

Chribel Template

Template Description v1.1.1

Joel / [Source Code](#)

Contents

How to use the template	1
Multiple Authors and Links	1
Callout	2
Language Context	2
Callout Styles	2
Custom Callouts	3
Invalid Callout Type	3
Adding Callout Types	3
Additional used snippets	4
Highlight Links	4
Arguments Box	4
Changelog	4
1.1.1	4
1.1.0	4
1.0.0	4

How to use the template

1. Install fonts:

- ↳ Atkinson Hyperlegible Next ([Link](#))
- ↳ Fantasque Sans Mono ([Link](#))
- ↳ Arvo ([Link](#))
- ↳ Tabler.io Icons ([Link](#) or if you don't have the money [Link](#))

2. Import the package

```
#import "@preview/chribel:1.0.0": chribel,
callout, // chribel-add-callout-template // if
custom defined callouts is desired
```

3. Then put the following block before your content to then edit or to see where the fields are all used in the document.

```
#show: chribel.with(
    title: [`<title>`],
    subtitle: [`<subtitle>`],
    authors: ([`<authors>`]),
    subject: (
        short: [`<subject-short>`],
        long: [`<subject-long>`]
    ),
    columns-count: 3,
    links: (text: [`<link-text>`], link: "<link-
href>"),
    university: [`<university>`],
    creation-date: datetime.today()
    accent-color: rgb("#2e6bba")
)
```

Arguments

title `content` default: `none`

The title of the document

subtitle `content` default: `none`

The subtitle of the document

authors `content` | `array (content)` default: `none`

A single author or list of authors, which gets displayed in the [titleblock](#) on the first page.

subject `dictionary` default: `(none,none)`

If the document is for a specific subject, this dictionary can be configured with the entries `long` and `short` (although `long` is actually not used). Use the entry `short` for everything else, if not applicable.

university `content` default: `none`

Name of the university or workplace

columns-count `integer` default: `2`

Splits the document into the given amount of columns.

links `dictionary` | `array (dictionary)` default: `none`

A dictionary with the entries `text` and `href` to display links in the [titleblock](#). Also supports multiple links by passing an array of dictionaries.

creation-date `datetime` default: `#datetime.today()`

The date of creation of the document, which will be displayed in the left footer side.

accent-color `color` default: `#rgb("2e6bba")`

The color used for the heading underlines and link boxes in the titleblock

Multiple Authors and Links

The template supports multiple authors and links, which have to be given as arrays. The authors are an array of `content`, while the links would be a dictionary array with the entries `text` for the replacement text and `link` for the hyperlink itself.

Example

```
title: [Template],
subtitle: [Summary Template for Typst],
authors: [
    [The author], [The other author], [Some funny]
```

```
author]
),
links: (
  (text: [Source Code], link: "https://codeberg.org/"),
  (text: [Search Engine], link: "https://duckduckgo.com/"),
  (text: [Touch Grass], link: "https://www.touchgrassss.com/")
)
```

Result:

Template

Summary Template for Typst

The author, The other author, Some funny author /
[Source Code](#) [Search Engine](#) [Touch Grass](#)

Callout

The template includes some simple, minimal callouts to put emphasis on some content. To create one, `#callout` with the following parameters can be used.

Information

This callout is created using following snippet

```
#callout(type: "info") [This callout is created using  
following command]
```

```
#callout(  
  style: "minimal", type: "info",  
  title: none, width: 100%,  
  height: auto, icon: auto,  
  paint: none,  
  content  
)
```

Arguments

style `string | function` default: "minimal"
 Determines how the callout is rendered. Valid values are "minimal", "quarto", "compact". If a function in the shape of `(title, icon, content, paint, height, width) => {}` is given, this function is instead called!

type `string` default: "info"
 Possible values are "info", "warning", "important", "caution", "tip", "correct", "incorrect", "example". If a different string is given, a "invalid" callout box is given, which is just gray and has a question mark as an icon.

title `content` default: none
 The title of the callout block.
`none`: the title for the given `type` is used.

width `length` default: 100%
 Width of the callout box. When smaller than the maximum width, the box is automatically centered.

height `length` default: auto
 Height of the callout box. Recommended to leave at `auto`.

icon `string` default: auto
 The icon, which gets displayed next to the callout title. The name can be taken from [tabler.io/icons](#).
`auto` uses the assigned icon of the given `type`.
`none` disables the icon.

paint `color` default: none
 Overwrite the defaulted color of the callout box and title color.
`none` uses assigned colors of the given `type`.

Language Context

The provided callouts adapt their default titles based on the document language. So far only German (de) and English (en) are supported and shown below:

```
#set text(lang: "en")
```

Information**Warning**

```
#set text(lang: "de")
```

Important**Information****Caution****Warnung****Tip****Wichtig****Vorsicht****Tipp**

Callout Styles

By default, the style "minimal" is used. There are two other predefined styles called "quarto" and "compact". The first one is a replication of the [Quarto Callouts](#), the latter is an inline based callout.

