The Typst `dmunipi` theme

A guide on usage and customization of the theme

Master Degree in Mathematics

Francesco Baldino (025613)

23 December 2025

Dipartimento
di Matematica
Università di Pisa

Table of Contents

Introduction

This template is a Typst porting of Fabio Durastante's `dmslide` theme for Beamer. It's made with the Touying package for creating slides and slide themes.

Some features of the original theme might be missing, but all the fundamentals are already implemented, and should be easier to use thanks to Typst.

In the following you will find a brief introduction on how to use Typst and this theme to prepare slides.

Table of Contents

2 Basic usage

Getting started with Typst

If you’re new to Typst, the easiest way to start a new project is with the Typst web app, similarly to Overleaf for LaTeX. The web app is free, but it requires creating an account.

Once you’re set up in the web app, you can simply create a new empty project and procede from there.

If you’re more experienced or prefer to work locally, you can use the CLI typst compiler, which is open source. You can find instructions on how to install the CLI in the GitHub repo.

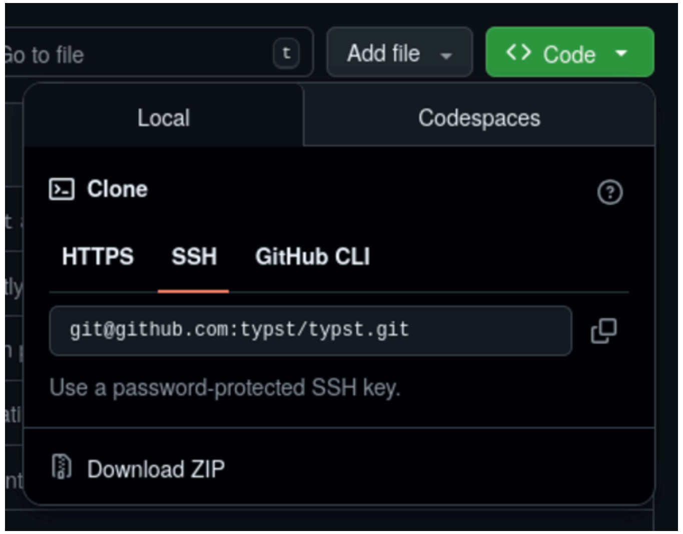
Once you’ve installed the CLI, creating a new project is as simple as creating a new folder and a `main.typ` file.

Getting started with this theme

To use this theme, you have to import the files of this repository in your empty project.

You can download the `ZIP` of this repository directly from GitHub through the green `<> Code` button in the top right.

From here you can either unzip it into your project folder if you're working locally, or unzip it and upload the resulting folder into the empty project on the web app.



```
git@github.com:typst/typst.git
```

Use a password-protected SSH key.

Now you can start editing the `main.typ` file! Note that only the `main.typ` file and `theme` folder are necessary, you can safely delete anything else.

Tip: is the font rendering weird? Go to the troubleshooting section

The `main.typ` file

Ignoring the comments, the `main.typ` file starts with the code on the right.

This is how the theme is imported and activated.

It's not important to understand right now what each keyword does, you just need to know that this is where the theme is activated and global informations like your title are declared.

If you want to override some of the styling of the theme, it must be done either here or below.

You can read the omitted comments to understand better what each option does and which options are available.

```
#import "./theme/lib.typ": *

#show: dmunipi-theme.with(
  config-info(
    title: [Your title],
    short-title: [yr ttl],
    subtitle: [Your subtitle],
    course: [Name of your course],
    author: [Your name],
    IDnumber: [012345],

    date: datetime.today(),
  ),
)
```

Anatomy of a Typst document

In its most basic form, a Typst document is composed of headings (sections, subsections, …) and paragraphs of text, like the following:

```
= Basic Usage

== Getting started with Typst

If you're new to Typst, the easiest way to start...
```

Within this theme, a section creates a new `Table of Contents` slide, and a subsection creates a new slide having the subsection title as its header, and the section title as the subheader.

Basic formatting

Simple text formatting such as bullets, numbering and text styling can be done as follows:

```
These are bullet points:
- something
- something else

The following are enumerated:
+ the first item
+ the second item

This word is *bold* while this word
is _italic_. This word is
`monospace`
```

➡️

These are bullet points:

- something
- something else

And these are enumerated:

1. the first item
2. the second item

This word is **bold** while this word is *italic*.
This word is `monospace`

Math text

Writing math in Typst is much easier than LaTeX, look at the following examples:

```
$ A = pi r^2 $

$ "area" = pi dot "radius"^2 $

$ cal(A) :=
    { x in RR | x "is natural" } $

#let x = 5
$ #x < 17 $

$ x_(n+1) = (x_n + a/x_n) / 2 $
```

➡️

$$A = \pi r^2$$

$$\text{area} = \pi \cdot \text{radius}^2$$

$$\mathcal{A} := \{x \in \mathbb{R} \mid x \text{ is natural}\}$$

$$5 < 17$$

$$x_{n+1} = \frac{x_n + \frac{a}{x_n}}{2}$$

you can find more informations here.

