Typst Markup typesetting

START HERE

`typst watch <file.typ>`
Recompile on every save

`typst compile <file.typ>`
Build a PDF once

`= Title`
Heading

`*bold* _italic_`
Emphasis

`- item`
Bullet list

`$ x^2 $`
Display math

`#set text(size: 11pt)`
Style from here on

`#image("<path>")`
Insert a picture

Markup text, = * _ → **Code** let, set, show (`#`) → **Content** back in markup (`[ ]`) → **PDF / PNG / SVG** one file or one per page (`compile`)

CONCEPTS

Markup is the default mode. # switches to code for one expression, [ ] switches back, and $ $ is math.

typst watch recompiles on save; open the PDF in a viewer that reloads changed files.

Typst is not LaTeX: no preamble, no packages to install by hand. Packages download on first use.

COMPILE AND WATCH

`typst compile <file.typ>`
Compile to a PDF next to the source
Add an output path as a second argument; the extension picks the format.

`typst watch <file.typ>`
Recompile on every change

`typst init @preview/<template>`
New project from a template
Append :<version> to pin one; the directory defaults to the template name.

`typst compile <file.typ> <out>.png`
Export to PNG or SVG
Several pages need a {p} or {0p} placeholder in the output name.

`typst compile <file.typ> --pages 2-4`
Export only some pages

`typst compile --input <key>=<value> <file.typ>`
Pass a value into the document
Read it in the document with sys.inputs.<key>.

`typst compile --root <dir> <file.typ>`
Set the root for absolute paths

`typst compile --font-path <dir> <file.typ>`
Add a font directory

`typst fonts`
List fonts Typst can use

`typst --version`
Show the installed version

`typst fonts --variants`
List fonts with their style variants

`typst compile - <out>.pdf`
Read the source from stdin

MARKUP

`= Heading`
Heading; more = for deeper levels

`*bold*`
Strong text

`_italic_`
Emphasis

`- item`
Bullet list entry
Indent to nest.

`+ item`
Numbered list entry

`` `code` ``
Inline monospace

`#link("<url>")[<text>]`
Hyperlink with custom text
A bare https:// URL is linked automatically.

`// comment`
Line comment

`` ```<lang> ``
Fenced code block

`\`
Forced line break

`\#`
Escape a special character

`#footnote[<text>]`
Footnote

`#pagebreak()`
Start a new page

`#v(<len>) / #h(<len>)`
Vertical or horizontal gap

`#align(center)[<content>]`
Align content

MATH

`$x^2 + y^2$`
Inline math (no spaces inside)

`$ x^2 + y^2 $`
Display math (spaces inside)

`a / b`
Fraction

`x^2 x_i`
Superscript and subscript

`sum_(i=1)^n i`
Sum with limits
Group multi-token sub- and superscripts in parentheses.

`sqrt(x)`
Square root

`mat(1, 2; 3, 4)`
Matrix, rows split by ;

`vec(1, 2, 3)`
Column vector

MATH (cont.)

`"text"`
Plain text inside math

`alpha pi → ≤ ≠`
Greek letters and symbols by name

LET, SET, SHOW

`#let <name> = <value>`
Define a variable

`#let <f>(<a>, <b>: 1) = [<body>]`
Define a function

`#set text(font: "<name>", size: 11pt)`
Font and size from here on

`#set page(paper: "a4", margin: 2cm)`
Paper size and margins

`#set page(numbering: "1")`
Number the pages

`#set par(justify: true, leading: 0.65em)`
Justify and line spacing

`#set heading(numbering: "1.1")`
Number headings

`#set text(lang: "de")`
Language for hyphenation

`#show heading: set text(blue)`
Restyle every heading

`#show: <template>.with(<args>)`
Apply a template to the document

FIGURES, TABLES, REFS

`#image("<path>", width: 50%)`
Insert an image
PNG, JPG, GIF, SVG and PDF are supported.

`#figure(<content>, caption: [<text>])`
Numbered figure with caption

`#table(columns: 3, [<a>], [<b>], [<c>])`
Table; cells fill row by row

`<label>`
Attach a label to the element before it

`@<label>`
Reference a label

`#grid(columns: (1fr, 1fr), gutter: 1em, ..)`
Grid layout

FIGURES, TABLES, REFS (cont.)

`#columns(2)[<content>]`
Split into columns

`#outline()`
Table of contents

`#bibliography("<file.bib>")`
Bibliography from BibTeX or Hayagriva

`@<key>`
Cite a bibliography entry

`#block(fill: luma(240), inset: 8pt)[<c>]`
Shaded box

CODE AND PACKAGES

`#import "@preview/<pkg>:<ver>": <name>`
Use a package from Typst Universe
Downloaded and cached on first compile.

`#import "<file.typ>": <name>`
Import from a local file

`#include "<file.typ>"`
Insert another file's content

`#for <x> in (1, 2, 3) [<body>]`
Loop

`#if <cond> [<a>] else [<b>]`
Conditional

`#json("<file>") / #csv("<file>")`
Load structured data

`#sys.inputs.at("<key>", default: none)`
Read a --input value safely

RECIPES

Minimal document

```
#set page(paper: "a4", margin: 2cm, numbering: "1")
#set text(font: "Libertinus Serif", size: 11pt)
#set heading(numbering: "1.1")

= Introduction
Hello *Typst*, see @fig.
#figure(image("plot.png", width: 70%), caption: [A plot]) <fig>
```

Reusable template

```
#let conf(title: "", body) = {
  set page(margin: 2cm)
  align(center, text(17pt, strong(title)))
  body
}
#show: conf.with(title: "My Report")
```