{}`, `\chapter{}` (not in all classes)
- `\section{}`, `\subsection{}`, `\subsubsection{}`
- `\paragraph{}`, `\subparagraph{}`
- `\minisec{}` (only in the KOMA classes)

The table of contents is typeset with `\tableofcontents` and is based on the above stated commands.

(Typically, three or four levels of sectioning are totally sufficient.)

Sectioning commands

Most document classes offer the following predefined sectioning commands:

- `\part{}`, `\chapter{}` (not in all classes)
- `\section{}`, `\subsection{}`, `\subsubsection{}`
- `\paragraph{}`, `\subparagraph{}`
- `\minisec{}` (only in the KOMA classes)

The table of contents is typeset with `\tableofcontents` and is based on the above stated commands.

(Typically, three or four levels of sectioning are totally sufficient.)

Exercise 7

Add some sectioning commands and a table of contents to your document.

Lists

the `itemize` environment can be used for a basic list:

```
\begin{itemize}
\item first item
\item[2\textsuperscript{nd}] second item
\end{itemize}
```

`enumeration` does the same with numbers:

```
\begin{enumerate}
\item first item
\item second item
\end{enumerate}
```

Exercise 8

Try out these two environments. What happens if one item is a list itself?

lists

for different numberings use the `enumitem` package:

```
% Roman numbers
\begin{enumerate}
[label=(\roman*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

```
% Arabic numbers
\begin{enumerate}
[label=\arabic*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

```
% Alphabetical
\begin{enumerate}
[label=\alph*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

lists

for different numberings use the `enumitem` package:

```
% Roman numbers
\begin{enumerate}
[label=(\roman*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

```
% Arabic numbers
\begin{enumerate}
[label=\arabic*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

