Introduction to Writing with LaTeX

The Basics

Aslak Johansen asjo@mmmi.sdu.dk

September 4, 2025

Part 1
Motivation

Observations

WYSIWYG editors provide plenty of distractions that compete for your attention.

- That attention is key to good writing.
- These distractions slows down the writing process.

WYSIWYG editors provide little help managing references of any kind.

WYSIWYG editors generally encourage bitmapped graphics.

WYSIWYG editors tends to result in little consistency in the produced documents.

The formats employed by WISIWYG editors are usually horrible in terms of version control.

WISIWYG editors are usually horrible for collaborative editing.

Why LaTeX? [1/2]

LaTeX is a programming language saved in clear text files using markup code.

- This makes it suitable for version control.
- Version control is a form of collaborative editing.
- Web-based editors (like overleaf) exists and work fairly well.

LaTeX (mostly) decouples structure and presentation.

- This results in highly consistent layouts.
- When writing, you can focus on the structure.
- If you wish to, you can *choose* to dig into the nitty gritty details.
- But there is rarely a need: Most templates make the result look great without intervention.

Why LaTeX? [2/2]

LaTeX is a highly extendable and mature system with lots of tools designed for technical documents.

- Working with a bibliography is trivial.
- Working with an index is trivial.
- Working with floats (like figures and tables) is trivial.
- Working with equations is simple.
- References to any of these *just work*.

LaTeX is a professional typesetting system and thus vector based.

- It is trivial to include vector graphics.
- But it is also trivial to include bitmapped graphics.

Note: This essentially mitigates the observations.

What is LaTeX?

LaTeX is a programming language that, when evaluated, produces a PDF file.

It relies heavily on the concept of macro expansion.

LaTeX is extended through packages. Currently there are 6000^{+} such packages on CTAN:

https://ctan.org

What is LaTeX? ▷ Styles

In LaTeX, contents and presentation are (mostly) separate.

During your day-to-day writing you care about structure: Sections, paragraphs, emphasis, listing ...

You don't care about how these are effectualized: Margins, fonts, colors, spacing, indents ...

These things are (generally speaking) covered by a *style*, and it typically does a very good job.

Should you be unhappy with the styling rules defined in your style, then you can:

1. Pick another.
2. Adjust it.
3. Hardcode specifics.

When to Not LaTeX?

- ***Note Taking*** I find that *markdown* fulfills this role adequately, and is faster to write.
- ***Audience Expectations*** Often, your audience will expect a Word file, and then it just doesn't make sense.
- ***Collaborator Skill Set*** Often, your collaborators won't be familiar with LaTeX, and then it simply isn't on the path of least resistance.
- ***Web*** It it has to go on the web, something that was originally meant for HTML production is likely the way to go.

Part 2
Building the PDF

Engines

In order to compile a LaTeX document into PDF you need a LaTeX engine.

Options:

- ***pdflatex*** The old engine that is fast but lacks native unicode and modern OpenType font support. These are not serious restrictions.
- ***xelatex*** A more modern version of ***pdflatex*** that fixes its downsides. But on the flipside it is slower and has not been actively developed since 2017.
- ***lualatex*** The most modern engine is scriptable through Lua. It also fixes the issues with ***pdflatex*** and is under active development. It is even slower though.
  - This is the way to go for new projects.
  - From this point on, we will assume ***lualatex***.
  - Don't worry too much about the lack of speed. It is rare that it becomes an annoyance.
  - While you can script it in Lua, the interactions with the LaTeX macro system are complex. Personally, I have decided against it.

Distributions

For you to work with LaTeX, you will need:

- A number of appropriate fonts.
- A LaTeX engine for compiling.
- A collection of packages with extensions.
- A number of other support programs.

All of this combined is referred to as a LaTeX distribution.

There exists a number of precompiled distributions:

- ***TeX Live*** The typical distribution (Linux, Windows and OSX).
- ***MiKTeX*** If you are on Windows (but also works in Linux and OSX).
- ***MacTeX*** If you are on a Mac.

Commandline

Running the LaTeX interpreter is done using the following command:

