



**GHENT
UNIVERSITY**



TYPST: A MODERN L^AT_EX REPLACEMENT

Workshop for 2nd year MSc. in Photonics Student

Outline


1. Introduction
2. It's hammer Typst time!
3. Let's get started!
4. Did you say demo?
5. About your thesis

INTRODUCTION


Why?

- LaTeX is **old**
- LaTeX is **hard**
- LaTeX is **ugly**
- Typst is **modern**
- Typst is **easy**
- Typst is **beautiful**

```
1 \title{My first LaTeX}
2 \author{Hubert}
3 \date{August 2022}
4 \begin{document}
5 \maketitle
```



```
1 // Easy as pie
2 #set document(
3   title: "My first Typst"
4   author: "Hubert Farnsworth",
5 )
```



- LaTeX is **slow**
- Typst is **fast**

Is LaTeX really that old?

- TeX was created in 1978 by **Donald Knuth**
 - Dude is absolutely bad-ass
 - Wrote “The Art of Computer Programming”
 - Created **METAFONT** & **Computer Modern**
 - Line-breaking algorithm
 - Document format: **DVI**
 - Created **WEB**
- LaTeX was created in 1983 by **Leslie Lamport**
 - Set of macros of TeX
 - Easier to use
 - Closer logical structure \leftrightarrow visual structure

Is it really that hard?

- LaTeX is known for its cryptic errors

```
1  ! Undefined control sequence.  
2  \enit@setresumekeys ...it@toks }\ifnum \enit@type  
3      =\z@ #3\def \enit@noexcs {...
```

- People just Google for info
- The documentation is often hard to search

LaTeX is ugly?

“Because it [LaTeX] is so hacky and messy. [...]”

— u/atloomis

- Have you ever wondered what goes on in your `documentclass`?

```
1  \def\@citex[#1]#2{%  
2  \let\@citea\@empty  
3  \@cite{\@for\@citeb:=#2\do  
4    {\@citea\def\@citea{], []}%  
5    \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
```


L^AT_EX

IT'S ~~HAMMER~~ TYPST
TIME!

What sets Typst apart?

- A **real** programming language


```
1  #let fib(n) = {  
2    if n <= 1 {  
3      1  
4    } else {  
5      fib(n - 1) + fib(n - 2)  
6    }  
7  }
```



What sets Typst apart?


- A **real** programming language

```
1  #let fib(n) = {  
2    if n <= 1 {  
3      1  
4    } else {  
5      fib(n - 1) + fib(n - 2)  
6    }  
7  }
```



- With **powerful** markup syntax

```
1  === My_paragraph  
2  Hello, world!  
3  *This text is in bold*.  
4  _And this one is emphasized_.  
5  #strike[This one is  
6    struck through].
```



What sets Typst apart?

- A simple **free** web-app for **everything**
 - Collaborative editing
 - Instant preview
 - Cloud storage
- Or running on your PC using a **single** binary
 - macOS, Linux, or Windows
 - No dependencies
 - Easy to install
 - Easy to update
- **Incredible** documentation
- **Packages** at your fingertips

LET'S GET STARTED!

The basics

- A Typst document is a **markup** with **code**
- The markup is the default mode or when surrounded in `[` and `]`.
- The `#` character is used to switch to code mode.

```
1 This is markup
2 #this-is-code()
3 #[ This is also markup ]
```



The basics (cont.)

- Heading levels are defined with `=` characters

```
1  = Heading 1
2  == Heading 2
3  === Heading 3
```



- Lists are defined with `-` characters

```
1  - Item 1
2  - Item 2
3    - Item 2.1
```



The basics (cont.)

- Numbered lists are defined with `+` characters

```
1  + Item 1
2  + Item 2
3      + Item 2.1
```



- Strong emphasis is done with `*` and `_` characters

```
1  *This is strong*
2  _This is emphasized_
```



The basics (cont.)

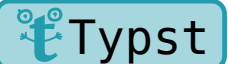
- You can create labels using the `<label>` syntax:

```
1  = Chapter 1 <my-label>
```



- And you can reference them using the `@` syntax:

```
1  @my-label // This creates a clickable link
```



- It works the same for bibliographies!

```
1  @my-bib-entry // This creates a clickable link
```



The basics (cont.)