```
% Alphabetical
\begin{enumerate}
[label=\alph*)]
   \item One
   \item Two
   \item Three
\end{enumerate}
```

(i) One
(ii) Two
(iii) Three

1) One
2) Two
3) Three

a) One
b) Two
c) Three

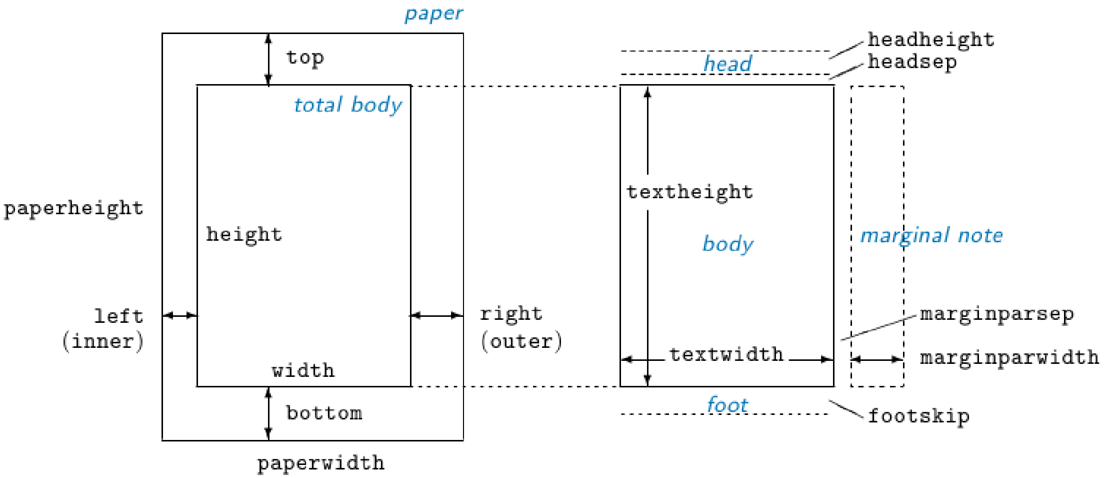
1. The Name of the Game
2. Introduction and Basic Usage of LaTeX
   - A complete LaTeX system
   - LaTeX in a nutshell
   - How to use LaTeX
   - Auxiliary programs, the command line
   - Classes, packages and documentation
   - Basic document structures
   - Lists and enumerations
3. Changing the look: page dimensions and fonts
   - Page dimensions and type area
   - Headers and footers with `scrlayer-scrpage`
   - So far ...
   - Font switching
   - Font packages
   - Loading system fonts

Page dimensions with geometry

- the `geometry` package offers various ways to adjust the type area
- type area and page dimensions can be changed in the document

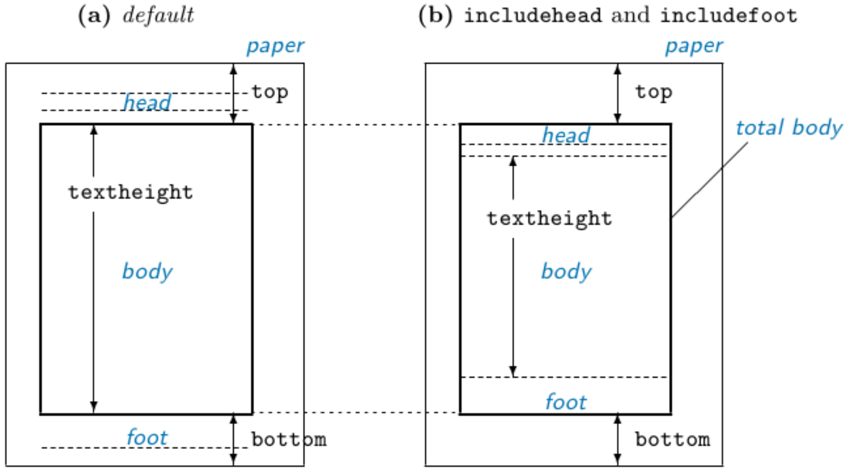
```
\usepackage[top=2cm,bottom=1cm]{geometry}
<OR>
\geometry{outer=3cm,inner=1.5cm}
```

Page dimensions with geometry



(Graphics taken from the `geometry` documentation.)

Page dimensions with geometry


(a) *default*

(b) includehead and includefoot

(Graphics taken from the geometry documentation.)

Page dimensions with geometry

Exercise 9

Change the page dimensions of your document according to the following specifications. Consult the `geometry` package for the names of the parameters.

- top: 2 cm
- bottom: 3 cm
- inner margin: 1 cm
- outer margin: 1.5 cm

If you are not satisfied with these parameters, change them and/or others at your will.

Page styles

- LaTeX defines a small set of standard page styles:
- `empty`, `plain`, `headings` (and `myheadings`)
- pagestyle can be changed via

  `\pagestyle{plain}` (for the whole document)
  `\thispagestyle{plain}` (for this page)

- the first page of a chapter is often set in style `plain` (depending on the document class)

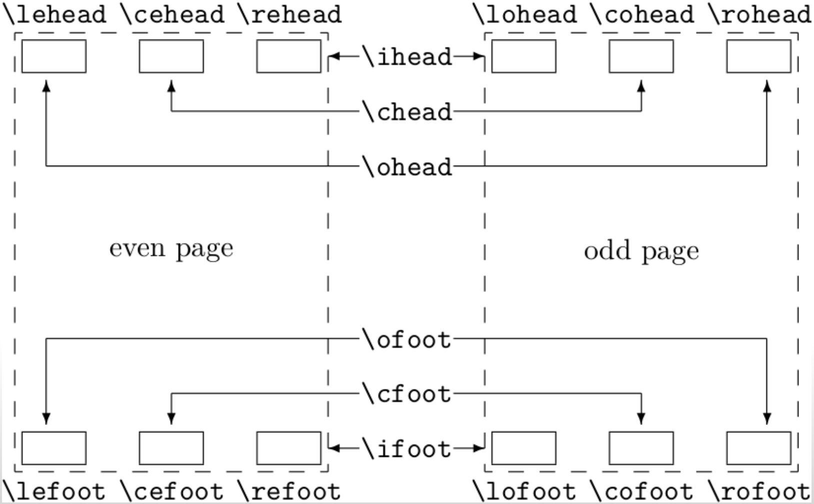
Exercise 10

Produce a document with several pages (use `\newpage` if necessary) and chapters. (Remember the document classes you need then.) Test the page styles given above – what effects do they have?

scrlayer-scrpage

- there are different packages to adapt the headers and footers
- scrlayer-scrpage is one of the most versatile and included in KOMA
- additional pagestyles scrheadings, scrplain
- adjustment of elements via `\lehead[plain conf.]{headings conf.}`
- note: the first page of a chapter has pagestyle scrplain
  (see optional argument ‘plain conf.’)
- for more: see scrguide documentation

scrlayer-scrpage



(picture taken from scrguide documentation)

scrlayer-scrpage

There special commands containing details on the running page:

`\pagemark`	number of the current page
`\leftmark`	current chapter name printed like “CHAPTER 3. THIS IS THE CHAPTER TITLE”
`\rightmark`	current section name printed like “1.6. THIS IS THE SECTION TITLE”
`\chaptername`	the name chapter in the current language. If this is English, it will display “Chapter”
`\thechapter`	current chapter number
`\thesection`	current section number

for example:

```
\chead[\pagemark]{\thechapter - \chaptername}
```

scrlayer-scrpage

Example

```
\documentclass{scrreprt}
\usepackage{scrlayer-scrpage}
\pagestyle{scrheadings}
\clearscrheadfoot % clears all scrheadings - definitions
\chead[\pagemark]{\thechapter{} - \chaptername}
\ohead{\pagemark}
\lohead[LOHEAD]{lohead}
\lehead[LEHEAD]{lehead}
\begin{document}
Content...
```

So far ...

- basic structure
- commands, comments
- classes (book, report, article, letter) and the KOMA counterparts (scrbook, scrreprt, scrartcl, scrlttr2)
- structuring commands `\chapter`, `\section`, ...
- table of contents (`\tableofcontents`)
- environments (itemize, enumerate)
- page dimensions (geometry)
- headers, footers (scrlayer-scrpage)

Basic font commands

command	switch	
`\textit{}`	`\itshape`	*italics*
`\textup{}`	`\upshape`	upright
`\textsc{}`	`\scshape`	SMALL CAPS
`\textbf{}`	`\bfseries`	**bold face**
`\textmd{}`	`\mdseries`	medium
`\textsf{}`	`\sffamily`	sans serif
`\texttt{}`	`\ttfamily`	`teletyper`
`\textrm{}`	`\rmfamily`	roman

Combine one shape, one series and one family (mostly) at will.

Font sizes

most document classes define all of the following font sizes:

command	
`\tiny`	example text
`\scriptsize`	example text
`\footnotesize`	example text
`\small`	example text
`\normalsize`	example text
`\large`	example text
`\Large`	example text
`\LARGE`	example text
`\huge`	example text
`\Huge`	example text

If you *really* need something else:

```
{\fontsize{20}{24}\selectfont text …}
```

Font packages

- special packages make fonts useable with TEX
- needed as the old TEX-variants cannot deal directly with system fonts
- useful packages e. g. `lmodern` , `libertine` , ...

`lmodern` - Latin Modern family of fonts
`libertine` - OpenType-Fonts

- font catalog: link

Best typographic results with `microtype`

- font expansion
- character protrusion
- (you might need tlcontrib for latest `microtype` -version)

Exercise 11

Load `microtype` and test it with TEX fonts and system fonts. Guess what the terms “font expansion” and “character protrusion” mean.
Change the parameters for font expansion and character protrusion, consult the documentation to find the corresponding options.

Best typographic results with `microtype`

page 4 of the microtype documentation demonstrates

- font expansion
- character protrusion

```
\usepackage{microtype}
\begin{document}
\microtypesetup{protrusion=true}
\microtypesetup{expansion=true}
content ...
\end{document}
```

Do not use inputenc

- many online references use `inputenc` for using umlaute
- `inputenc` was useful for pdfLaTeX ...
- dont use it!

Compare the results by cut-and-paste from the pdf-output!

```
\usepackage{fontspec}
\begin{document}
Täglich Öffnung Fuß Sauerstoffflasche
\end{document}
```

```
\usepackage[latin1]{inputenc}
\begin{document}
Täglich Öffnung Fuß Sauerstoffflasche
\end{document}
```

maybe utf8 or ansinew instead of latin1

Footnotes

Footnotes are easily set with `\footnote{..footnote text...}`.
In some cases, it may be necessary to split the footnote mark from the footnote text:
`\footnotemark` and `\footnotetext{...}`

4 Cross references

5 Typesetting of Mathematics

- Basic maths
- Some examples
- Typesetting numbers and units
- The unicode-math package
- plotting data and functions

Cross references

- labels are easily defined with `\label{keyword}`
- ...and easily referred to with `\ref{keyword}`
- refer to the page of the object by `\pageref{keyword}`
- The object to be referred is automatically set by the context.
- all keywords share the same namespace! It is commonly used to add object-specific abbreviations.

```
In chapter \ref{ch:first} we will ..
\chapter{first}
\label{ch:first}
```

Exercise 12

Add labels to your document and refer to these. Have a look into the aux-file!

Cross references

```
In chapter \ref{ch:first} we will ..
\chapter{first}
\label{ch:first}
\section{The name of the game}
\label{sec:name}
Let us talk about the name of the game.
\subsection{alias names}
\label{ssec:alias}
```

4 Cross references

5 Typesetting of Mathematics

- Basic maths
- Some examples
- Typesetting numbers and units
- The unicode-math package
- plotting data and functions

The essentials of math

There are two modes of math: inline-math $a^2 + b^2 = c^2$, and display-math:

$$a^2 + b^2 = c^2$$

Inline-math is activated with a single $ and ended with a single $:

```
Pythagoras found that $a^2 + b^2 = c^2$.
```

Display-math starts with `\[` end ends with `\]`:

```
Pythagoras found that \[a^2 + b^2 = c^2.\]
```

Exercise 13

Typeset some simple formulae. How do letters and numbers look compared to the textmode? What about the spacing around operators and relations (+- or = ≥ ≠ etc.)?

Numbered equations

- normally, equations in display mode are enumerated
- use `\begin{equation} ... \end{equation}` for this
- you might want to define a shorthand:

```
\def\[{\begin{equation}}
\def\]{\end{equation}}
```

(there are better ways to define macros, but this is the shortest one here ...)

Basic math input

- sub- and superscript: `a_b a^c a_b^c a_{b^c}` (grouping is important!)
- special commands for standard functions: `\sin \cos \tan` etc.
- integrals and sums: `\int \sum` $\int \sum \Sigma$
- Greek letters: `\alpha \beta \gamma \varepsilon`
- roots: `\sqrt[]{}`
- fractions: `\frac{}{}`
- a huge number of mathematical and other symbols

Basic math input

- sub- and superscript: `a_b a^c a_b^c a_{b^c}` (grouping is important!)
- special commands for standard functions: `\sin \cos \tan` etc.
- integrals and sums: `\int \sum` $\int \sum \Sigma$
- Greek letters: `\alpha \beta \gamma \varepsilon`
- roots: `\sqrt[]{}`
- fractions: `\frac{}{}`
- a huge number of mathematical and other symbols
- detexify is very helpful in searching LaTeX commands: `http://detexify.kirelabs.org/classify.html`

Exercise 14

Typeset some more math, use sub- and superscripts, also try them as limits for operators like $\int$ and Σ. Look for some special symbols using detexify and use them!

Arrays of equations

- load the package `amsmath`
- try out the environments `eqnarray` and `align` as shown below