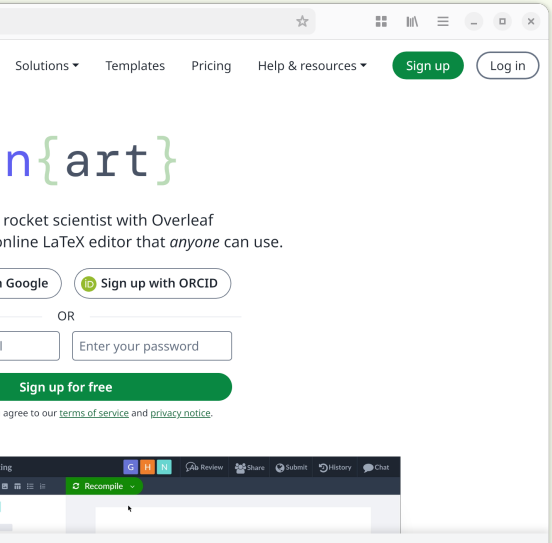
```
lualatex document.tex
```

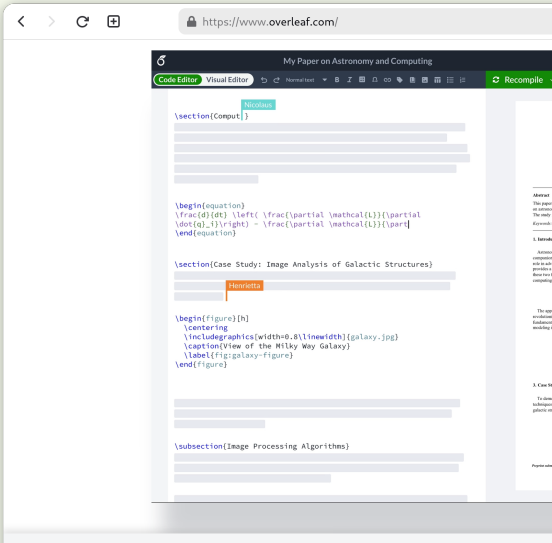
or, if you are using `minted` for source code highlighting:

```
lualatex -shell-escape document.tex
```

Note: When using references, you may have to run the command multiple times.

Overleaf


Solutions
Templates
Pricing
Help & resources
Sign up
Log in
n{art}
rocket scientist with Overleaf
online LaTeX editor that *anyone* can use.
Google
Sign up with ORCID
OR
Enter your password
Sign up for free
agree to our terms of service and privacy notice.
Review
Share
Submit
History
Chat
Recompile


https://www.overleaf.com/
My Paper on Astronomy and Computing
Code Editor
Visual Editor
Recompile
Nicolaus
Henrietta
\section{Comput }
\begin{equation}
\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial
\dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\part
\end{equation}
\section{Case Study: Image Analysis of Galactic Structures}
\begin{figure}[h]
\centering
\includegraphics[width=0.8\linewidth]{galaxy.jpg}
\caption{View of the Milky Way Galaxy}
\label{fig:galaxy-figure}
\end{figure}
\subsection{Image Processing Algorithms}

Overleaf ▷ Hosting Options

Cloud:

- Paid and free tier.
- Paid tier provides GIT support.
- Issues with uptime.

Self-hosted:

- No GIT support.
- But if all else fails, you can pull the data out yourself.

Building with Make

```
TARGETS = \
        introduction_presentation.pdf \
        intermediate_presentation.pdf \

INTRODUCTION_DEPS = \
        ./figs/vector.pdf \

INTERMEDIATE_DEPS = \
        ../src/table_heatmap/code.txt \
        ../src/table_heatmap/code.tex \
        figs/sdi.tex \
        .intermediate.timestamp \


all: ${TARGETS}

clean:
        touch ${TARGETS}
        rm    ${TARGETS}

mrproper: clean
        touch dummy~ d.aux d.log d.nav d.out d.snm d.toc d.vrb d.minted
        rm        *~ *.aux *.log *.nav *.out *.snm *.toc *.vrb *.minted


.intermediate.timestamp: ../bin/generate-tex-includes intermediate_*.tex ./figs/vector0.svg
        cd .. ; ./bin/generate-tex-includes PDF doc/intermediate_presentation.tex doc
        touch .intermediate.timestamp

./figs/vector.pdf: ./figs/vector.svg
        inkscape ./figs/vector.svg -z -D -o ./figs/vector.pdf

./figs/vector0.svg: ../bin/process-vector.py ./figs/vector.svg
        ../bin/process-vector.py
```

Versioning with GIT

GIT is used for keeping tract of revisions of (primarily) human-readable files.

This is usually used for program code.

LaTeX is a programming language, and when we execute LaTeX code (through an interpreter) then we get a PDF file.

A GIT repository can easily contain both your program code and your report.

Typical structure:

- Program code in `/src` of the repository.
- Report and presentation in `/doc` of the repository.
- Figures shared between report an presentation in `/doc/figs` of the repository.

Part 3
Basics

Overall Structure

```
\documentclass[a4paper, oneside]{memoir}
```

Preample

```
\begin{document}
```

Front Matter

Main Matter

Back Matter

```
\end{document}
```

Overall Structure ▷ Packages

A package is a LaTeX term for a file (or collection of files) that contain a set of macros (commands and environments) for dealing with a specialized topic.

The `dirtytalk` package gives access to the `\say` command. This command takes one parameter and wraps it in the proper quotation marks, like “so”.

