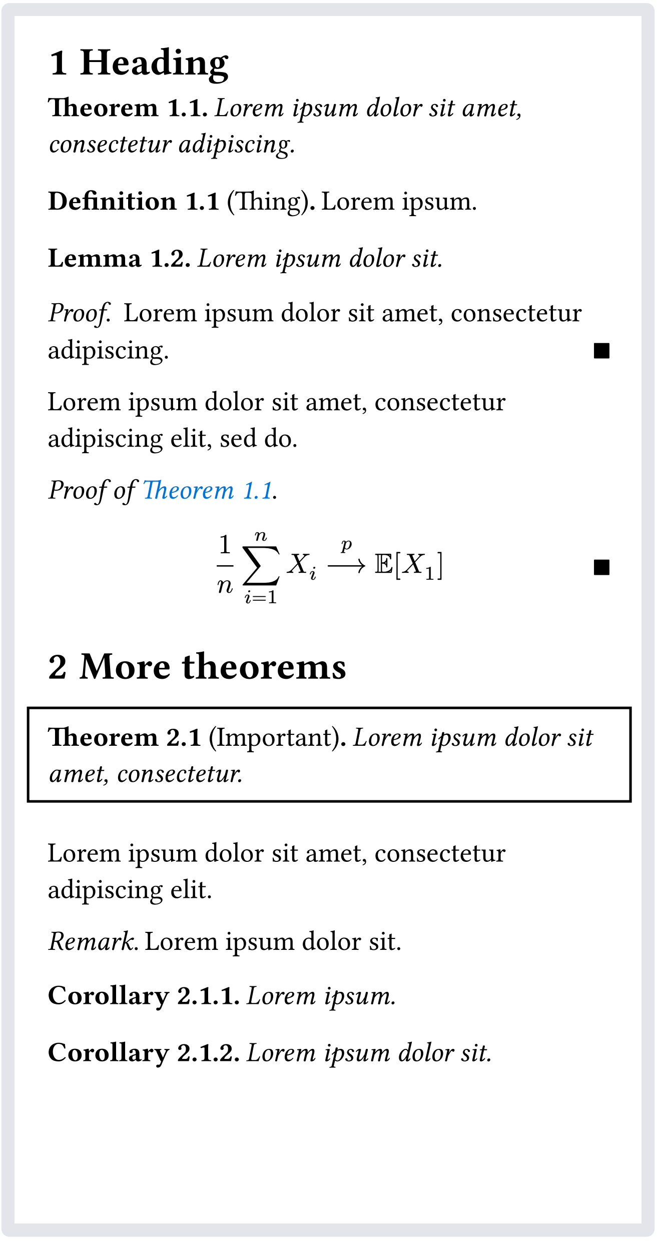
typst-theorems/ctheorems documentation

thm-rules

Rules for styling theorem environments, references, proofs, etc. Must appear at the beginning of the document.

```
#show: thm-rules
#set heading(numbering: "1.1")

#let theorem = thm-plain("Theorem")
#let lemma = thm-plain(
  "Lemma",
  counter: "Theorem",
)
#let corollary = thm-plain(
  "Corollary",
  base: "Theorem"
)
#let definition = thm-def("Definition")
#let remark = thm-rem("Remark")
#let proof = thm-proof("Proof")

= Heading

#theorem[#lorem(7)] <mythm>
#definition("Thing")[#lorem(2)]
#lemma[#lorem(4)]
#proof[
  #lorem(7)
]
#lorem(10)
#proof([of @mythm])[
  $
    1/n sum_(i = 1)^n X_i ⟶^p EE[X_1] #qedhere
  $
]

= More theorems

#let theorem-standout = theorem.with(
  stroke: 1pt,
  outset: 0.7em,
  padding: (y: 1em)
)
#theorem-standout("Important")[#lorem(6)]
#lorem(8)
#remark[#lorem(4)]
#corollary[#lorem(2)]
#corollary[#lorem(4)]
```



1 Heading

Theorem 1.1. *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Definition 1.1 (Thing)**.** Lorem ipsum.

