Typst Cheat Sheet

Using Touying

by Darstib

2025-02-23

Outline

Outline

1. Syntax

1.1 Modes

New mode	Syntax	Example
Code	Prefix the code with `#`	Number: `#(1 + 2)`
Math	Surround equation with `$..$`	`$-x$` is the opposite of `$x$`
Markup	Surround markup with `[..]`	`#let name = [*Typst!*]`

1.2 Markup

Typst is a markup language. This means that you can use simple syntax to accomplish common layout tasks.

- symbol
  1. **strong**
  2. *emphasis*
  3. `raw text`
  4. `<label>` and `@reference`
  5.

- function
  1. **strong**
  2. *emphasis*
  3. `fn main();`
  4. Link to raw usage
  5. ~~overline~~
  6. underline
  7. highlight
  8. ~~strike~~
  9. [1]

[1] this is a footnote

1.3 Math

A in line function: $a^2 + b^2 = c^2$

A block function:

$$e^{i\pi} + 1 = 0$$

more complex:

$$\mathrm{M} := (1 \;\; 2 \;\; 3 \;\; 4 \;\; 5 \;\; 6 \;\; 7 \;\; 8 \;\; 9)$$

More symbol

1.4 Code

Data type:

- Length: 1em, 2pt, 3mm
- Angle: 1deg, 2rad, 3grad
- Fraction: 1fr
- Ratio: 50%
- Array: (1, 2, 3)
- Dictionary: (a: 1, b: 2, c: “hi”)
- Content block: `[content]`
- Code block: { let x =1; x+1}

Usage:

- Field access: field.name
- Function call: function()
- Arg spreading: function(..args)
- Method call: field.method()
- Unnamed func: `(a,b)=>a+b`
- Named: `let add(a,b)=a+b`
- Let blinding: `#let a = 1`
- `#set` and `#show`
- `# context`

2. Get deeper

2.1 Scripting

2.1.1 Field

```
#let it = [== subtitle]
#let dict = (greet: "Hello")
#it.fields()
#dict.greet
#emoji.face
```

```
(depth: 2, body: [subtitle])
```

Hello

😀

2.1 Scripting

2.1.2 Method

Just like in python

```
#let demo_str = "Hello,
typst!"
#demo_str
#demo_str.len()
#str.len(demo_str)
```

Hello, typst!

13

13

2.1 Scripting

2.1.3 Flow

Loop

```
#let n = 2
#while n < 10 {
    n = (n * 2) - 1
    (n,)
}

#let s = "Hello, typst!"
#for char in s [
    (#char,)
]
```

Just like in python

Condition

```
#if 1 < 2 [
  This is shown
] else [
  This is not.
]
```

2.3 Selector

2.3.1 constructor

- **func element**: `heading figure`
- **special field**:
  `self.where(..any)`
- **regex**: `^[a-z]+`
- **location**: `#here().page()`
- **<lable>**

2.3.2 definition

- **self.or**(selector)
- **self.and**(selector)
- **self.before**(selector)
- **self.after**(selector)

2.4 Function

2.4.1 define

Using `#let` blinding to define a function, with a code block as the body.

Warning:
This is a warning message.

```
#let alert(body, fill: red,
inset: 8pt, radius: 4pt) = {
  set text(white)
  set align(center)
  rect(
    fill: fill,
    inset: inset,
    radius: radius,
    [*Warning:\ #body*],
  )
}
```

2.4 Function

2.4.2 Import

Functions can be imported from one file (module) into another using import. For example, assume that we have defined the alert function from the previous example in a file called foo.typ. We can import it into another file by writing `import 'foo.typ': alert.`

2.5 Figure

2.5.1 image

Figure 1: Kamisato Ayaka

The Figure 1 is a demo picture.