It can be included by putting the following in the *preample*:

```
\usepackage{dirtytalk}
```

This is a very simple package. Others – like TikZ and `minted` – are much more extensive.

Overall Structure ▷ Language and Hyphenation

All of this goes into the *preample*.

Most parts of LaTeX has multilingual support, selectable through the `babel` package:

```
% use english text to ToC, sections, figures ...
\usepackage[english]{babel}
```

Custom hyphenation rules can be added:

```
\usepackage{hyphenat}
\hyphenation{al-go-rithm me-cha-nics}
```

Overall Structure ▷ File Inclusion

As programmers, we usually have a good understanding of the problems that can stem from having all of our code in a single file.

In LaTeX one can, in essence, include the contents of an other file like so:

```
\input{filename.tex}
```

This is often used for top-level sections or figures.

As programmers, we rarely have a good understanding of the problems of having all of our code split across an ocean of files.

So, use with care.

Overall Structure ▷ Structure of Contents

The *main matter* is split into a section tree, with lists of paragraphs as leaves.

What is available to you depends on your *style*.

```
\part{Reasoning}
\chapter{Introduction}

paragraph 1

paragraph 2

\section{Problem}
\section{Approach}
\chapter{Analysis}
\section{Themes}
\subsection{Build Systems}
\subsection{Scripting}
\section{Sub-Conclusion}
```

Commands and Environments

A *command* is a macro that can be expanded by writing its name prefixed y a backslash and with parameters, each in a set of curly braces:

```
\say{Hello, World}
```

An environment is another form of macro, that can be expanded by wrapping code in a `\begin` and an `\end` command like so:

```
\begin{center}
  Some text.
\end{center}
```

Spacing

Whitespaces are treated the same way as in most other programming languages: Either there is at least one, or there is none. No need to worry about having two spaces instead of one.

LaTeX will break the line as it tries to optimize the document layout (by minimizing *badness*) according to the rules of the style.

Note: This can make it hard to force a specific placement.

Horizontal and vertical space can be added with the `\hspace` and `\vspace` commands. They both take one parameter; a distance.

Spacing ▷ Horizontal Space

```
first\hspace{1cm}last
```

first last

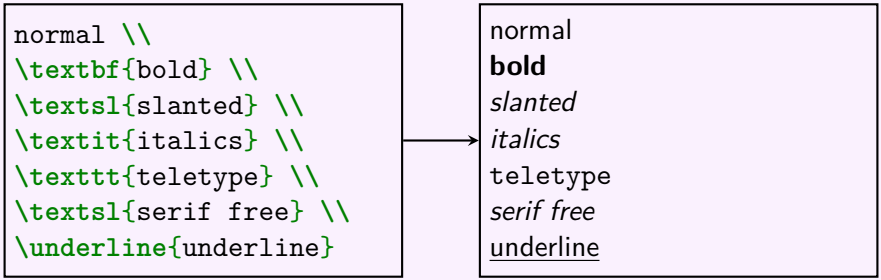
Spacing ▷ Vertical Space

```
first

\vspace{1cm}
last
```

first

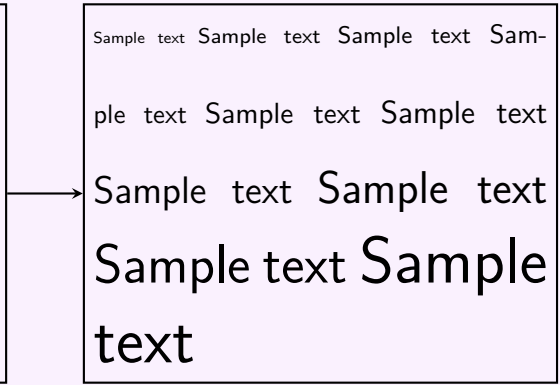
last

Formatting


normal \\
\textbf{bold} \\
\textsl{slanted} \\
\textit{italics} \\
\texttt{teletype} \\
\textsl{serif free} \\
\underline{underline}
normal
bold
slanted
italics
teletype
serif free
underline