```
\begin{eqnarray}
a &=& b \\
c &=& d
\end{eqnarray}
\begin{align}
a &= b \\
c &= d
\end{align}
```

Arrays of equations

$\Longrightarrow$ Never use eqnarray!

How to do what?

- very many packages are available for different needs
- every document should load the `amsmath` package

- searching the internet is useful, but may result in (very) old results (often `eqnarray` is suggested!)

⟹ for everything else, consult the document mathmode by Herbert Voss (it used to be available by texdoc mathmode, better try online and search for 'mathmode voss')

Exercise 15

Open the mathmode-document and jump to an arbitrary page. Try out some of the given code or search for something special that you're interested in.

Some math-mode examples

- small spaces `\,` `\:` `\;` (increasing from left to right)
- more space `\quad` `\qquad`
- negative space: `\!`
- place text inside formulars with `\text{}` (from `amsmath` package)

```
\newcommand*{\dif}{\mathop{}\!\mathrm d}
\begin{equation}
 \int_0^\infty f(x)\dif x
 \quad\text{or for example}\quad
 \int\limits_0^\infty f(x)\dif x
\end{equation}
```

$$\int_0^\infty f(x)\,\mathrm{d}x \quad\text{or for example}\quad \int\limits_0^\infty f(x)\,\mathrm{d}x \tag{1}$$

Some math-mode examples

- small spaces `\,` `\:` `\;`   (increasing from left to right)
- negative space: `\!`

```
\newcommand*{\dif}{\mathop{}\!\mathrm d}
\newcommand*{\D}{\,\mathrm d}
\begin{equation}
  \frac{\dif f}{\dif x}
 \quad\text{ or }\quad
  \frac{\D f}{\D x}
\end{equation}
```

$$\frac{\mathrm{d}f}{\mathrm{d}x} \quad\text{ or }\quad \frac{\,\mathrm{d}f}{\,\mathrm{d}x} \tag{2}$$

Some math-mode examples

- brackets are sized correctly with `\left` and `\right`

```
\begin{equation}
\left[
    \left( \frac{a}{\sin(2\pi f)} \left( a+b \right) \right)
   -1 \right]
 = \left[
     1 + \frac{\left( \frac{1}{2} + b \right)}{\cos(\omega t)}
   \right]
\end{equation}
```

$$\left[\left(\frac{a}{\sin(2\pi f)}\left(a+b\right)\right)-1\right]=\left[1+\frac{\left(\frac{1}{2}+b\right)}{\cos(\omega t)}\right] \tag{3}$$

Numbers and units

- to distinguish between a variable and a unit:
- variables are typeset in *italics*, units upright
- $v = 5\cdot 5.32\,\mathrm{m/s}$ `$v=5\cdot 5.32\,\mathrm{m/s}$`
- $\neq 5\cdot 5.32 m/s$ `$5\cdot 5.32 m/s$`
- $\neq 5\cdot$ 5.32m/s `$5\cdot$ 5.32m/s`

Numbers and units

- to distinguish between a variable and a unit:
- variables are typeset in *italics*, units upright
- $v = 5 \cdot 5.32\,\mathrm{m/s}$ `$v=5\cdot 5.32\,\mathrm{m/s}$`
- $\neq 5 \cdot 5.32 m/s$ `$5\cdot 5.32 m/s$`
- $\neq 5 \cdot$ 5.32m/s `$5\cdot$ 5.32m/s`
- by hand: `$v = 5\cdot 5.32\,\mathrm{m/s}$`
- using `siunitx` : `$v = 5\cdot \SI{5.32}{m/s}$`
- package option `[math-rm]` might be necessary for the correct font!

Exercise 16

Typeset a small math example with numbers and units. Make use of the `\SI` macro, also consult the `siunitx` documentation about the macros `\num` and `\si` . Test these macros both in math mode and in text mode.

Numbers and units

siunitx

- using `siunitx` : `$v = 5\cdot \SI{5.32}{m/s}$`
- package option `[math-rm]` might be necessary for the correct font!
- `\per` with several options: `[per-mode = reciprocal]`
- `\SI{5.35}{\metre \per \second\squared}`

Numbers and units

english / german

english

$b = 3.14156 \times 10^{-4}$

```
$b = 3.14156 \times 10^{-4}$
```

german

$b = 3{,}14156 \cdot 10^{-4}$

```
$b = 3{,}14156 \cdot 10^{-4}$
```

In the german case: take care of the comma!

Numbers and units

german and `siunitx`

```
\usepackage{siunitx}
```

`$\SI{3.14156d-4}{\metre\per\second}$` → $3.14156 \times 10^{-4}\,\mathrm{m\,s^{-1}}$

