wicked

$$: \overline{\Psi}_\alpha (x) \gamma^\mu_{\alpha\beta} A_\mu (x) \Psi_\beta (x) \overline{\Psi}_\eta (y) \gamma^\nu_{\eta\rho} (y) A_\nu \Psi_\rho (y) :$$

```
$ :
wick(id: #1, overline(Psi))_alpha (x)
gamma^mu_(alpha beta)
wick(pos: #top, A)_mu (x)
wick(Psi)_beta (x)
wick(overline(Psi))_eta (y)
gamma^nu_(eta rho) (y)
wick(pos: #top, A)_nu
wick(id: #1, Psi)_rho (y)
: $
```

This small Typst package handles the typesetting of Wick contractions $\phi^\dagger \phi$. We shall not fall into the same typographical limitations that stopped the italian physicist Gian Carlo Wick from using this notation in his papers.

> *"Houriet and Kind's symbol is a line connecting the two factors like a string attached at both ends. It is very convenient for handwriting, but has been abandoned here for typographical reasons."*
>
> — Gian Carlo Wick [PhysRev.80.268]

Basic usage

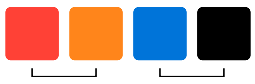
In the examples which follow we must suppose that the following preamble has been used.

```
#import "@preview/wicked:0.2.0": wick

// Only used to make the examples shorter
#set box(width: 20pt, height: 20pt, radius: 2pt)
```

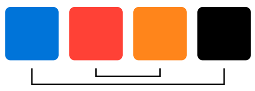
The whole package functionality is contained within the `wick` function. By calling `wick(..)` on some content you place a *contraction point*. Every second contraction point you place, gets contracted with the previous one.

```
$
#wick(box(fill: red))
#wick(box(fill: orange))
#wick(box(fill: blue))
#wick(box(fill: black))
$
```



By default every contraction point is given an `id` equal to the integer `0`. Since only contraction points with the same `id` get contracted together, by specifying different `id`s is possible to get multiple overlapping contractions.

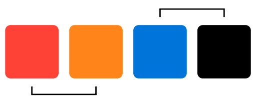
```
$
#wick(id: 1, box(fill: blue))
#wick(box(fill: red))
#wick(box(fill: orange))
#wick(id: 1, box(fill: black))
$
```



The `id` is not required to be an integer; for example `string`s are another reasonable choice. That being said, integer `id`s have a special behavior. Did you notice how the outer contraction in the last example automatically moved further away? When `id` is of type `int`, the distance of the contraction line, which is otherwise specified by the parameter `dist`, is instead computed as `dist * (1 + id / 2.0)`.

The `pos` parameter of the `wick` function allows the user to specify whether the contraction should be drawn above `pos: top` or below `pos: bottom` the equation. The default is to contract below.

```
$
#wick(pos: bottom, box(fill: red))
#wick(pos: bottom, box(fill: orange))
#wick(pos: top, box(fill: blue))
#wick(pos: top, box(fill: black))
$
```

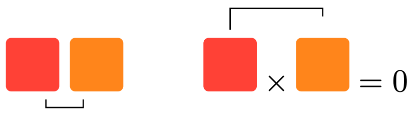


Only contraction points with the same value of `pos` can be contracted together, just like for `id`. Other then these two, there are no other constraints of which contraction points can be actually contracted.

Point specific styling

At the moment, the only point specific styling options are `dx` and `dy`, which shift the contraction point location.

```
$
#wick(box(fill: red), dx: 5pt)
#wick(box(fill: orange), dx: -5pt)
#h(30pt)
#wick(pos: top, box(fill: red)) times
#wick(pos: top, dy: 5pt, box(fill: orange)) = 0
$
```

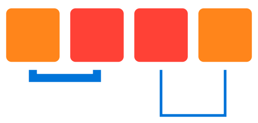


A negative/positive `dx` moves the point left/right, while a negative/positive `dy` brings the point closer/further.

Shared styling options

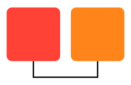
There are some styling options that regard the contraction as a whole. In this scenario both the first and the second contraction point can specify the styling option, and we say that the option is *shared*. The default value of every shared styling option is `auto`. If the two contraction points set different values for the same shared option, the one specified by the second contraction point wins. If no value is specified a default is used.

```
$
#wick(box(fill: orange), stroke: blue + 3pt)
#wick(box(fill: red))
#wick(box(fill: red), dist: 20pt, stroke: 3pt)
#wick(box(fill: orange), stroke: blue)
$
```



Setting the `offset` parameter, it's possible to specify the clearance between the contracted elements of the equation and the start of the contraction line.

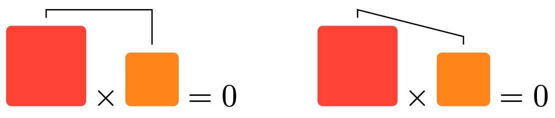
```
$
#wick(box(fill: red))
#wick(box(fill: orange), offset: 0pt)
$
```



Since `dist` is the total vertical distance between the contracted element and the horizontal contraction line (up to the effect of the before mentioned special rule for integer `id`s), setting `offset` to a value bigger than `dist` should be avoided in normal circumstances.

A boolean option called `flat` allows to specify whether the contraction line should be kept horizontal or not.

```
#wick(pos: top, box(width: 30pt, height: 30pt, fill: red)) times
#wick(pos: top, box(fill: orange)) = 0
#h(30pt)
#wick(pos: top, box(width: 30pt, height: 30pt, fill: red)) times
#wick(pos: top, flat: false, box(fill: orange)) = 0
```



Argument	**Positional**	**Type**	**Default**	**Shared**
`id`	no	`any`	`0`	-
`pos`	no	`alignment`	`bottom`	-
`dx`	no	`length`	`0pt`	no
`dy`	no	`length`	`0pt`	no
`dist`	no	`length`	`auto` is `0.5em`	yes
`offset`	no	`length`	`auto` is `0.25em`	yes
`stroke`	no	`stroke`	`auto` is `0.5pt`	yes
`flat`	no	`bool`	`auto` is `true`	yes
`body`	yes	`content`	-	no

Table 1: Input arguments of the `wick` function