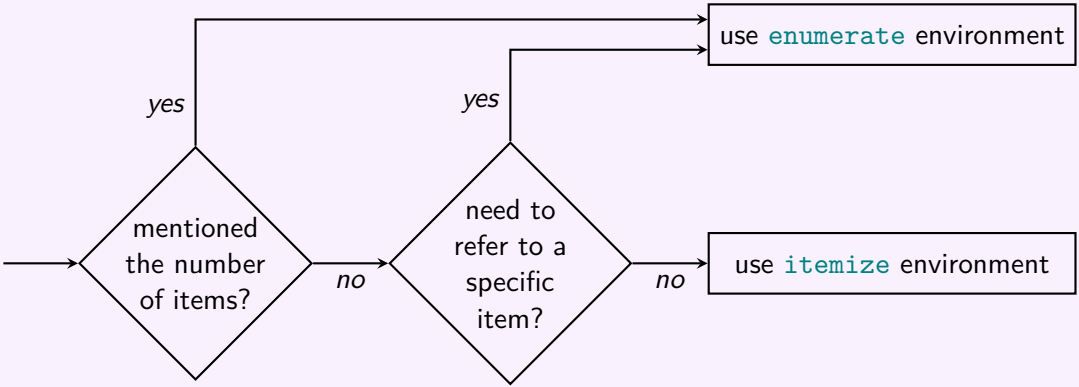
Text Size

```
\tiny{Sample text}
\scriptsize{Sample text}
\footnotesize{Sample text}
\small{Sample text}
\normalsize{Sample text}
\large{Sample text}
\Large{Sample text}
\LARGE{Sample text}
\huge{Sample text}
\Huge{Sample text}
```



Listings

General considerations when deciding between ordered or unordered lists:



Listings ▷ Itemization

```
Programming languages:
\begin{itemize}
  \item Rust
  \item Elixir
  \item Python
  \item \LaTeX ?
\end{itemize}
```

Programming languages:

- Rust
- Elixir
- Python
- LaTeX?

Listings ▷ Enumerations

```
Programming languages:
\begin{enumerate}
  \item Rust
  \item Elixir
  \item Python
  \item \LaTeX ?
\end{enumerate}
```

Programming languages:

1. Rust
2. Elixir
3. Python
4. LaTeX?

References

Numbered *things* can be labeled and referenced by their number or their page.

These include:

- Section at any level of depth.
- Figures and tables.
- Equations.
- Enumerated lists.

They do not include:

- Bibliography.

References ▷ Use

A label is created using the `\label` command. This will refer to the latest numerable construct.

The number referred to by a label can be accessed using the `\ref` command. Be sure to preface it with something that indicates what you are referring to (e.g., a section).

The page number containing a label can be accessed using the `\pageref` command.

Note: References need two parses of the LaTeX compiler to function:

- First pass notes down the position of the labels.
- Second pass inserts these positions at the references.

References ▷ Example

```
Algorithm:
\begin{enumerate}
  \item Initialize.
  \item \label{id} Do stuff.
  \item Go to step \ref{id}.
\end{enumerate}
```

Algorithm:

1. Initialize.
2. Do stuff.
3. Go to step 2.

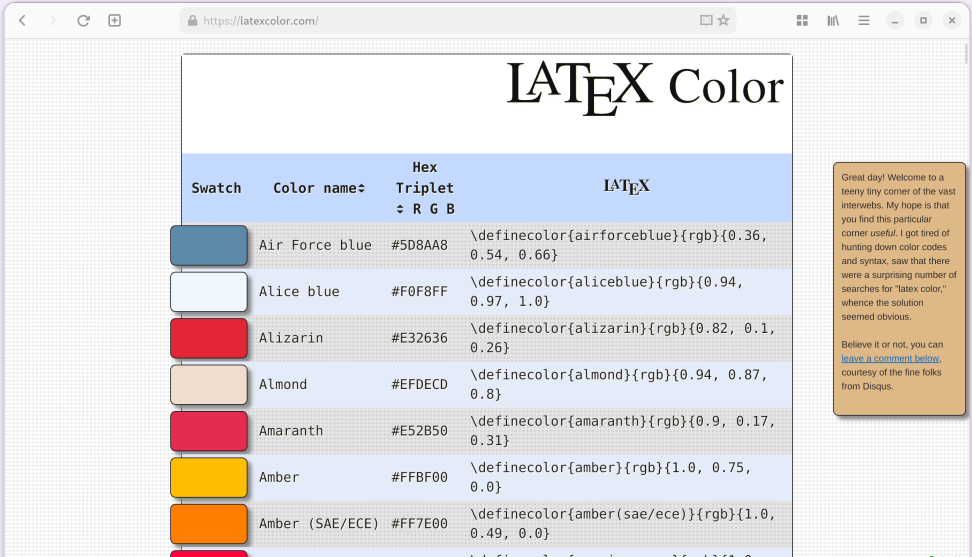
Colors

```
% -- preample

\usepackage{color}

% -- main matter

\definecolor{myfavcolor}{rgb}{0.49,0.79,0.62}

Text can be \textcolor{purple}{colored}!

Colors can be
\textcolor{myfavcolor}{customized}!

And they can be declared
\textcolor{red!27!green}{inline}!
```

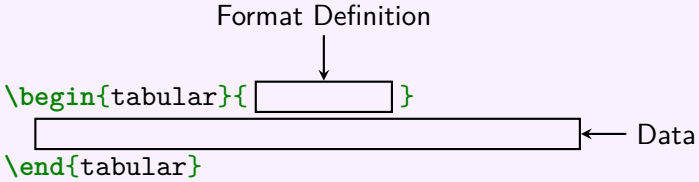
Text can be colored!
Colors can be customized!
And they can be declared inline!