→ Using style "quarto":

Information

Lorem ipsum dolor sit.

Warning

Lorem ipsum dolor sit.

Important

Lorem ipsum dolor sit.

Caution

Lorem ipsum dolor sit.

Tip

Lorem ipsum dolor sit.

Correct

Lorem ipsum dolor sit.

Incorrect

Lorem ipsum dolor sit.

Example

Lorem ipsum dolor sit.

→ Using style "compact":

Information Lorem ipsum dolor sit **Warning**
 Lorem ipsum dolor sit **Important** Lorem ipsum dolor sit **Caution** Lorem ipsum dolor sit **Tip** Lorem

ipsum dolor sit ✓ **Correct** Lorem ipsum dolor sit.
 ✗ **Incorrect** Lorem ipsum dolor sit ✗ **Example** Lorem ipsum dolor sit.

Custom Callouts

If you don't like the provided styles, you can create your own ones. Define a function (`#let func(..)`) or inline function (`(..) => {}`) similar to the example below. Then pass a function in the shape of...

```
(title, icon, content, paint, height, width) => {}
```

...to the `style`-parameter of the `#callout(..)` function.

A Caution The parameter `icon` returns the converted icon (not the label, but the actual icon)

```
#let custom-callout(
  title,
  icon,
  content,
  paint,
  height,
  width) = {
  block(
    stroke: paint, inset: 0.5em,
    width: 100%, fill: white,
    stack(
      dir: ttb, spacing: 0.5em,
      strong[#icon #title #icon],
      content
    )
  )
}

#callout(style: custom-callout)[Hello World]
```

Output
 ⓘ Information ⓘ
 Hello World

Set default styling

If you want to replace the default styling ("minimal") with your own one, overwrite the `#callout(..)` function:

```
#callout[Hello]

#let custom-callout(title,icon,content,paint,
height,width) = { box(stroke: paint, inset:
0.25em, fill: white, radius: 0.5em,
[#strong[#icon#title] #content])
}

#let callout = callout.with(style: custom-callout)

#callout[Hello]
```

Output
 ⓘ Information
 Hello

ⓘ Information Hello

Invalid Callout Type

As mentioned before, passing a callout type not in the type list generates an "invalid" callout. It's just a grayed out callout and a respective title is set. The title is also language specific!

? No Associated Title

```
#callout(type:"blabla")[]
```

Output
 ⓘ Information ⓘ
 Hello World

Adding Callout Types

It's also possible to add new callout styles in your template if needed. For this additionally import the `chribel-add-callout-template()` function:

```
#import "@preview/chribel:1.0.0": chribel, callout,
chribel-add-callout-template
```

And then you can add the configuration function anywhere in the function (but set styles **cannot** be used before it!).

```
#chribel-add-callout-template(
  name,
  config : (
    color: ...,
    icon: ...,
    placeholder: ...
  )
)
```

Arguments

name string

The name of the new callout type. This will be used in the `[callout]` function to access the type.

config string

The callout configuration, which gets applied to the

color color

The accent color for the callout stroke, background color and icon color.

icon string

The icon label name of the desired [tabler.io icons](#). Don't forget to add `-filled` if you want the filled variant of the icon.

placeholder content

The title of the callout. The name *Placeholder* is used, since you can overwrite the title and this content is used if no custom title is set.

```
#chribel-add-callout-template("dog", (
  color: rgb("#7c5735"),
  icon: "dog",
  placeholder: "Hello World",
))
#callout(type: "dog")[
  Hello World
]
```

↓ ----- Output ----- ↓

Hello World
Hello World

```
set align(left)
// title block
place(top+left, dy: -8pt - 0.5em, dx: -2pt,
  box(fill:white, inset: (x: 2pt), outset: (y:
1pt), text( gray, "Arguments")))
)
content
}
}
```

Additional used snippets

Highlight Links

```
#show link: underline
#show link: set text(blue)
```

Arguments Box

```
#let argument(content) = {
  show terms.item: it => context {
    set par(justify: false)
    let term = text(font: "Fantasque Sans
Mono", strong(it.term))
    set par(hanging-indent: 2em)
    term
    terms.separator
    it.description
    linebreak()
  }

  set text(0.9em)
  block(
    above: 1.5em, below: 1.5em,
    width: 100%, height: auto,
    stroke: gray + 0.5pt,
    inset: (top: 8pt, rest: 6pt),
    radius: 3pt,
  }
```

Changelog

1.1.1

- Reimplemented the `"compact"` styling to work inside other callouts.
- Added current version number to this manual
- Replaced `func` with extended `style` parameter → allows now selecting custom function or predefined styles.

1.1.0

- Expanded `#callout` functions with support for custom render functions (via parameter `func: ...`)
- Added `style` parameter to `#callout` with three options: `"minimal"`, `"quarto"`, `"compact"`
- Added callout type `"caution"` and `"important"`.
- Removed `sticky` parameter from raw blocks

1.0.0

- initial release – added functions `#callout`
`#chribel`, `#chribel-add-callout-template`