2.5 Figure

2.5.2 table

Table 1: Timing results

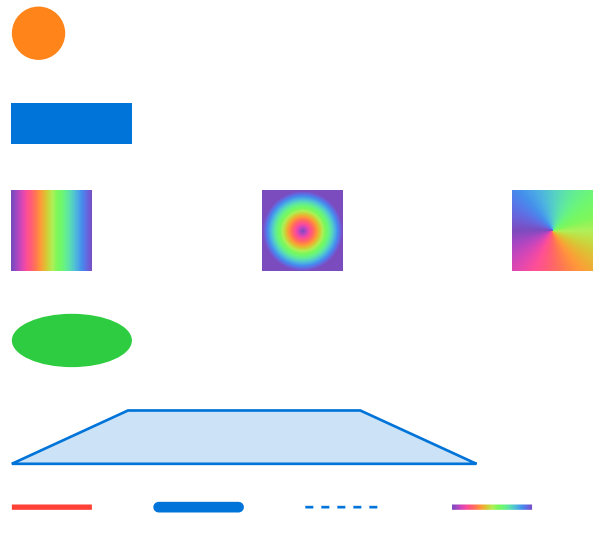
	Volume	Parameters
function1	$\pi h \frac{D^2 - d^2}{4}$	h: height D: outer radius d: inner radius
function2	$\frac{\sqrt{2}}{12} a^3$	a: edge length

The Table 1 is a demo table.

2.6 visualize

2.6.1 basic graph

```
- #circle(radius: 10pt, fill: orange)
- #rect(height: 20pt, fill: blue)
- #square with gradient
- #ellipse(height: 20pt, fill: green)
- #polygon
- #line
```



2.7 Other

2.7.1 Sinks & Spreading

We can specify an argument sink which collects all excess arguments as `..args` and just spread it with `..args`

ArtosFlow

Written by Jane, Joe and Jake 9

```
#let format(title, ..authors)
= {
  let by = authors.pos()
    .join(", ", last: " and
")
  [*#title* \ _Written by
#by;_]
}
#format("ArtosFlow", "Jane",
"Joe", "Jake")
#let arr = (1, 3, 5, 7, 9)
#calc.min(..arr)
```

2.7 Other

2.7.2 Regex

```
// Works with string methods.
#"a,b;c".split(regex("[,;]"))   ("a", "b", "c")

// Works with show rules.
#show regex("\d+"): set
text(red)

The numbers 1 to 10.
```

The numbers 1 to 10.

2.7 Other

2.7.3 Quote

> ... ἔοικα γοῦν τούτου γε σμικρῷ τινι αὐτῷ τούτῳ σοφώτερος εἶναι, ὅτι ἃ μὴ οἶδα οὐδὲ οἴομαι εἰδέναι.
>
> — Plato

3. Animation

3.1 Simple Animation

We can use `#pause` to

Meanwhile,

3.1 Simple Animation

We can use `#pause` to display something later.

Meanwhile, we can also use `#meanwhile` to

3.1 Simple Animation

We can use `#pause` to display something later. Just like this.

Meanwhile, we can also use `#meanwhile` to display other content synchronously.

3.2 Complex Animation

At subslide 1, we can

use for reserving space,

use for not reserving space,

call `#only` multiple times ✕ for choosing one of the alternatives.

3.2 Complex Animation

At subslide 2, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

3.3 Callback Style

At subslide 1, we can

use for reserving space,

use for not reserving space,

call `#only` multiple times ✕ for choosing one of the alternatives.

3.3 Callback Style

At subslide 2, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

3.3 Callback Style

At subslide 3, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

3.4 Math Equation

Equation with pause:

$$f(x) =$$

Here,

3.4 Math Equation

Equation with pause:

$$f(x) = x^2 + 2x + 1$$

$$=$$

Here, we have the expression of $f(x)$.

3.4 Math Equation

Equation with pause:

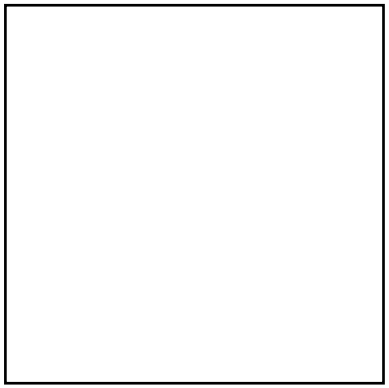
$$f(x) = x^2 + 2x + 1$$
$$= (x + 1)^2$$

Here, we have the expression of $f(x)$.

By factorizing, we can obtain this result.