LaTeX provides a set of predefined color names: https://latexcolor.com

Colors ▷ Predefined



Arranging Tabular Data ▷ Environment



Arranging Tabular Data ▷ Format Definition

The format definition specifies how to arrange the cells of a single row, and where to place vertical lines.

Consider the row data a queue of cells.

Options include:

- "`|`" Place a vertical bar.
- "`l`" Consume a cell and left justify its contents.
- "`c`" Consume a cell and center its contents.
- "`r`" Consume a cell and right justify its contents.
- "`p{5cm}`" Consume a cell and make a (multi-line) paragraph out of its contents.

Arranging Tabular Data ▷ Data

In the data segment we have a sequence of horizontal lines (`\hline`) and rows of data.

A row of data needs to respect the format of the `tabular` environment. This means that:

- The cell contents needs to be separated by a `&` character.
- Each row needs to be terminated by a manual linebreak(`\\`).

Cells can be merged and their background colored. For this, use search terms like "multirow", "multicol" and "cell color".

Arranging Tabular Data ▷ Example

```
\begin{tabular}{|lrr|}
  \hline
  \textbf{Country} & \textbf{Area/[$km^2$]} & \textbf{Population} \\
  \hline
  Denmark & 43.094  &  6.001.008 \\
  Greece  & 131.957 & 10.400.720 \\
  Finland & 338,455 &  5.635.971 \\
  \hline
\end{tabular}
```

Arranging Tabular Data ▷ Result

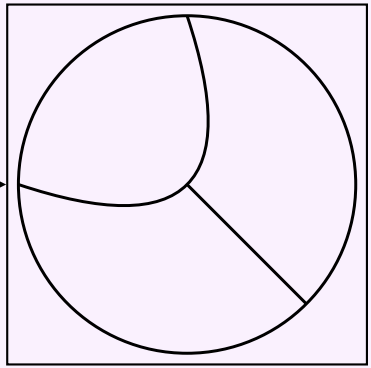
Country	**Area/[km^2]**	**Population**
Denmark	43.094	6.001.008
Greece	131.957	10.400.720
Finland	338,455	5.635.971

Including Visuals

When it comes to visual aids, LaTeX supports a number of different types.

Including Visuals ▷ PDF Graphics

```
\includegraphics[width=45mm]{
  ./figs/vector.pdf
}
```



Including Visuals ▷ SVG Graphics

While that figure was included as a PDF file, it was created as an SVG file using *Inkscape*: https://inkscape.org

There is, however, no native process for including an SVG file in LaTeX.

Instead, we can use Inkscape to convert it to PDF:

```
$ inkscape ./figs/vector.svg -z -D -o ./figs/vector.pdf
```

That line is straight out of the build system for this presentation.

And the SVG file, in its entirety, looks like this ...

Including Visuals ▷ SVG Source

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!-- Created with Inkscape (http://www.inkscape.org/) -->

<svg
   width="50mm"
   height="50mm"
   viewBox="0 0 50 50"
   id="SVGRoot"
   version="1.1"
   inkscape:version="1.4 (e7c3feb100, 2024-10-09)"
   sodipodi:docname="vector.svg"
   xmlns:inkscape="http://www.inkscape.org/namespaces/inkscape"
   xmlns:sodipodi="http://sodipodi.sourceforge.net/DTD/sodipodi-0.dtd"
   xmlns="http://www.w3.org/2000/svg"
   xmlns:svg="http://www.w3.org/2000/svg">
  <sodipodi:namedview
     inkscape:document-units="mm"
     inkscape:zoom="5.0329577"
     inkscape:cx="94.278559"
     inkscape:cy="107.09409"
     id="namedview1"
     pagecolor="#ffffff"
     bordercolor="#000000"
     borderopacity="0.25"
     inkscape:showpageshadow="2"
     inkscape:pageopacity="0.0"
     inkscape:pagecheckerboard="0"
     inkscape:deskcolor="#d1d1d1"
     inkscape:window-width="1920"
     inkscape:window-height="1131"
     inkscape:window-x="0"
     inkscape:window-y="32"
     inkscape:window-maximized="1"
     inkscape:current-layer="layer1"
     showgrid="true">
    <inkscape:grid
       id="grid1"
       units="mm"
       originx="0"
       originy="0"
       spacingx="0.1"
       spacingy="0.1"
       empcolor="#0099e5"
       empopacity="0.30196078"
       color="#0099e5"
       opacity="0.14901961"
       empspacing="10"
       enabled="true"
       visible="true" />
  </sodipodi:namedview>
  <defs
     id="defs1" />
  <g
     inkscape:label="Layer 1"
     inkscape:groupmode="layer"
     id="layer1">
    <ellipse
       style="fill:none;stroke:#000000;stroke-width:0.264583"
       id="path3"
       cx="24.999998"
       cy="25"
       rx="14.999999"
       ry="15" />
    <path
       style="fill:none;stroke:#000000;stroke-width:0.264583"
       d="m 10,25 c 15,5 20,0 15,-15"
       id="path4"
       sodipodi:nodetypes="cc" />
    <path
       style="fill:#ffffff;stroke:#000000;stroke-width:0.264583"
       d="M 25,25 35.600001,35.600001"
       id="path5"
       sodipodi:nodetypes="cc" />
  </g>
</svg>
```

Including Visuals ▷ Bitmapped Graphics

