Module documentation

Geometry (`geometry.typ`)

Generalist functions for 1D and 2D geometry.

- between()
- intersects()
- resolve()

between

Testing `a <= b <= c`, helps only computing `b` once.

Parameters

```
between(
  a: length,
  b: length,
  c: length
) -> bool
```

a `length`

Lower bound.

b `length`

Tested value.

c `length`

Upper bound. Asserted to be `>= c`.

intersects

Tests if two intervals intersect.

Parameters

```
intersects(
  i1: (length, length),
  i2: (length, length),
  tolerance: length
)
```

i1 `(length, length)`

First interval as a tuple of `(low, high)` in absolute lengths.

i2 `(length, length)`

Second interval.

tolerance `length`

Set to nonzero to ignore small intersections.

Default: `0pt`

resolve
Converts relative and contextual lengths to absolute. The return value will contain each of the arguments once converted, with arguments that contain `'x'` or start with `'w'` being interpreted as horizontal, and arguments that contain `'y'` or start with `'h'` being interpreted as vertical.

```
#context resolve(
    (width: 100pt, height: 200pt),
    x: 10%, y: 50% + 1pt,
    width: 50%, height: 5pt,
)
```

```
(x: 10pt, y: 101pt, width: 50pt, height: 5pt)
```

Parameters

```
resolve(
  size: (width: length, height: length),
  ..args: dictionary
) -> dictionary
```

size `(width: length, height: length)`

Size of the container as given by the `layout` function.

Tiling (`tiling/default.typ`)

- separate()

Variables

- pat-forbidden
- pat-allowed

separate
Splits content into obstacles, containers, and flowing text.

An "obstacle" is any content inside a `place` at the toplevel. It will be appended in order to the `placed` field as `content`.

A "container" is a `box(place({}))`. Both `box` and `place` are allowed to have `width`, `height`, etc. parameters, but no inner contents. It will be appended in order to the `free` field as a block, i.e. a dictionary with the fields `x`, `y`, `width`, `height` describing the upper left corner and the dimensions of the container.

Everything that is neither obstacle nor container is flowing text, and will end in the field `flow`.

```
#separate[
  // This is an obstacle
  #place(top + left, box(width: 50pt, height: 50pt))
  // This is a container
```

```
  #box(height: 50%, place({}))
  // This is flowing text
  #lorem(50)
]
```

Parameters

`separate(ct: content) -> (flow: array(block), placed: array(content), free: content)`

pat-forbidden

Pattern with red crosses to display forbidden zones.

pat-allowed

Pattern with blue pluses to display allowed zones.