

東京大学大学院 工学系研究科 航空宇宙工学専攻
修士論文

Typst で書く修論のテンプレ

12-345678 右往 左往

2024 年 1 月 2 日 提出

目次

1	序論	p. 1
1.1	Typst は優秀だ	p. 1
1.1.1	エレガントに書ける	p. 1
2	先行研究	p. 2
2.1.1	LATEX はコンパイルが遅い	p. 2
3	定義	p. 3
3.1	定義例	p. 3
	参考文献	p. 5

第 1 章

序論

Typst は markdown like なコーディングで pdf, ポスター, スライド等のドキュメントを作成できます. Rust 言語で書かれており, コンパイルが L^AT_EX に比べて早いのが特長です.

1.1 Typst は優秀だ

こんな感じで @ss8843592 or #cite(<ss8843592>) と引用できます

こんな感じで [1] or [1] と引用できます

1.1.1 エレガントに書ける 数式

```
$ \mat(1, 2; 3, 4) $ <eq1>
```

と書くと

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \tag{1.1}$$

式 (1.1) を書くことができます.

関数を作れば

typst

図 1.1 イメージ

図 1.1 を表示できますし,

表 1.1 テーブル [2]

t	1	2	3
y	0.3s	0.4s	0.8s

表 1.1 も表示できます.

第 2 章

先行研究

L^AT_EX はカスタム性の高さ, 歴史的なところからまだまだ廃れないと思いますが, 卒論や修論や学会の予稿等の作成においては Typst [2] の使いやすさから置き換わるのではないかと思います(半分願望).



図 2.1 Typst + git [2]

2.1.1 L^AT_EX はコンパイルが遅い

本資料は, L^AT_EX でコンパイルの待ち時間中に作りました. 他にも

`#include path.typ`

とすれば, 他ファイルを参照できるので, 長い分量の本などを作成する際に, 章ごとにファイルを分けるなどができるようになります.

便利なので広まれば良いと思います.

詳しくは 公式ドキュメント をご覧ください

第3章

定義

Typst では関数定義が簡単であるので定理の書き方などをカスタマイズできます.

3.1 定義例

thmbox 関数を作ってカスタマイズをできるようにしました.

```
#let theorem = thmbox(  
  "theorem", //identifier  
  "定理",  
  base_level: 1  
)
```

```
#theorem("ライラ-") [  
  Typst はすごいのである.  
> ] <theorem>
```

定理 3.1 (ライラ-): Typst はすごいのである.

```
#let lemma = thmbox(  
  "theorem", //identifier  
  "補題",  
  base_level: 1,  
)
```

```
#lemma [  
  Tex はさようならである.  
> ] <lemma>
```

補題 3.2: Tex はさようならである.

このように, 定理 3.1, 補題 3.2 を定義できます.

カッコ内の引数に人名などを入れることができます. また, identifier を変えれば, カウントはリセットされる. identifier 毎にカウントを柔軟に変えられるようにしてあるので, 様々な論文の形式に対応できるはずです.

```
#let definition = thmbox(  
  "definition", //identifier  
  "定義",
```

```
base_level: 1,
stroke: black + 1pt
)
#definition("Prime numbers")[
  A natural number is called a prime number if it is greater than $1$ and
  cannot be written as the product of two smaller natural numbers.
] <definition>
```

定義 3.1: Typst is a new markup-based typesetting system for the sciences.

定義 3.1 のようにカウントがリセットされています.

```
#let corollary = thmbox(
  "corollary",
  "Corollary",
  base: "theorem",
)

#corollary[
  If $n$ divides two consecutive natural numbers, then $n = 1$.
] <corollary>
```

Corollary 3.2.1: If n divides two consecutive natural numbers, then $n = 1$.

base に identifier を入れることで Corollary 3.2.1 のようにサブカウントを実現できます.

```
#let example = thmplain(
  "example",
  "Example"
).with(numbering: none)

#example[
  数式は\$\$ で囲む
] <example>
```

例: 数式は \$\$ で囲む

thmplain 関数を使って plain 表現も可能です.

参考文献

- [1] S. Hussain, S. Bai, and S. Khoja, “Content MathML(CMML) conversion using LATEX Math Grammar (LMG)”, in *2019 7th International Conference on Smart Computing & Communications (ICSCC)*, 2019, pp. 1–5. doi: 10.1109/ICSCC.2019.8843592.
- [2] L. Mädje, “A Programmable Markup Language for Typesetting”, 2022.