```
\includegraphics[width=45mm]{
  ./figs/dscf0013016.png
}
```

Floats

When writing a report, you should expect your reader to follow the main track of the text. It should be possible to read it from beginning to end.

But at times you may want to provide extra materials though figures and tables. These should not break the flow of the text.

Instead, they should exists outside the flow of the text, and be referenced. The typesetting system should find space for them closeby, at a location where they don't break the flow.

In short, they should *float*.

Floats ▷ Figures and Tables

You define a figure like this:

```
\begin{figure}[tbp]
  % contents goes here
  \caption{Description of figure}
  \label{fig:tag}
\end{figure}
```

You define a table like this:

```
\begin{table}[tbp]
  % contents goes here
  \caption{Description of table}
  \label{table:tag}
\end{table}
```

Floats ▷ Positioning

```
\begin{figure}[tbp]
  % contents goes here
  \caption{Description of figure}
  \label{fig:tag}
\end{figure}
```

← What about this?

These are the positioning instructions. It instructs LaTeX where it is allowed to place the float. It supports the following codes:

- "`t`" Float can be placed at the top of a page.
- "`b`" Float can be placed at the bottom of a page.
- "`p`" Float can be placed at a page with nothing but floats.
- "`h`" Float should be placed *approximately here*.
- "`!`" Override internal parameters that LaTeX uses to determine *good* positions.

In general, avoid the latter two.

Footnotes

```
Text\footnote{Short
detailing.}.
```

Text[a].

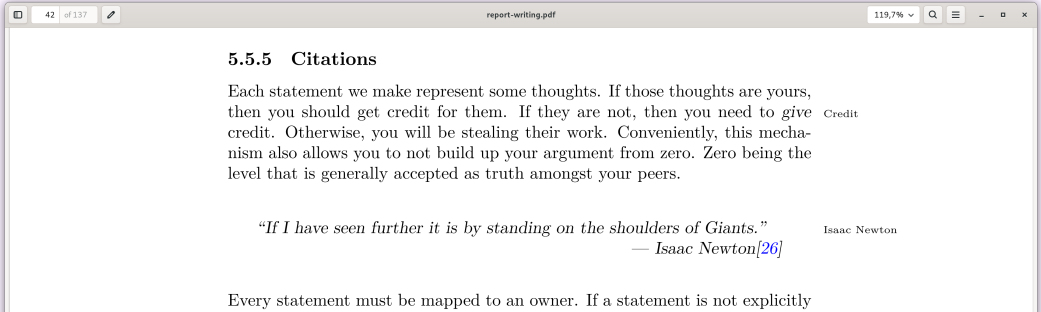
[a] Short detaining.

Margin Paragraphs

Margin paragraphs works in much the same way as footnotes.

However, they are (i) not numbered, and (ii) appear in the outer margin.

Instead of the `\footnote` command you use `\marginpar` (and make sure that the `marginnote` package is included).



Code Inclusion

There is a number of packages for including code. In this presentation we will look at `minted`.

The `minted` package support a long list of file format, and even things like shell interactions.

It also support various forms of highlights and line numbering.

Code Inclusion ▷ Inline

```
The formatting of a function
call in rust looks like this:
\mintinline{rust}{fun(1+1)}.
```

The formatting of a function call in rust looks like this: `fun(1+1)`.

Code Inclusion ▷ Environment

```
The code could look like this:

\begin{minted}{elixir}
doubles =
  [1,2,3,4]
  |> Enum.map(fn value -> value*2 end)
\end{minted}

And that's not too bad, right?
```


The code could look like this:

```elixir
doubles =
  [1,2,3,4]
  |> Enum.map(fn value -> value*2 end)
```

And that's not too bad, right?

Mathematics

One of the things LaTeX is know for is its ability to typeset mathematical expressions.

At first, it can be quite daunting to remember the syntax and code for all the different constructs.

But the reality is that there is a system behind the madness, and the set of constructs you are going to be using on a daily basis is very small and easily remembered.

For instance, the command for producing a fraction is called `\frac` and takes two parameters: First a nominator, and then a denominator. The `\sqrt` command produces a square root symbol and takes one parameter.

And the web is littered with tables containing codes for Greek letters and all kinds of weird symbols. Just do an image search.

Mathematics ▷ Inline