```
\usepackage[mode=text,per-mode=symbol,
exponent-product = \cdot]{siunitx}
\sisetup{locale=DE}
```

`$\SI{3.14156d-4}{\metre\per\second}$` → $3{,}14156 \cdot 10^{-4}\,\mathrm{m/s}$

See also `https://de.wikipedia.org/wiki/Wissenschaftliche_Notation`

Unicode math

- TEX's traditional font mechanism allows for only 256 characters in a font
- math fonts need much more characters, so several font files are needed
- unicode defines also math characters, and OpenType math fonts contain all data in *one file*!

⟹ no additional packages needed for all common symbols

Unicode math

- TeX's traditional font mechanism allows for only 256 characters in a font
- math fonts need much more characters, so several font files are needed
- unicode defines also math characters, and OpenType math fonts contain all data in *one file*!

⟹ no additional packages needed for all common symbols

- however, some special symbols might not fit the font style …
- allows to use direct input of e. g. ℵ√Λℂ∫∀Σ⟺⟹∩ℵΠ
  (or still traditional input)
- all (and more) math alphabets are directly accessible
  (e. g. blackboard math etc.)

Plotting data and functions

- several packages allow plotting
- `gnuplottex` uses the external open source program `gnuplot` for calculations and plots
  $\Longrightarrow$ good output, but not independent of the system
  (requires gnuplot-installation)

Plotting data and functions

- several packages allow plotting
- `gnuplottex` uses the external open source program `gnuplot` for calculations and plots
  ⇒ good output, but not independent of the system
  (requires gnuplot-installation)
- `pgfplots` is based on the very useful T*i*kZ package
- uses TEX interna for calculations
  ⇒ very good output, but can be extremely slow ...

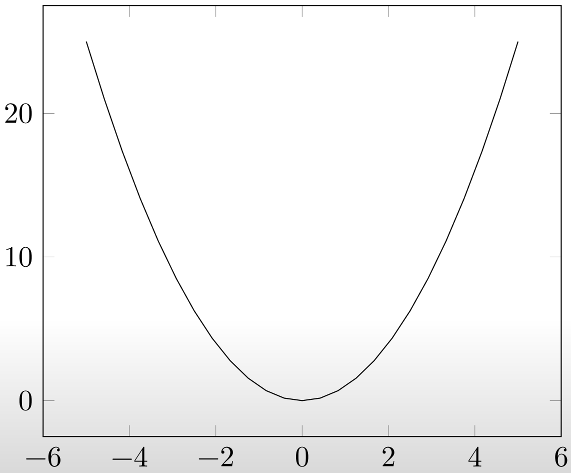
Plotting with pgfplots

Plotting a function

```
\begin{tikzpicture} %% as it is based on TikZ
\begin{axis}
\addplot [mark=no]{x^2};
\end{axis}
\end{tikzpicture} %% remember to close!
```

Plotting with `pgfplots`

Plotting a function



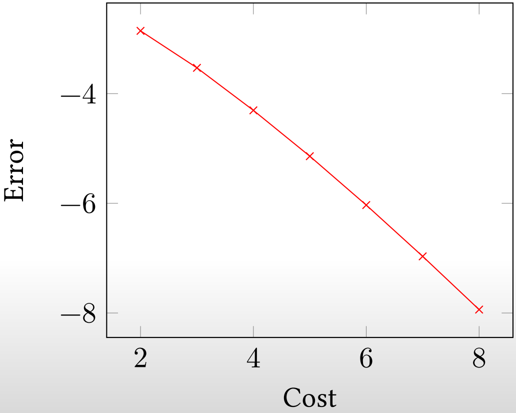
Plotting with pgfplots

plotting data

```
\begin{tikzpicture} %% as it is based on TikZ
\begin{axis}[       %% set up axis
xlabel=Cost,        %% labels etc.
ylabel=Error]
\addplot[color=red,mark=x] coordinates {
(2,-2.8559703)
(3,-3.5301677)
(4,-4.3050655)
(5,-5.1413136)
(6,-6.0322865)
(7,-6.9675052)
(8,-7.9377747)
};
\end{axis}
\end{tikzpicture} %% remember to close!
```

Plotting with pgfplots

Plotting data



Exercise 17

$\Longrightarrow$ consult the documentation on how to plot external data files, adapt the style etc.

Plotting with pgfplots

create external PDFs

- for complex plots, which take a long time to create, it is useful to use the `external` library.
- you have to extend the call of lualatex with '-shell-escape'
- every tikzpicture will be created as an external PDF-file and included on further LaTeX-runs.

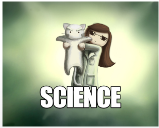
```
\usepackage{pgfplots}
\usepgfplotslibrary{external}
\tikzexternalize % activate externalization
```

6. Figures and tables
   - How to include a figure
   - Tabulars
   - Float environments
7. Writing a thesis
   - Preparation and organisation
   - Bibliographies
8. Typographic details
9. Presentations
10. Advanced topics: Lua code, defining macros, ...

Including an external figure

- the `graphicx` package is needed to include figures
- use `\includegraphics{filename}` with or without extension
- possible filetypes (for pdfLaTeX and LuaLaTeX) are `.pdf`, `.png`, `.jpg` (and only with the latest TeX Live versions: `.eps`)
- adjustment of size by `[width = ..., height = ...]`

```
\includegraphics[height=.2\textheight]{exo} %exo.png
\includegraphics[height=.2\textheight]{info} %info.eps
```



Exercise 18

Put some test graphic inside your document folder and include it into your document.

A simple tabular

- the most simple tabular-environment is the tabular-environment

```
\begin{tabular}{lcrp{2.5cm}}
left & centered & right & paragraph-like column, 2.5cm wide\\
le & ce & ri & short content
\end{tabular}
```

left	centered	right	paragraph-like column, 2.5cm wide
le	ce	ri	short content

More flexibility with `array`

- additional column types `m` and `b`
- with `>{}` and `<{}` one can insert content before/after every cell of a column:

```
\begin{tabular}{>{start}c<{stop}>{$}l<{$}}
first & a = b^2 \\
second &  x \mathrm d x = ½x²
\end{tabular}
```

start first stop	$a = b^2$
start second stop	$\int x \mathrm{d}x = \frac{1}{2}x^2$

Guidelines for nice tabulars

> “
>
> 1 Never, ever use vertical rules.
>
> 2 Never use double rules.
>
> 3 Put the units in the column heading (not in the body of the table).
>
> 4 Always precede a decimal point by a digit; thus 0.1 not just .1.
>
> 5 Do not use ‘ditto’ signs or any other such convention to repeat a previous value. In many circumstances a blank will serve just as well. If it won’t, then repeat the value.
>
> ”

(taken from the `booktabs` documentation by Simon Fear)

$\Longrightarrow$ *use horizontal lines of different thickness*

(Not so) nice tabulars

```
\begin{tabular}{|l|l|l|}
\hline\hline
first & second & third \\
\hline
content & more & content\\
\hline
content & more & content\\
\hline\hline
\end{tabular}
```

first	second	third
content	more	content
content	more	content

Nice tabulars with the booktabs package

```
\usepackage{booktabs}

\begin{tabular}{lll}
\toprule
first & second & third \\
\midrule
content & more & content\\
content & more & content\\
\bottomrule
\end{tabular}
```

first	second	third
content	more	content
content	more	content

Variable columns with `tabu`

- there is a large number of packages and environments for specialized tabulars
- `tabularx` , `tabulary` , `tabu` , `widetable` , ...
- the `tabu` package is one of the newest, most flexible packages and easy to use
- environments `tabu` replaces `tabular`
- given the same header, it behaves the same
- additional column type `X` spreads as much as necessary
- optional argument of the `X`-*column* determines the relative width, positioning and justification
- `l`,`r`,`c`,`L`,`R`,`C`: left, right, centered (uppercase allows hyphenation)
- `j`,`J`: justified (default is `j`)

A tabu tabular

```
\begin{tabu}{|X[1.5,l]|X[0.8,L]|X[J]}
some example e
& someexample text to see the effect
& some example texttoseethe effect \\
\end{tabu}
```

Exercise 19

Try out the `tabu` environment. Use different column styles and options for the `X` column.

Floating environments

- everything that has not a fixed place in the text is called a “float” or “floating environment”
- most floats are tables or figures
- environments with these names are meant for tabulars or pictures
- consist of the content, a caption, label, and positioning commands

A table

```
\begin{table}[h]
\begin{tabular}{ll}
\toprule
a & tabular \\
with & three \\
lines \\
\bottomrule
\end{tabular}
\caption{The caption has to stand \emph{before} the label.}
\label{tab:threelinetabular}
\end{table}
```

A figure

```
\begin{figure}[h]
\includegraphics[width=20mm]{warn}
\caption{The caption has to stand \emph{before} the label.}
\label{fig:image1}
\end{figure}
```

Figure: The caption has to stand *before* the label.

With KOMA classes

```
\setcapindent{0em}
\setkomafont{captionlabel}{\sffamily\bfseries}
\renewcaptionname{ngerman}{\figurename}{Abb.}
```

A figure with sidecaption with KOMA classes

```
\setcapindent{0em}
\setkomafont{captionlabel}{\sffamily\bfseries}
\begin{figure}
\begin{captionbeside}[tof-entry]{description}%
[orientation][width][offset]
...
\end{captionbeside}
\end{figure}
```

Figure 1: Example of captionbeside, with inside caption. Possible orientations are i-inner o-outer l-left r-right.

Side-caption with sidecap

```
\begin{SCfigure}[2][h]
        \includegraphics[width=30mm]{unilogo}
        \caption{The caption has to stand \emph{before} the label.}
        \label{fig:image1a}
\end{SCfigure}
```

Abb. 2.5: The caption has to stand *before* the label.

Floats

- the optional argument `\begin{table}[...]` gives a hint for the positioning
- possible values: `tbhp`, with `float` package also `H`
  (**t**op; **b**ottom; **h**ere, if possible; **p**age with floats; **H**ere, definitely)
- `\caption` types the caption (mand. arg.) and an entry in the lof (opt. arg.)
- caption position on the side by package `sidecap`

Exercise 20

Insert some `table` and/or `figure` environments and see how positioning works

6 Figures and tables
- How to include a figure
- Tabulars
- Float environments

7 Writing a thesis
- Preparation and organisation
- Bibliographies

8 Typographic details

9 Presentations

10 Advanced topics: Lua code, defining macros, ...

Organisation

- to avoid huge source files: separate your document corresponding to the content (per chapter or so)
- put *everything* you need (text files, pictures, additional sources, ...) into one folder and subfolders
- to separate content from formatting: prepare a separate header file that contains all “TEXnical stuff”
- define one main document and use `\input` to load the header file
- use `\include` to include the content files (*inside the main body*)
- use `\includeonly`/`\excludeonly` to include only certain files (*in the header*)
- for chapters that are *not* included, still the `.aux` file is loaded so that references, page numbers etc. are still correct! (only with `\include`/`\includeonly`, not with `\input`)

The main file

```
\input{etc/header} % in a subfolder called etc
\usepackage{excludeonly}
\excludeonly{
%0-Abstract,
%1-Introduction,
2-Experimental Setup,
% ...
}
\begin{document}
    \include{0-Abstract}
    \include{1-Introduction}
    \include{2-Experimental Setup}
    ...
\end{document}
```

The main file

