Normal `#raw()` works as expected:

```
/*
 * Example taken from
 * https://typst.app/docs/tutorial/formatting/
 */
#show "ArtosFlow": name => box[
  #box(image(
    "logo.svg",
    height: 0.7em,
  ))
  #name
]

// Long line that breaks
This report is embedded in the ArtosFlow project. ArtosFlow is a project of the
Artos Institute.

// Very long line without linebreak
This_report_is_embedded_in_the_ArtosFlow_project._ArtosFlow_is_a_project_of_the_Artos_Institute.

// End example
```

Using `#sourcecode()` will add line numbers. Very long lines will be clipped.

```
1   /*
2    * Example taken from
3    * https://typst.app/docs/tutorial/formatting/
4    */
5   #show "ArtosFlow": name => box[
6     #box(image(
7       "logo.svg",
8       height: 0.7em,
9     ))
10    #name
11  ]
12
13
14  This report is embedded in the ArtosFlow project. ArtosFlow is a project of the
    Artos Institute.
15
16
17  This_report_is_embedded_in_the_ArtosFlow_project._ArtosFlow_is_a_project_of_the
18
19
```

Sourcecode can be loaded from a file and passed to `#sourcefile()`. Any CODELST sourcecode can be wrapped inside `#figure()` as expected.

CODELST blocks line numbers can be formatted via a `#show()` rules like:

```
show <codelst>: (code) => { ... }
show <lineno>: (n) => { ... }
```

To the right in Listing 1 you can see the `typst.toml` file of this package with some *fance line numbers*.

```toml
[package]
name = "codelst"
version = "0.0.3"
entrypoint = "codelst.typ"
authors = ["Jonas Neugebauer"]
license = "MIT"
description = "A typst package to render sourcecode"
repository = "https://github.com/jneug/typst-
```

Listing 1: typst.toml

CODELST attempts to add a minimal amount of formatting. You can use your own styles via `#show()` rules. For easy formatting, some default styles like a colored block can be applied using `#codelst-styles()`:

```
#show: codelst-styles
```

`#sourcecode()` accepts a number of arguments to affect the output like *highlighting lines*, *restrict the line range* or *place labels* in specific lines to reference them later.

```
9   #"hello world!" \
10  #"\"hello\n  world\"!" \
11  #"1 2 3".split() \
12  #"1,2;3".split(regex("[,;]")) \
13  #(regex("\d+") in "ten euros") \
14  #(regex("\d+") in "10 euros")
```

To reference a line use `#lineref()`:

- See line 11 for an example of the `split()` function.