Lemma 1.2. *Lorem ipsum dolor sit.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

Proof of Theorem 1.1.

$$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{p} \mathbb{E}[X_1] \qquad ■$$

2 More theorems

Theorem 2.1 (Important)**.** *Lorem ipsum dolor sit amet, consectetur.*

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Remark. Lorem ipsum dolor sit.

Corollary 2.1.1. *Lorem ipsum.*

Corollary 2.1.2. *Lorem ipsum dolor sit.*

Parameters

```
thm-rules(
  qed-symbol: content,
  doc
)
```

qed-symbol `content`

Symbol displayed at the end of proofs. See `thm-proof`, `qedhere`, `proof-body-fmt()`. Use as

```
#show: thm-rules.with(
  qed-symbol: $square$
)

#let proof = thm-proof("Proof")

#proof[#lorem(3)]
#proof[
  #lorem(5)
  $ integral_0^oo sin(x)/x = pi/2. #qedhere $
]
```

Proof. Lorem ipsum dolor. $\square$

Proof. Lorem ipsum dolor sit amet.

$$\int_0^\infty \frac{\sin(x)}{x} = \frac{\pi}{2}. \qquad \square$$

Default: `$qed$`

thm-env

Creates a theorem environment, which is a function of the form

```
(
  ..thm-args,
  body,
  number: auto,
  numbering: "1.1",
  base: base,
  base-level: base-level,
  restate: false,
  defer: false,
  restate-keys: (counter, ),
  supplement: counter,
  ref-fmt: (supplement, thm) => {
    if supplement != none { supplement = [#supplement~] }
    link(thm.loc, [#supplement#(thm.number)])
  },
) -> content
```

The `body` contains the content of the theorem environment, and `thm-args` get passed to the formatting function `fmt`. The first positional argument from `thm-args` is interpreted as the `name` of the theorem environment.

The `numbering` option specifies the numbering used for the theorem environment (set to `none` for turning numbering off). Setting the `number` option lets you override the automatic numbering with content.

The `base` and `base-level` options are inherited from the `thm-env` call; see the list of parameters below.

The `supplement` determines the default supplement used when a labeled theorem environment is referenced. The `ref-fmt` lets you specify custom formatting for references; see `thm-display()` for more details on the `thm` dictionary.

See `thm-restate()` for more information about the `restate`, `defer`, and `restate-keys` options.

```
#show: thm-rules
#set heading(numbering: "1.1")

#let theorem = thm-env(
  "Theorem",
  (name, number, body, color: black) ⇒ {
    if name ≠ none { name = [~(#name)] }
    text(color)[
      *Theorem~#number*#name:~#body\
    ]
  },
  base: "heading"
)

= First heading
#theorem[#lorem(5)]
#theorem("Named")[#lorem(7)]

Refer to @thm.

== First Subheading
#theorem[#lorem(3)]
#theorem[#lorem(4)]

== Second Subheading
#theorem[#lorem(6)]
#theorem(color: red)[#lorem(2)] <thm>

= Second heading
#theorem[#lorem(4)]
#theorem(number: $dagger$)[#lorem(9)]
#theorem[#lorem(7)]
```

1 First heading

Theorem 1.1: Lorem ipsum dolor sit amet.
Theorem 1.2 (Named): Lorem ipsum dolor sit amet, consectetur adipiscing.

Refer to Theorem 1.2.2.

1.1 First Subheading

Theorem 1.1.1: Lorem ipsum dolor.
Theorem 1.1.2: Lorem ipsum dolor sit.

1.2 Second Subheading

Theorem 1.2.1: Lorem ipsum dolor sit amet, consectetur.
Theorem 1.2.2: Lorem ipsum.