```
The Pythagorean theorem
states that $a^2+b^2 = c^2$,
but can also be expressed as
$c = \sqrt{a^2+b^2}$.
```

The Pythagorean theorem states that $a^2 + b^2 = c^2$, but can also be expressed as $c = \sqrt{a^2 + b^2}$.

Mathematics ▷ Equations

```
\begin{align*}
  a^2 &= b^2 + c^2 \Leftrightarrow \\
  a   &= \sqrt{b^2 + c^2}
\end{align*}
```

$$a^2 = b^2 + c^2 \Leftrightarrow$$
$$a = \sqrt{b^2 + c^2}$$

Mathematics ▷ Equation References

```
\begin{align}
  \label{pyth1}
  a^2 &= b^2 + c^2 \Leftrightarrow \\
  \label{pyth2}
  a   &= \sqrt{b^2 + c^2}
\end{align}
Pythagoras can be expressed as
either of equations \ref{pyth1}
or \ref{pyth2}.
```

$$a^2 = b^2 + c^2 \Leftrightarrow \quad (1)$$

$$a = \sqrt{b^2 + c^2} \quad (2)$$

Pythagoras can be expressed as either of equations 1 or 2.

Mathematics ▷ Raising and Lowering Text

```
Water is $H_2O$.

$H_{20}$ is not a thing.

The square of three is $3^2$.
```

Water is H_2O.
H_{20} is not a thing.
The square of three is 3^2.

URLs

```
Search at
\textcolor{blue}{
  \texttt{https://search.brave.com}
}

Search at
\url{https://search.brave.com}

Search at
\href{https://search.brave.com}{Brave}
```

Search at https://search.brave.com

Search at `https://search.brave.com`

Search at Brave

Nooks and Crannies

The `\ldots` can be used to add three periods ...

Just as the `\LaTeX` command, it will eat any following space. If this is not what you want, then you have to escape the following space with a backslash.

The `center` environment and a `\centering` command can be used to center elements.

Part 4
Auto-Generated Lists

Table of Contents

When constructing the table of contents (aka ToC), we can decide how deep we want it to follow into the documents section tree.

If we want to include subsections, then we set the `tocdepth` counter to 2:

```
\setcounter{tocdepth}{2}
```

Alternatively, we can omit this.

The ToC itself can be placed using this command:

```
\tableofcontents
```

Note: All of this goes into the front matter of the document.

List of Figures and Tables

A list of figures can be inserted using the command:

```
\listoffigures
```

A list of tables can be inserted using the command:

```
\listoftables
```

Bibliography

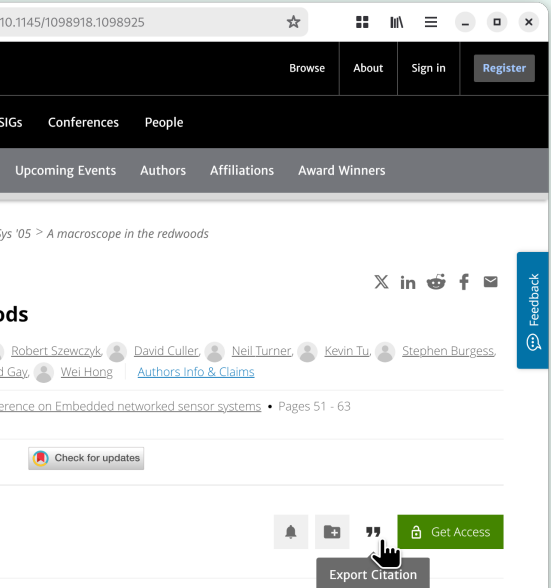
When writing, we need to give and to take credit for the individual parts that make up our work.

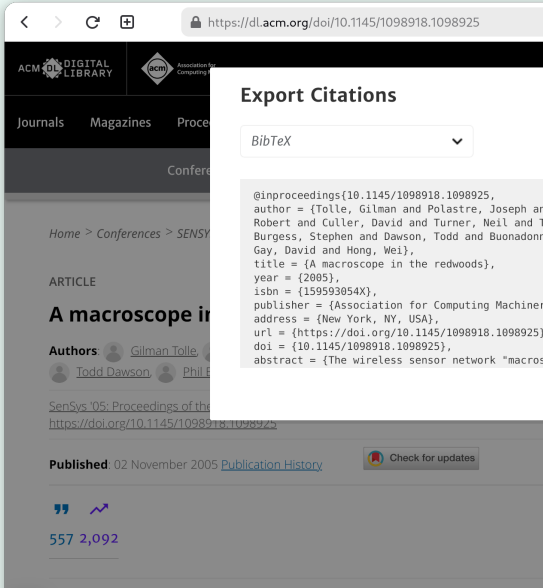
- We build on what others have accomplished, and give them credit for their work.
- We come up with new insights, and take credit for those.

In order to give credit to some work, we *cite* it.

LaTeX has an extensive framework for working with such citations.

Bibliography ▷ Database Population





Bibliography ▷ Database

Remember this!