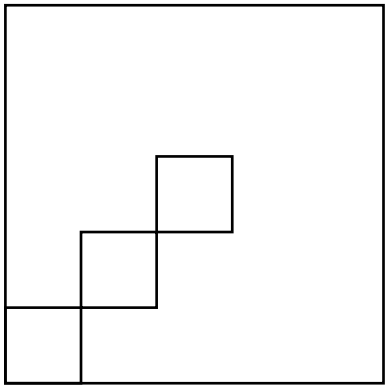
3.5 CeTZ Animation

CeTZ Animation in Touying:



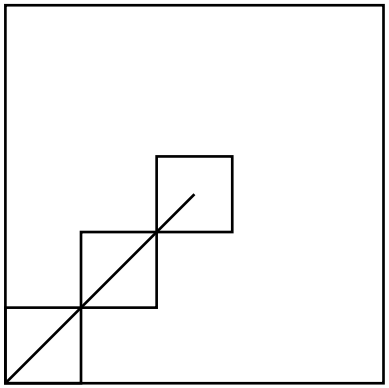
3.5 CeTZ Animation

CeTZ Animation in Touying:



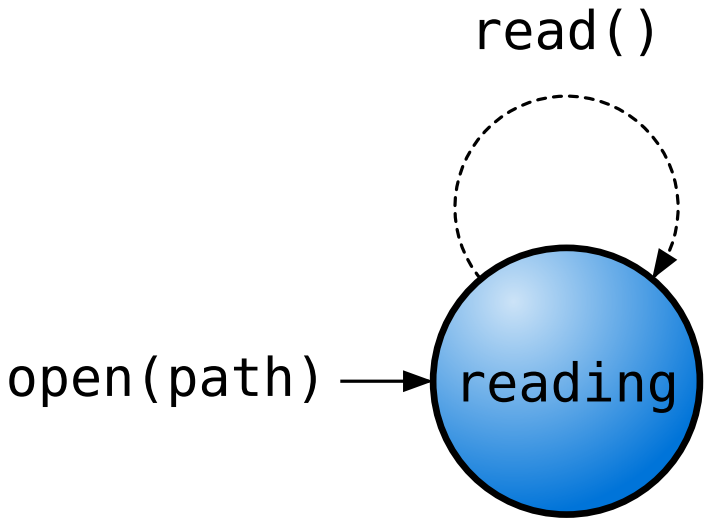
3.5 CeTZ Animation

CeTZ Animation in Touying:



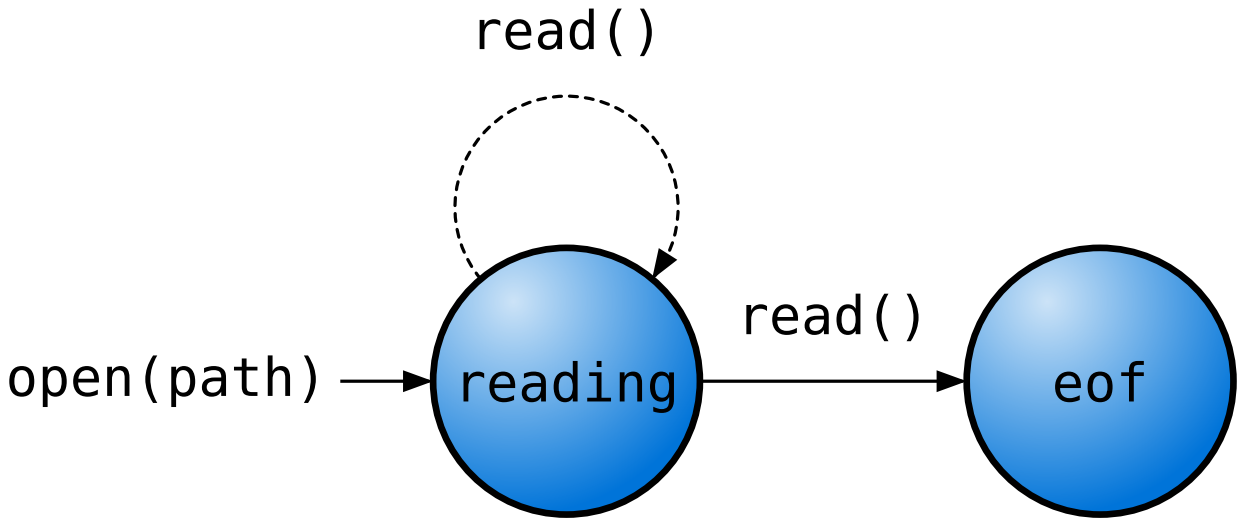
3.6 Fletcher Animation

Fletcher Animation in Touying:



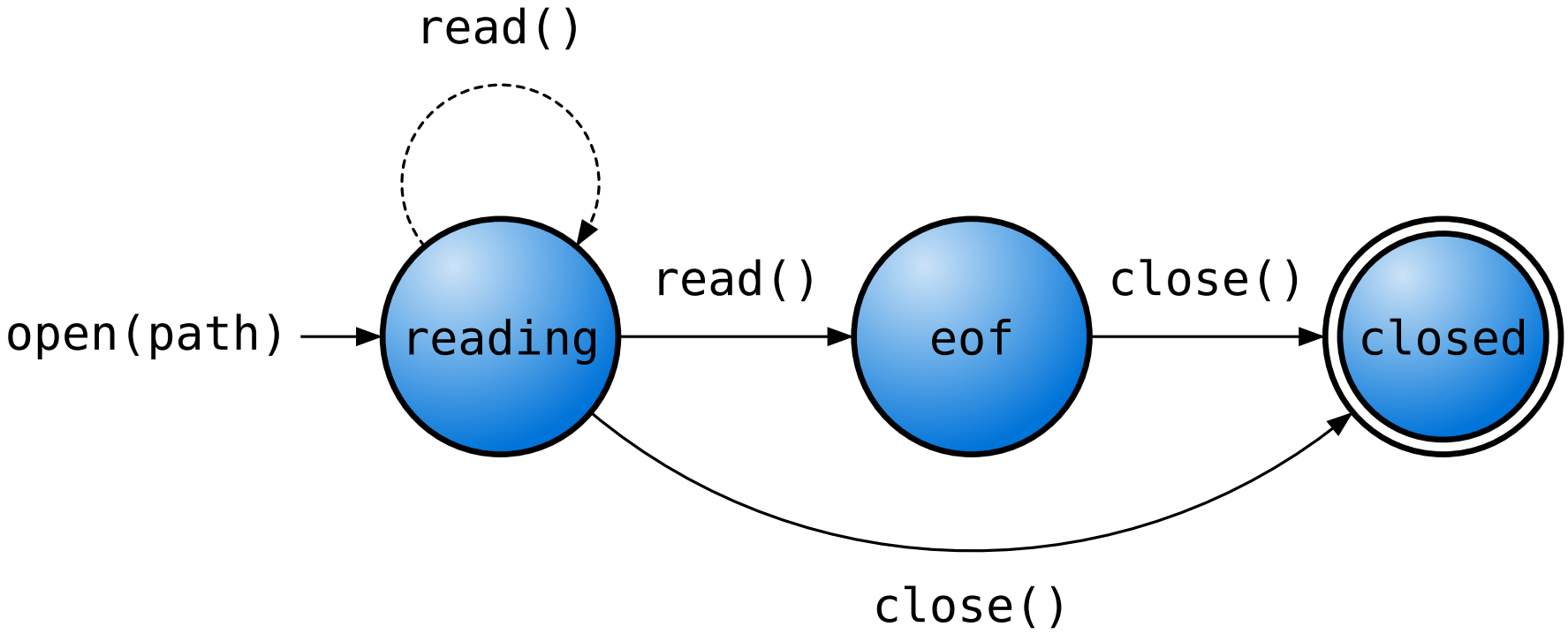
3.6 Fletcher Animation

Fletcher Animation in Touying:



3.6 Fletcher Animation

Fletcher Animation in Touying:



4. Theorems

4.1 Prime num

Definition 4.1.1: 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.

Example: The numbers 2, 3, and 17 are prime. Corollary 4.1.1.1 shows that this list is not exhaustive!

Theorem 4.1.1 (Euclid): There are infinitely many primes.

4.1 Prime num

Proof: Suppose to the contrary that $p_1, p_2, \ldots, p_n$ is a finite enumeration of all primes. Set $P = p_1 p_2 \ldots p_n$. Since $P + 1$ is not in our list, it cannot be prime. Thus, some prime factor p_j divides $P + 1$. Since p_j also divides P, it must divide the difference $(P + 1) - P = 1$, a contradiction. □

Corollary 4.1.1.1: There is no largest prime number.

Corollary 4.1.1.2: There are infinitely many composite numbers.

4.1 Prime num

Theorem 4.1.2: There are arbitrarily long stretches of composite numbers.

Proof: For any $n > 2$, consider

$$n! + 2, \quad n! + 3, \quad \ldots, \quad n! + n \qquad \square$$

5. Others

5.1 Side-by-side

First column.

Second column.

5.2 Multiple Pages

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque

5.2 Multiple Pages

earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

6. Appendix

6.1 Appendix

Please pay attention to the current slide number.