2 Second heading

Theorem 2.1: Lorem ipsum dolor sit.
Theorem †: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed.
Theorem 2.2: Lorem ipsum dolor sit amet, consectetur adipiscing.

Parameters

```
thm-env(
  counter: string,
  fmt: function,
  base: string,
  base-level: integer
) -> function
```

counter `string`

Environment counter name.

fmt `function`

Formatting function, of the form `(name, number, body, ..fmt-args) → content`. When a theorem environment is called, the named arguments from `thm-args` are passed into `fmt-args`.

base `string`

Base counter name, whose numbering prefixes the theorem environment numbering. If `none`, the theorem environment maintains a global count with no prefix.

Default: `none`

base-level `integer`

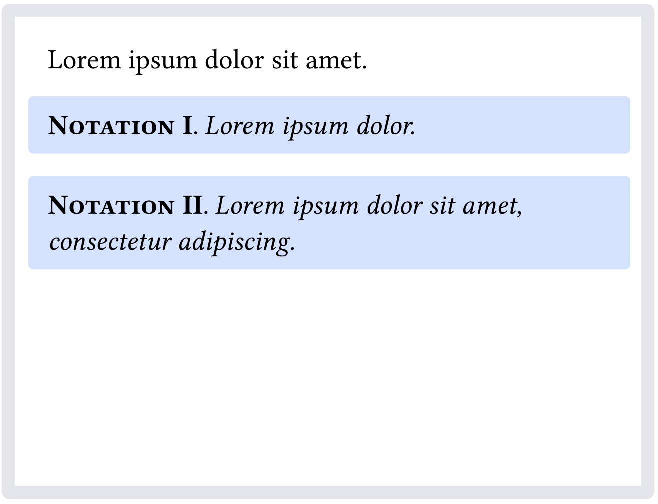
Base level, determining the number of levels of the `base` numbering to use during the theorem environment numbering. If `none`, all levels from the `base` numbering are used.

Default: `none`

thm-box

Creates a theorem environment wrapped in a padded block, with sensible default styling. The block has `width: 100%` applied by default. The `fmt` function is of the form `(name, number, body, title: auto, ..fmt-args) → content`. All named arguments from `args`, followed by all named `fmt-args`, are passed to the `block` call.

```
#show: thm-rules

#let notation = thm-box(
  "Notation",
  base: none,
  numbering: "I",
  title-fmt: t => smallcaps(strong(t)),
  body-fmt: emph,
  outset: 0.7em,
  padding: (y: 0.5em),
  radius: 2pt,
  fill: rgb("#d4e2fe"),
)

#lorem(5)
#notation[#lorem(3)]
#notation[#lorem(7)]
```



Lorem ipsum dolor sit amet.

Notation I. *Lorem ipsum dolor.*

Notation II. *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Parameters

```
thm-box(
  head: content,
  counter: string,
  ..args,
  numbering: string function,
  supplement: string,
  padding: dictionary,
  name-fmt: function,
  title-fmt: function,
  body-fmt: function,
  separator: content,
  base: string,
  base-level: integer
) -> function
```

head `content`

Environment heading.

counter `string`

Environment counter name. If `auto`, set to `head`.

Default: `auto`

numbering `string` or `function`

Environment numbering style.

Default: `"1.1"`

supplement `string`

Supplement for references. If `auto`, set to `head`.

Default: `auto`

padding `dictionary`

Padding around the block.

Default: `(y: 0.1em)`

name-fmt `function`

Formatting for the environment name.

Default: `x => [(#x)]`

title-fmt `function`

Formatting for the environment title (head and number).

Default: `x => x`

body-fmt `function`

Formatting for the environment body.

Default: `x => x`

separator `content`

Separator between title and body.

Default: `[.#h(0.2em)]`

base `string`

Base counter name.

Default: `"heading"`

base-level `integer`

Base level.