```
\input{etc/header} % in a subfolder called etc
\includeonly{chapter1}

\begin{document}
    \tableofcontents
    \include{introduction}
    \include{chapter1}
    \include{chapter2}
    \include{chapter3}
\end{document}
```

Defining the main document

- it is annoying to change to the main document for compilation
- most editors allow to define a main or “master” document
- this is automatically compiled instead of the one you’re editing
- in some TEXeditors you might have to add

  ```
  % !TeX root = maindoc.tex
  ```

  at the beginning of each file (good practice to add it anyway)

The header file

- contains everything that should be in the header
- to keep the main document as simple as possible
- do *not* use a predecessor's header file as you might not understand it

⟹ can lead to last-minute-problems in the formatting/compilation etc.

- write your own file with your own adaptions, formatting etc.

⟹ Takes more time, but is safe.

The header file

- contains everything that should be in the header
- to keep the main document as simple as possible
- do *not* use a predecessor's header file as you might not understand it

⇒ can lead to last-minute-problems in the formatting/compilation etc.

- write your own file with your own adaptions, formatting etc.

⇒ Takes more time, but is safe.

- you might use an additional file for definitions, new commands etc.

typical document class definition:

```
\documentclass[a4,ngerman,twoside,
  BCOR=2cm,fleqn,openright,tocdepth=0
]{scrreprt}
```

Abstract

- english and ngerman

```
\documentclass[twoside]{scrreprt}
\usepackage[english,ngerman]{babel}
\begin{document}
\let\raggedsection\centering
\section*{\abstractname}
Dies ist die Zusammenfassung auf Deutsch.

\selectlanguage{english}
\section*{\abstractname}
This is an english abstract.
 \let\raggedsection\raggedright
\selectlanguage{ngerman}% or which ever you main language is
\chapter{Einleitung}
\end{document}
```

Shortened list of useful packages

```
\usepackage{
  amsmath,amssymb,babel,braket,booktabs,csquotes,float,
  fontspec,geometry,graphicx,hyperref,luacode,marginnote,
  microtype,nicefrac,paralist,pgfplots,siunitx,subfig,
  subfloat,tabu,unicode-math,xspace,auto-pst-pdf,cleveref
}
\usepackage{scrlayer-scrpage}
\automark[section]{chapter}
\cfoot[]{}
\ofoot[\pagemark]{\pagemark}
\pagestyle{scrheadings}

\usepackage[backend=biber,sorting=none]{biblatex}
\bibliography{etc/bib-final}
```

Using hyperlinks

- with the `hyperref` package, all links become clickable hyperlinks
- look of the links adjustable with `\hypersetup{}`

my headers most often contain:

```
\hypersetup{
  colorlinks = true,
  linkcolor = black,
  urlcolor = blue,
  citecolor = red
}
```

Lists

- `\tableofcontents` (after the front matters)
- `\listoftables` (mostly at end of document)
- `\listoffigures` (mostly at end of document)
- remember the short version of `\caption[...]{...}`

Here is an example python script for you to start a dummy thesis: link

BibTEX and biber

- BibTEX was used for many years to sort the bibliography and produce a nice output
- several successors, latest one being biber
- for LATEX, the package `biblatex` allows very flexible and easy to use bibliographies

BibTEX and biber

- BibTEX was used for many years to sort the bibliography and produce a nice output
- several successors, latest one being biber
- for LATEX, the package `biblatex` allows very flexible and easy to use bibliographies
- in the document, use `\cite{TTT_kar}` to get the cite key
- in an external bibliography file, an according entry has to be present
- in the header, the file needs to be specified (using `biblatex`: `\addbibresource {etc/bib-final}`
- finally, `\printbibliography` typesets the bibliography

biblatex

- use \autocite in favor of \cite to switch between different styles
- bibencoding=utf8 for utf8-encoded bibliography file
- autocite=superscript for different styles (superscript,plain,inline,footnote)
- to have the bibliography in the toc: bibliography=totocnumbered bibliography=totoc

```
\documentclass[german,bibliography=totocnumbered]{scrbook}
\usepackage[backend=biber,autocite=superscript,
bibencoding=utf8]{biblatex}
...
\autocite{TTT_kar}
...
```

The bibliography file

- needs the file extension `.bib`
- contains entries like

```
@article{ TTT_kar,
author = "Tamasi Kar and Danilo Lima and Andre Sch{\"o}ning and Jike Wang"
title = "{{``Triplet Track Trigger for Future High Rate Experiments''}}",
journal = "in Proceedings of Connecting the Dots and Workshop on Intellige
year = "2019",
note = {arXiv:\href{https://arxiv.org/pdf/1910.06946.pdf}{1910.06946[physi
det]}}
}
```

- Wiki link to different possible entry fields (with optional and compulsory arguments): Bibliography Management

biber-example

```
\usepackage[backend=biber]{biblatex}
\bibliography{exbibliography}
\begin{document}
Kar \cite{kar2020}
\printbibliography
\end{document}

%file exbibliography.bib
@misc{kar2020,
  title={Introduction to \LaTeX},
  author={Kar, Tamasi},
  ALTauthor={Kar, Tamasi},
  date={2022-04-14}
}
```

biber-example2

```
\usepackage[backend=biber,bibencoding=utf8]{biblatex}
%\usepackage[style=verbose,autocite=footnote,language=german,bibencoding=utf8
\bibliography{exbibliography}
\begin{document}
Kar \cite{kar2020}
\printbibliography
\end{document}