```
@inproceedings{10.1145/1098918.1098925,
  author = {Tolle, Gilman and Polastre, Joseph and Szewczyk, Robert and Culler, David and Turner, Neil and Tu, Kevin and
  ↪  Burgess, Stephen and Dawson, Todd and Buonadonna, Phil and Gay, David and Hong, Wei},
  title = {A macroscope in the redwoods},
  year = {2005},
  isbn = {159593054X},
  publisher = {Association for Computing Machinery},
  address = {New York, NY, USA},
  url = {https://doi.org/10.1145/1098918.1098925},
  doi = {10.1145/1098918.1098925},
  abstract = {The wireless sensor network "macroscope" offers the potential to advance science by enabling dense temporal and
  ↪  spatial monitoring of large physical volumes. This paper presents a case study of a wireless sensor network that
  ↪  recorded 44 days in the life of a 70-meter tall redwood tree, at a density of every 5 minutes in time and every 2
  ↪  meters in space. Each node measured air temperature, relative humidity, and photosynthetically active solar radiation.
  ↪  The network captured a detailed picture of the complex spatial variation and temporal dynamics of the microclimate
  ↪  surrounding a coastal redwood tree. This paper describes the deployed network and then employs a multi-dimensional
  ↪  analysis methodology to reveal trends and gradients in this large and previously-unobtainable dataset. An analysis of
  ↪  system performance data is then performed, suggesting lessons for future deployments.},
  booktitle = {Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems},
  pages = {51-63},
  numpages = {13},
  keywords = {wireless sensor networks, microclimate monitoring, macroscope, application analysis},
  location = {San Diego, California, USA},
  series = {SenSys '05}
}
```

Bibliography ▷ Citation and Bibliography Generation

Assuming that you have that database stored in `references.bib`, put the following in your preample:

```
\usepackage[
  backend=biber,
]{biblatex}
\addbibresource{references.bib}
```

Now you can start citing your statements:

```
Some profound statement\cite{10.1145/1098918.1098925}.
```

We can then add the bibliography in the back matter:

```
\printbibliography
```

In between the first and the later runs of LaTeX, make sure to run the following command:

```
biber document.bcf
```

Index

At its core, an index allows us to look up a term and get a set of pages relevant for that term.

To tag a position for inclusion in the index under a specific term, use the command:

```
\index{Term}
```

At the back matter of your document, then add:

```
\printindex
```

In between the first and the later runs of LaTeX, make sure to run the following command:

```
makeindex document.idx
```

It will generate your index.

Part 5
Dealing with Errors

Locating the Error

Unfortunately, LaTeX errors are notorious for being hard to reason about.

Often, the line number does not reflect the actual line in the source file.

Because of this, I frequently build the document.

If in doubt, use binary search by commenting out large blocks of code to locate the issue.

Googling the Error

Most errors are easily googleable, and will yield hits on sites like StackOverflow.

In order to get good results, you should:

- Carefully consider which part to copy paste in. These errors can be verbose.
- Consider the type of error.
- Try to figure out what could cause such an error, and figure out whether these conditions apply to you.
- Don't be afraid to take an extra step down the rabbit hole.

Part 6
Resources

General Resources

Wide resources:

- WikiBooks LaTeX: https://en.wikibooks.org/wiki/LaTeX
- Overleaf Learn: https://www.overleaf.com/learn

Code generators:

- Table generator: https://www.tablesgenerator.com/latex_tables
- Equation editor: https://latexeditor.lagrida.com

Misc:

- Ti*k*Z: https://tikz.dev

Source and Templates

The source code for these slides can be found here:
https://github.com/aslakjohansen/prosa-latex

Templates for reports and presentations can be found here:
https://github.com/aslakjohansen/latex-template

More material will appear at https://latex.asjo.dk

Part 7
Exercises

Exercises

1. ***New Project:*** Create a new project (on Overleaf, GitHub or similar). Make sure that you have a report LaTeX document that builds.
2. ***Tabular Data:*** Find some tabular data, place it in a `tabular` environment, and make it look pretty.
3. ***Formula:*** Pick any formula (e.g., the discrete fourier transformation), and express it though LaTeX code.
4. ***Report:*** Pick any old report, and port it to LaTeX. The point is not to go through the whole report, but rather to get a feel of (i) the effort needed to port an old report, and (ii) the perceived typographical quality per effort.