Default: `none`

proof-body-fmt

Used as the `body-fmt` in `thm-proof`, for properly styling proofs by inserting a `qed` symbol at the end of the body. Also see `qedhere`.

```
#show: thm-rules.with(qed-symbol: "Q.E.D.")

#proof-body-fmt[#lorem(3)]
#v(2em)

#proof-body-fmt[
  $
    phi.alt(x) = 1/sqrt(2 pi) e^(-x^2\/2) #qedhere
  $
]
```

Lorem ipsum dolor. Q.E.D.

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \qquad \text{Q.E.D.}$$

Parameters

```
proof-body-fmt(body: content) -> content
```

body `content`

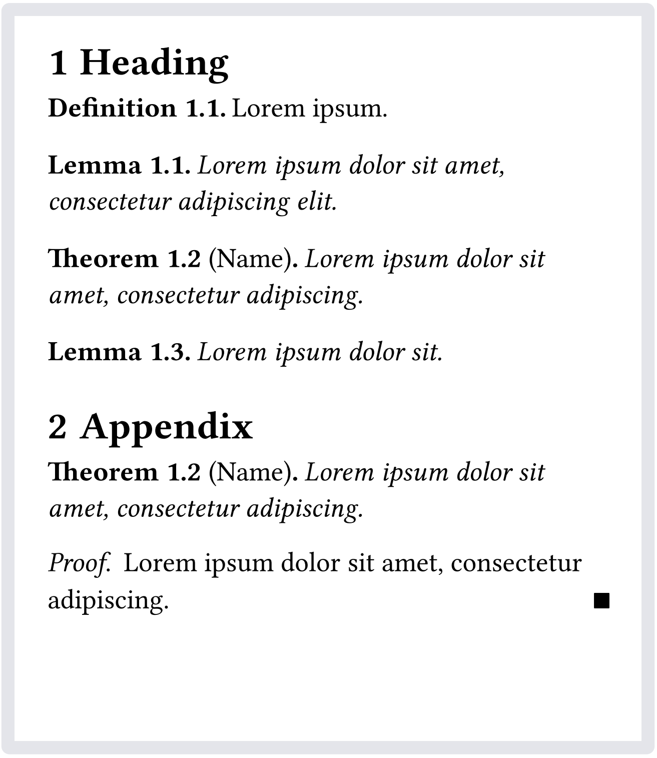
Proof body.

thm-restate

Displays theorem environments which have been marked to be restated or deferred, can be filtered. Useful for pushing content to the appendix. See `thm-display()` for the structure of a `thm`.

The following example illustrates the basic usage of `thm-restate`, combined with the `restate` and `defer` flags for theorem environments.

```
#show: thm-rules
#set heading(numbering: "1.1")

#let theorem = thm-plain("Theorem")
#let lemma = thm-plain(
  "Lemma",
  counter: "Theorem",
)
#let definition = thm-def("Definition")
#let proof = thm-proof("Proof")

= Heading

#definition[#lorem(2)]
#lemma[#lorem(8)]

#theorem("Name", restate: true)[#lorem(7)]
#proof(defer: true)[
  #lorem(7)
]

#lemma[#lorem(4)]

= Appendix

#thm-restate()
```



1 Heading

Definition 1.1. Lorem ipsum.

Lemma 1.1. *Lorem ipsum dolor sit amet, consectetur adipiscing elit.*

Theorem 1.2 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Lemma 1.3. *Lorem ipsum dolor sit.*

2 Appendix

Theorem 1.2 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

Parameters

```
thm-restate(
  ..keys: string array function ,
  fmt: function ,
  at: label selector location function ,
  final: boolean
) -> content
```

..keys `string` or `array` or `function`

String keys, array of keys, or functions used to filter theorem environments. A `thm` is displayed if it passes *any* of the filters.