Coding

2 Basic usage

You can even display code with the correct syntax highlight by specifying the language:

~~~
Here is some rust code
```rust
fn main() {
}
```
And here is some Typst!
```typ
= Basic Usage

== Getting started with Typst

If you're new to Typst...
```
~~~

➡️

Here is some rust code

```rust
fn main() {
}
```

And here is some Typst!

```typ
= Basic Usage

== Getting started with Typst

If you're new to Typst...
```

Functions

2 Basic usage

Typst also has functions, which are denoted by a `#`. Some common functions that you might find useful are:

- `#align()` to change the alignment of the given content.
- `#v()` & `#h()` to add vertical and horizontal spacing
- `#columns()` to break the given content into columns
- `#text()` to change the style of the text
- `#image()` to add images (`.png`, `.jpg`, `.gif`, `.svg`, `.pdf`, `.webp`), and `#figure()` to wrap content and add a caption
- `#table()` and `#grid()` to organize content in grids and tables

There are also the extremely powerful commands `#set` and `#show` which will take a while to understand and are usually used to change the global styling.

`#basic-block` and `#title-block`

In this theme there are also two useful functions for wrapping content in blocks, which are `#basic-block()` for a standard block and `#title-block()` for a block with a title header:

```
#basic-block[This is a simple block]

#title-block(title: "A title")[
  This block has an header with a
title
]
```

➡️

This is a simple block

A title

This block has an header with a title

Theorems and other statements

This theme comes with useful functions for statements such as theorems:

```
#theorem(title: "Euclid's Theorem")[
  There are infinitely many primes.
]

#theorem(number: "5")[
  There are infinitely many primes.
] <thm:euclid>

A reference to @thm:euclid
```

Theorem (Euclid's Theorem)

There are infinitely many primes.

Theorem 5

There are infinitely many primes.

A reference to Theorem 5

There are also: `#definition`, `#lemma`, `#corollary`, `#proposition`, `#axiom`, `#postulate`, `#assumption`, `#property` and `#conjecture`!

#pause and #meanwhile

Of course, you might want to gradually display the content of the slide!

You can use `#pause` to create a slide with only the content preceding the `#pause`.

If there's some content that you want displayed despite the `#pause`, you can put it after a `#meanwhile`:

```
Show this content #pause and then
this #pause and finally this!

#meanwhile
But always show this!
```

➡️

Show this content

But always show this!

#pause and #meanwhile

Of course, you might want to gradually display the content of the slide!

You can use `#pause` to create a slide with only the content preceding the `#pause`.

If there's some content that you want displayed despite the `#pause`, you can put it after a `#meanwhile`:

```
Show this content #pause and then
this #pause and finally this!

#meanwhile
But always show this!
```

➡️

Show this content and then this

But always show this!

#pause and #meanwhile

Of course, you might want to gradually display the content of the slide!

You can use `#pause` to create a slide with only the content preceding the `#pause`.

If there's some content that you want displayed despite the `#pause`, you can put it after a `#meanwhile`:

```
Show this content #pause and then
this #pause and finally this!

#meanwhile
But always show this!
```

➡️

Show this content and then this and finally this!

But always show this!

`#uncover`

If you need to be more precise with your showing and hiding, you can use `#uncover`!

It is useful to show content only on a given interval of "subslides" while always reserving space for it.

```
3… #pause 2… #pause 1… #pause Tadaa!

#meanwhile
Display this #uncover("2-2")[$f(x) =
x^2$] only on drum-roll `2`, and
this #uncover("3-")[$f(x) = x^3$]
after the third, but always reserve
space for them
```

➡️

3…

Display this   only on drum-roll `2`,
and this   after the third, but
always reserve space for them

Note: there are also `#only` and `#alternatives`, but they don't behave nicely with handout mode and I suggest not to use them. If you want you can find more in the documentation.

`#uncover`

If you need to be more precise with your showing and hiding, you can use `#uncover`!

It is useful to show content only on a given interval of “subslides” while always reserving space for it.

```
3… #pause 2… #pause 1… #pause Tadaa!

#meanwhile
Display this #uncover("2-2")[$f(x) =
x^2$] only on drum-roll `2`, and
this #uncover("3-")[$f(x) = x^3$]
after the third, but always reserve
space for them
```

➡️

3… 2…

Display this $f(x) = x^2$ only on drum-roll `2`, and this after the third, but always reserve space for them

Note: there are also `#only` and `#alternatives`, but they don’t behave nicely with handout mode and I suggest not to use them. If you want you can find more in the documentation.

`#uncover`

If you need to be more precise with your showing and hiding, you can use `#uncover`!