%file exbibliography.bib
@misc{kar2020,
  title={Introduction to \LaTeX},
  author={Kar, Tamasi},
  ALTauthor={Kar, Tamasi},
  date={2022-04-14}
}
```

The bibliography file

- several tools help in building the bib file
- for short/simple bibliographies: build it yourself!
- ask e. g. `http://scholar.google.de/` for a paper title
- in the google preferences, you can add the import of BibTEX citations!

6. Figures and tables
   - How to include a figure
   - Tabulars
   - Float environments

7. Writing a thesis
   - Preparation and organisation
   - Bibliographies

8. Typographic details

9. Presentations

10. Advanced topics: Lua code, defining macros, ...

Typography in a nutshell

- use nice fonts (*not* Comic Sans ...)
- Hyphen, dash, minus sign: `- -- $-$` - – —
- blanks: `\ \, ~ \quad \qquad` e. g. i. e. `i.\,e. or e.\,g.`
- quotation marks: „"; use `csquotes` for best results and context-sensitive quotation marks (or ``` `` text '' ```)
- avoid single lines at the beginning/end of a page (use \\, maybe `\break` or `\newpage` or `\vspace`)
- in math mode: delimiters have to be larger than everything between them – use `\left` and `\right` (or e. g. package `braket` for quantum mechanics)

$$(\frac{1}{2}) \text{ vs. } \left(\frac{1}{2}\right), \quad \langle 0 \,|\, W \,|\, 1 \rangle$$

6. Figures and tables
   - How to include a figure
   - Tabulars
   - Float environments
7. Writing a thesis
   - Preparation and organisation
   - Bibliographies
8. Typographic details
9. Presentations
10. Advanced topics: Lua code, defining macros, ...

The beamer class

- presentations are a special kind of document
- the resulting pdf is thought for full-screen view
- (un)covering effects are simulated using several pdf-pages
- the beamer class offers a large variety of layouts

```
\documentclass{beamer}
\begin{document}
\ begin{frame}{frame title}{sub title}
slide content
\ end{frame}
\end{document}
```

Frames

- frames can have a title and subtitle (but both is optional!)
- for dynamic effects, use `\only <1>{}` or `\uncover <2-5>{}`
- `\item` also takes the `<1-8>` argument

Frames

- frames can have a title and subtitle (but both is optional!)
- for dynamic effects, use `\only <1>{}` or `\uncover <2-5>{}`
- `\item` also takes the `<1-8>` argument
- adjust the look of the frames using themes, inner themes, outer themes, color themes etc.:

```
  \usetheme{Frankfurt}
  \useoutertheme{infolines}
  \usecolortheme[RGB={0,100,130}]{structure}
  \useinnertheme{rounded}
  \setbeamertemplate{navigation symbols}{}
% last line turns off the navigation symbols!
```

6 Figures and tables
- How to include a figure
- Tabulars
- Float environments

7 Writing a thesis
- Preparation and organisation
- Bibliographies

8 Typographic details

9 Presentations

10 Advanced topics: Lua code, defining macros, ...

Lua code

- using LuaTeX, one has access to the programming language Lua directly in the document
- with the package `luacode` , the environment `luacode` is most convenient
- can be used for simple calculations up to complex manipulations of TeXs interna!
- for more information about Lua: `http://www.lua.org/`

```
\begin{luacode}
tex.print("hallo")
tex.print(5+5)
\end{luacode}
```

Defining macros with `\newcommand`

- the `\newcommand` command defines new commands
- if one optional argument is given, it denotes the number of arguments the command takes
- arguments can be used inside the definition via `#1`, up to 9 arguments possible

```
\newcommand\mycommand[2]{The first #1 argument%
                            and the second: #2}
```

Defining macros

```
\usepackage{luacode}
\newcommand\mysqrt[1]{\luadirect{tex.print(math.sqrt(#1))}}
...
the square root of 2 is approximately \mysqrt{2}
```

results in:

the square root of 2 is approximately 1.4142135623731

```
% the TeX way to define a command
\def\mylua#1{\luadirect{tex.print(math.sqrt(#1))}}
\def\rmylua#1{%
\luadirect{%
tex.print(math.floor(math.sqrt(#1)*1000+0.5)/1000)%
}}
```

Defining macros 2

```
\begin{tabular}{SSS}
\toprule
{value $x$} & {square root ($\sqrt x$)} & {rounded} \\
\midrule
999 & \mylua{999} & \rmylua{999} \\
200 & \mylua{200} & \rmylua{200} \\
2   & \mylua{2}   & \rmylua{2} \\
0.2 & \mylua{0.2} & \rmylua{0.2} \\
\bottomrule
\end{tabular}
```

Defining macros 2

value x	square root ($\sqrt{x}$)	rounded
999	31.606 961 258 558	31.607
200	14.142 135 623 731	14.142
2	1.414 213 562 373 1	1.414
0.2	0.447 213 595 499 96	0.447

Defining macros using `xparse`

- the `xparse` -package makes heavy use of the LaTeX3 structures
- extremely flexible ways to implement many kinds of macros and use different argument types
- arbitrary argument delimiters possible
- arguments are given symbolically, similar to the `tabular` heading:

```
\NewDocumentCommand\mycommand{mom}{
 #1, #2, #3
}
```

⟹ the second argument is optional!

⟹ consult the documentation – the package is being actively developed and improved!

Happy
TeXing!