If `k` in `keys` is a `string`, theorem environments containing `k` in its array of `restate-keys` will be matched.

```
#show: thm-rules
#set heading(numbering: "1.1")

#let theorem = thm-plain("Theorem")
#let lemma = thm-plain(
  "Lemma",
  counter: "Theorem",
)
#let proof = thm-proof("Proof")

= Heading

#theorem(restate: true)[#lorem(6)]
#lemma(restate: true)[#lorem(4)]
#proof(defer: true)[#lorem(7)]
#lemma(restate: true)[#lorem(3)]

= Restate lemmas/proofs
#thm-restate("Lemma", "Proof")
```

1 Heading

Theorem 1.1. *Lorem ipsum dolor sit amet, consectetur.*

Lemma 1.2. *Lorem ipsum dolor sit.*

Lemma 1.3. *Lorem ipsum dolor.*

2 Restate lemmas/proofs

Lemma 1.2. *Lorem ipsum dolor sit.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

Lemma 1.3. *Lorem ipsum dolor.*

If `k` in `keys` is an array of `strings`, theorem environments containing *all* keys from `k` in its array of `restate-keys` will be matched.

```
#show: thm-rules
#set heading(numbering: "1.1")

= Heading

#theorem(
  "Result A",
  restate: true,
  restate-keys: ("Theorem", "Result A")
)[#lorem(6)]
#proof(
  defer: true,
  restate-keys: ("Proof", "Result A")
)[#lorem(7)]
#theorem(restate: true)[#lorem(6)]
#theorem(
  "Result B",
  restate: true,
  restate-keys: ("Theorem", "Result B")
)[#lorem(6)]
#proof(
  defer: true,
  restate-keys: ("Proof", "Result B")
)[#lorem(7)]

= Restate Result A
#thm-restate("Result A")

= Restate theorems tagged Result B
#thm-restate(("Theorem", "Result B"))
```

3 Heading

Theorem 3.1 (Result A)**.** *Lorem ipsum dolor sit amet, consectetur.*

Theorem 3.2. *Lorem ipsum dolor sit amet, consectetur.*

Theorem 3.3 (Result B)**.** *Lorem ipsum dolor sit amet, consectetur.*

4 Restate Result A

Theorem 3.1 (Result A)**.** *Lorem ipsum dolor sit amet, consectetur.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

5 Restate theorems tagged Result B

Theorem 3.3 (Result B)**.** *Lorem ipsum dolor sit amet, consectetur.*

If `k` in `keys` is a `function`, it must be of the form `restate-keys → boolean`.

```
#show: thm-rules
#set heading(numbering: "1.1")

= Heading

#theorem(
  restate: true,
  restate-keys: (
    "Theorem", "Unproven claim"
  )
)[#lorem(6)]
#theorem(restate: true)[#lorem(6)]
#lemma(
  "Claim D",
  restate: true,
  restate-keys: ("Lemma", "Claim D")
)[#lorem(6)]

= Restate claims
#thm-restate(
  keys ⇒ keys.any(
    k ⇒ lower(k).contains("claim")
  )
)
```

6 Heading

Theorem 6.1. *Lorem ipsum dolor sit amet, consectetur.*

Theorem 6.2. *Lorem ipsum dolor sit amet, consectetur.*

Lemma 6.3 (Claim D)**.** *Lorem ipsum dolor sit amet, consectetur.*

7 Restate claims

Theorem 6.1. *Lorem ipsum dolor sit amet, consectetur.*

Lemma 6.3 (Claim D)**.** *Lorem ipsum dolor sit amet, consectetur.*

fmt `function`

Formatting function of the form `thm → content`. The default `auto` uses the same `fmt` originally supplied to the `thm-env`. See corresponding option in `thm-display()`.

Default: `auto`

at `label` or `selector` or `location` or `function`

Location up to which theorem environments will be displayed. The default `auto` uses the location where `thm-restate` was called. See corresponding option in `thm-display()`.