It is useful to show content only on a given interval of “subslides” while always reserving space for it.

```
3… #pause 2… #pause 1… #pause Tadaa!

#meanwhile
Display this #uncover("2-2")[$f(x) =
x^2$] only on drum-roll `2`, and
this #uncover("3-")[$f(x) = x^3$]
after the third, but always reserve
space for them
```

➡️

3… 2… 1…

Display this   only on drum-roll `2`, and this $f(x) = x^3$ after the third, but always reserve space for them

Note: there are also `#only` and `#alternatives`, but they don’t behave nicely with handout mode and I suggest not to use them. If you want you can find more in the documentation.

#uncover

If you need to be more precise with your showing and hiding, you can use `#uncover`!

It is useful to show content only on a given interval of “subslides” while always reserving space for it.

```
3… #pause 2… #pause 1… #pause Tadaa!

#meanwhile
Display this #uncover("2-2")[$f(x) =
x^2$] only on drum-roll `2`, and
this #uncover("3-")[$f(x) = x^3$]
after the third, but always reserve
space for them
```

➡️

3… 2… 1… Tadaa!

Display this   only on drum-roll `2`, and this $f(x) = x^3$ after the third, but always reserve space for them

Note: there are also `#only` and `#alternatives`, but they don't behave nicely with handout mode and I suggest not to use them. If you want you can find more in the documentation.

`#slide`, `#title-slide` and more

Some slides such as the title slide have their own function which needs to be called to be created. This is true for:

- `#title-slide()`, which creates the initial slide of the presentation
- `#ending-slide()`, which creates the *“Thank you for listening!”* slide
- `#focus-slide()[your content here]`, similar to an ending slide in appearence but for other type of contents

Additionally, while slides are automatically created for each subsection, you can manually declare a slide with `#slide()[slide content here]`. This is most useful if you want to override some stile only on a single slide, as you will see in the Customization section.

Table of Contents

3 Customization

Language

By default, some parts of these slides appear in English, such as *“Table of Contents”*, *“Theorem”* or the month in the date.

You can change this behaviour by changing the text language with a simple `#set text()` at the top of the `main.typ`. This will change the language setting for all the presentation.

```
#theorem(title: "Euclid's Theorem")[
  There are infinitely many primes.
]
```

➡️

Theorem (Euclid's Theorem)

There are infinitely many primes.

```
#set text(lang: "it")
#theorem(title: "Euclid's Theorem")[
  There are infinitely many primes.
]
```

➡️

Teorema (Euclid's Theorem)

There are infinitely many primes.

Per-slide customizations

Some of the following customization happen by changing some values inside of some `config-xyz()` in the initialization of the theme.

These settings can also be changed on single slides! To do so, you must manually create a slide with `#slide()` and then pass the config with the values you want, such as the following:

```
#slide(config: config-colors(primary: rgb("#006565")))[
  A slide with *updated primary color!*
]
```

If you need to use multiple `config-xyz()`, you can merge them using `utils.merge-dicts()`,

```
#slide(config: utils.merge-dicts(config-colors(/* … */), config-store(/* … */)))[
  A slide with many updated settings!
]
```

Headless statements

By default, statements such as theorems are rendered in a `#title-block` with the theorem title at the top. This might occupy too much space in some cases, and can be disabled by setting `config-store(headless-statements: true)`

```
#show: dmunipi-theme.with(
  // […]
  config-store(
    headless-statements: true,
  ),
)

#theorem(title: "Euclid's Theorem")[
  There are infinitely many primes.
]
```

➡️

Theorem (Euclid’s Theorem): *There are infinitely many primes.*

Fancy footer

By default, the footer of the slides only shows the slide counter. You can change the style of the footer by setting `config-store(fancy-footer: true)` to make it look like the footer in the current slide!

```
#show: dmunipi-theme.with(
  // […]
  config-store(
    fancy-footer: true,
  ),
)
```

Colorscheme

You can even change the colorscheme of the slides! The colorscheme uses:

- the `neutral-lightest` color for the background
- the `neutral-darkest` color for most of the text
- the `primary` color for most of the colored objects
- the `secondary` color for links

```
 #show: dmunipi-theme.with(
  config-colors(
    primary: rgb("#006565"),
    secondary: rgb("#00d595"),
  ),
)
```

Advanced customization

These were the type of customization that come “pre-configured” in this theme, and are accessible through existing variables.

Typst actually allows you to customize the styling of the document much more in depth, thanks to the commands `#set` and `#show`

Explaining the usage of these commands is above the purpose of this guide. If you’re interested, you will find everything you need to start in the documentation.

The Typst `dmunipi` theme

Thank you for listening!
Any question?

Table of Contents

4 Troubleshooting

Font issues

By default, this theme uses the `Roboto` font family.

If the font in your pdf looks like this instead of how it appears in the rest of this guide, then it’s likely that you don’t have the `Roboto` font installed on your system

You can either choose to download it and install it, or edit the `theme/lib.typ` file to change the default font to something installed on your system of your liking