- You can insert comments using `//` and `/* */`

```
1 // This is a comment
2 /* This is a block comment */
```



- You can insert code using the `#` character

```
1 #let x = 1
2 #let y = 2
3 #let z = x + y
```



The basics (cont.)

- You can insert a block of code using the `#` character

```
1  #{  
2    let x = 1  
3    let y = 2  
4  }
```



- You can declare functions and variables using the `let` keyword

```
1  #let add(a, b) = a + b  
2  #let c = add(5, 6)
```



Bibliographies

- Please use **Zotero** for managing your bibliographies
- You can export your bibliography to a `.bib` file
- Or you can use **Hayagriva**'s `.yaml` files
- You can then import them in your Typst document

```
1 #bibliography("my-bibliography.bib", style: "ieee")
```



- You can then cite them using the `@` syntax

```
1 @my-bib-entry // Will display like [1] and be clickable.
```



- Everything will be formatted automatically!

Images and figures

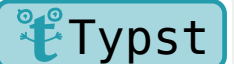
- You can insert images using the `#image` function

```
1 #image("my-image.png")
```



- And wrap them in a figure using the `#figure` function

```
1 #figure(caption: "My caption") [  
2   #image("my-image.png")  
3 ] <my-image>
```



- It automatically detects the type of figure (image, table, etc.)
- You can reference it by giving it a label and referencing it using the `@` syntax.

Equations

- Two types: inline and display
- Inline equations are surrounded by `$` characters: $x^2 + y^2 = z^2$

```
1 $x^2 + y^2 = z^2$
```



- Display equations are surrounded by `$` characters with a space:

$$x^2 + y^2 = z^2$$

```
1 $ x^2 + y^2 = z^2 $
```



- You can also give them a label and reference them.
- Syntax is similar to LaTeX but with a few differences.

Outlines & queries

- Outlines are built from a query
- Queries allow you to ask questions of your document

```
1  #locate(loc => {  
2    query(heading.where(level: 1), loc)  
3  })
```



- And are made easy to use

```
1  #outline(target: figure.where(kind: image))
```



Packages

- Packages are a way to extend Typst
- They are written in Typst itself
- They are easy to import

```
1 #import "@preview/polylux:0.3.1": *
```



- And easy to use

```
1 #show: codly-init.with()  
2 #codly()
```



- Downloaded on demand

show rules, and set commands

- `show` rules are used to change how an object is displayed

```
1  #show link: it => text(color: blue, it)
2  #link("https://www.google.com/") will be blue.
```



- `set` commands are used to change the state of a function

```
1  #set text(font: "New Computer Modern")
2  This text will be in New Computer Modern.
```



- Rules are applied in order of appearance
- Rules are scoped to the current block

Nifty_packages

- `@preview/polylux` for making slides (like this one)
- `@preview/codly` for beautiful code blocks
- `@preview/cetz` for creating diagrams
- `@preview/tablex` for creating beautiful tables
- `@preview/glossarium` for creating glossaries
- `@preview/lemmify` and `@preview/ctheorems` for creating theorems
- `@preview/jogs` to run JS code
- `@preview/pyrunner` to run Python code from Typst
- So many more

DID YOU SAY DEMO?

ABOUT YOUR THESIS

Use these tools

- Zotero for managing your bibliography
- Typst for writing your thesis
- <https://draw.io> for creating diagrams
- The Typst Discord server: <https://discord.gg/2uDybryKPe>

Thanks for coming!

- <https://typst.app>
- <https://discord.gg/2uDybryKPe>
- <https://github.com/typst/typst>
- <https://github.com/Dherse/masterproef>
- <https://github.com/Dherse/ugent-templates>
- Questions?

Sébastien d'Herbais de Thun

Student

E sebastien.dherbaisdethun@ugent.be

www.ugent.be

 Universiteit Gent

 @ugent

 @ugent

 Ghent University

 Sébastien d'Herbais de Thun