Default: `auto`

final `boolean`

If `true`, display environments up to the end of the document. See corresponding option in `thm-display()`.

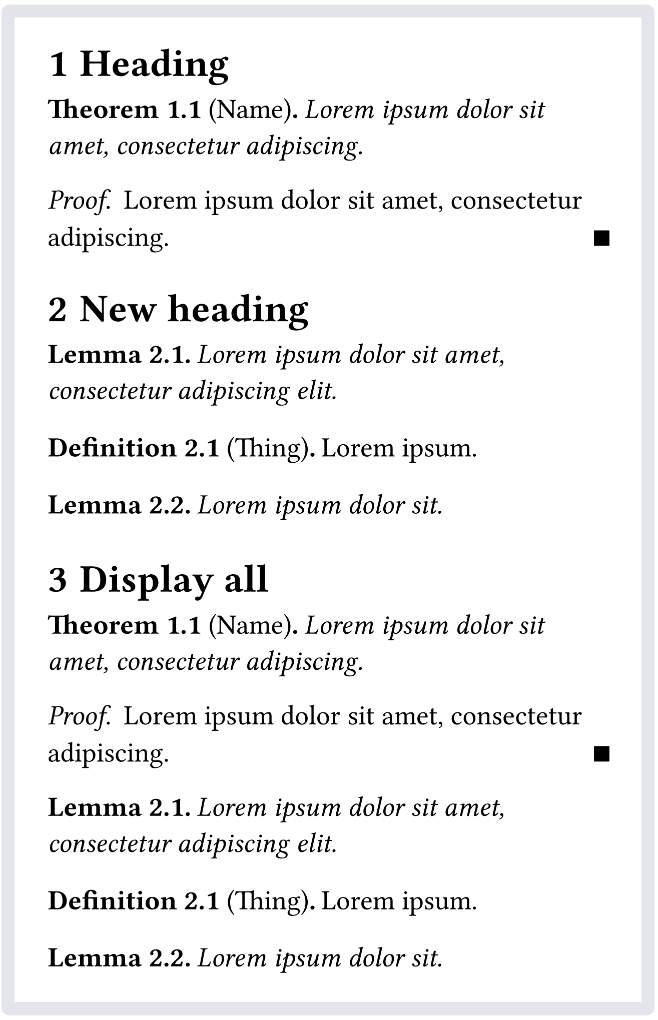
Default: `false`

thm-display

Displays all theorem environments, can be filtered. A `thm` is a dictionary storing information about a theorem environment, with keys

```
(
  args,
  name,
  body,
  supplement,
  fmt,
  number,
  numbering,
  restate,
  defer,
  restate-keys,
  ref-fmt,
  loc,
  counter,
  base,
  base-level
)
```

```
#show: thm-rules
#set heading(numbering: "1.1")

#let theorem = thm-plain("Theorem")
#let lemma = thm-plain(
  "Lemma",
  counter: "Theorem",
)
#let definition = thm-def("Definition")
#let proof = thm-proof("Proof")

= Heading <h1>

#theorem("Name")[#lorem(7)]
#proof[
  #lorem(7)
]

= New heading <h2>

#lemma[#lorem(8)]
#definition("Thing")[#lorem(2)]
#lemma[#lorem(4)]

= Display all

#thm-display()
```



1 Heading

Theorem 1.1 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

2 New heading

Lemma 2.1. *Lorem ipsum dolor sit amet, consectetur adipiscing elit.*

Definition 2.1 (Thing). Lorem ipsum.

Lemma 2.2. *Lorem ipsum dolor sit.*

3 Display all

Theorem 1.1 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

Lemma 2.1. *Lorem ipsum dolor sit amet, consectetur adipiscing elit.*

Definition 2.1 (Thing). Lorem ipsum.

Lemma 2.2. *Lorem ipsum dolor sit.*

The key `loc` gives the location of the theorem environment in the document. The `number` gives the (calculated and formatted) number of the theorem environment. The remaining keys contain information as detailed in `thm-env()`.

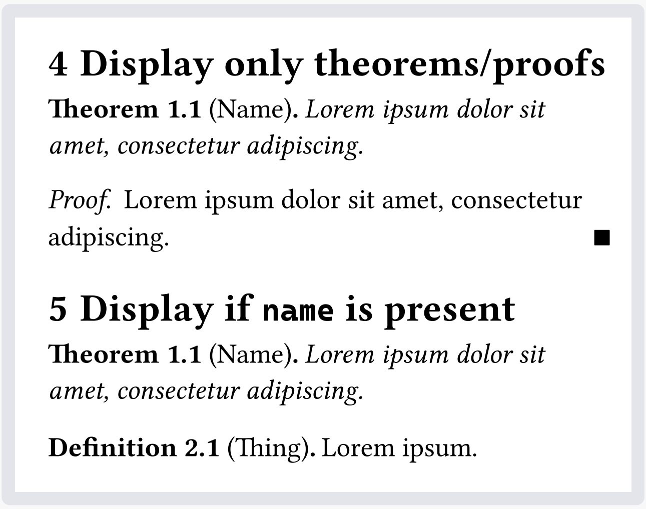
Parameters

```
thm-display(
  ..filters: function,
  fmt: function,
  at: label selector location function,
  final: boolean
) -> content
```

..filters `function`

Filtering functions. Each `f` in `filters` is a function `thm → boolean`. A `thm` is displayed if it passes *any* of the filters.

```
#show: thm-rules
#set heading(numbering: "1.1")

= Display only theorems/proofs

#thm-display(
  thm => thm.supplement == "Theorem",
  thm => thm.supplement == "Proof",
)

= Display if `name` is present

#thm-display(
  thm => thm.name != none
)
```



4 Display only theorems/proofs

Theorem 1.1 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

5 Display if `name` is present

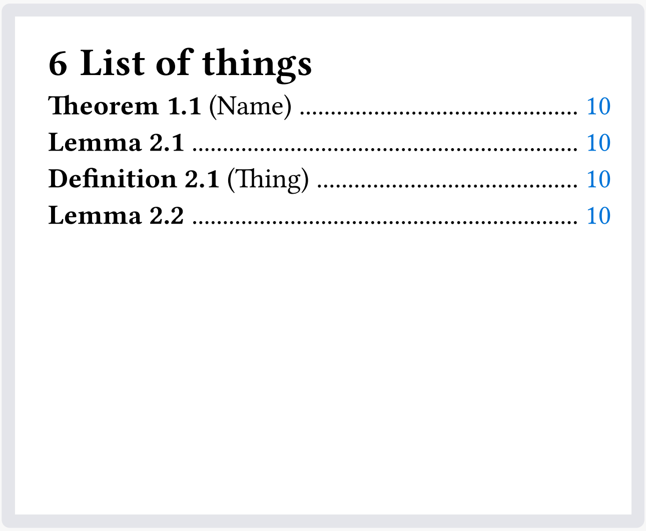
Theorem 1.1 (Name). *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Definition 2.1 (Thing). Lorem ipsum.

fmt `function`

Formatting function of the form `thm → content`. The default `auto` uses the same `fmt` originally supplied to the `thm-env`.

```
#show: thm-rules
#set heading(numbering: "1.1")

= List of things

#thm-display(
  thm => thm.supplement != "Proof",
  final: true,
  fmt: thm => {
    let head = [*#thm.supplement~#thm.number*]
    if thm.name != none {
      head = head + [~(#thm.name)]
    }
    let page = thm.loc.position().page
    let page = link(thm.loc, [#page])
    [#head~#box(width: 1fr, repeat[.])~#page\ ]
  }
)
```



6 List of things

Theorem 1.1 (Name) 10
Lemma 2.1 10
Definition 2.1 (Thing) 10
Lemma 2.2 10

The `final: true` ensures that even if this `thm-display` call is placed at the beginning of the document, all theorem environments are listed.

Default: `auto`

at `label` or `selector` or `location` or `function`

Location up to which theorem environments will be displayed. The default `auto` uses the location where `thm-display` was called.

```
#show: thm-rules
#set heading(numbering: "1.1")

= Display up to `<h2>`

#thm-display(at: <h2>)
```

7 Display up to `<h2>`

Theorem 1.1 (Name)**.** *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Proof. Lorem ipsum dolor sit amet, consectetur adipiscing. ■

Default: `auto`

final `boolean`

If `true`, display all theorem environments up to the end of the document. Useful for creating lists of theorems in the beginning of documents, before they've been stated. Overrides `at`.

Default: `false`

thm-stored

State containing theorem environment data, as an array of `thm` dictionaries. See `thm-display()` for details on the structure of each `thm`.

thm-plain

Creates a plain theorem environment. Identical to `thm-box()`, with different defaults.

```
#show: thm-rules

#let theorem = thm-plain(
  "Theorem",
  base: none
)

#let lemma = thm-plain(
  "Lemma",
  counter: "Theorem",
  base: none
)

#let corollary = thm-plain(
  "Corollary",
  base: "Theorem"
)

#lemma[#lorem(3)]
#theorem("Named")[#lorem(4)]
#corollary[#lorem(7)]
#theorem[#lorem(7)]
```

Lemma 1. *Lorem ipsum dolor.*

Theorem 2 (Named). *Lorem ipsum dolor sit.*

Corollary 2.1. *Lorem ipsum dolor sit amet, consectetur adipiscing.*

Theorem 3. *Lorem ipsum dolor sit amet, consectetur adipiscing.*

thm-def

Creates a theorem environment, suitable for definitions. Identical to `thm-box()`, with different defaults.

```
#show: thm-rules

#let definition = thm-def(
  "Definition",
  base: none
)

#definition[#lorem(7)]
#definition[#lorem(4)]
```

Definition 1. Lorem ipsum dolor sit amet, consectetur adipiscing.

Definition 2. Lorem ipsum dolor sit.

thm-rem

Creates a theorem environment, suitable for remarks. Identical to `thm-box()`, with different defaults.

```
#show: thm-rules

#let remark = thm-rem(
  "Remark",
  base: none
)

#remark[#lorem(3)]
#remark[#lorem(6)]
```

Remark. Lorem ipsum dolor.

Remark. Lorem ipsum dolor sit amet, consectetur.

qedhere

If placed in a block equation/enum/list within a proof, place a qed symbol to its right.

```
#show: thm-rules

#let proof = thm-proof("Proof")

#proof[
  #lorem(3)
  $ x^2 + y^2 = z^2. #qedhere $
]

#proof[
  + #lorem(4)
  + #lorem(5) #qedhere
]

#proof[
  $
    (a + b)^2 &= (a + b)(a + b) \
              &= a^2 + 2 a b + b^2. #qedhere
  $
]
```

Proof. Lorem ipsum dolor.

$$x^2 + y^2 = z^2. \qquad ■$$

Proof.
1. Lorem ipsum dolor sit.
2. Lorem ipsum dolor sit amet. ■

Proof.

$$\begin{aligned} (a+b)^2 &= (a+b)(a+b) \\ &= a^2 + 2ab + b^2. \qquad ■ \end{aligned}$$

thm-proof

Creates a proof environment Identical to `thm-rem`, with different defaults.

```
#show: thm-rules

#let theorem = thm-plain(
  "Theorem",
  base: none
)
#let proof = thm-proof("Proof")

#theorem[#lorem(6)]
#proof[#lorem(3)]
```

Theorem 1. *Lorem ipsum dolor sit amet, consectetur.*

Proof. Lorem ipsum dolor. ■