

# Rubber Article - Template Documentation

v.0.4.2

## Contents

Example Usage .....	1
Styling functions .....	1
appendix .....	1
article .....	2
Constructor functions .....	6
abstract .....	6
ctable .....	7
fig-outline .....	8
maketitle .....	9
shortcap .....	9
tab-outline .....	10
Utility functions .....	11
balance .....	11

## Example Usage

```
1 #import "@preview/rubber-article:0.4.2": *
2 #show: article.with()
3 #maketitle(
4   title: "The Title of the Paper",
5   authors: ("Authors Name",),
6   date: datetime.today().display("[day].[month].[year]"),
7 )
```

 Typst

## Styling functions

These functions are used to style certain elements of the document. They are usually called with a `#show` statement. They do not output any content, but rather modify the appearance of the document.


- `appendix()`
- `article()`

### appendix

Function to format the Appendix. This function is intended to be used after the document has been styled with the `article` function.

Example usage:

```
1 #show: article.with()
2 // A lot of content goes here...
3
4 #show: appendix.with(
5   title: "Appendix",
6   title-align: center,
```

 Typst

## Parameters

```
appendix(
  numbering: none or str or function,
  title: none or str or content,
  title-align: alignment,
  title-size: length,
  numbering-start: int,
  content
) -> content
```

**numbering** none or str or function

The numbering of the Appendix

Default: "A.1"

**title** none or str or content

The title of the Appendix

Default: none

**title-align** alignment

The alignment of the title

Default: center

**title-size** length

The size of the title

Default: none

**numbering-start** int

Starting the appendix after this number

Default: 0

## article

A Template recreating the look of the classic Article Class.

Example usage:

```
1 #show: article.with(
2   lang: "de",
```

This will format the document with the specified options. For more styling options, explore the following parameters.

### Parameters

```
article(
  lang: str,
  eq-numbering: str | none,
  eq-chapterwise: bool,
  text-size: length,
  page-numbering: none | str | function,
  page-numbering-align: alignment,
  heading-numbering: none | str | function,
  margins: auto | relative | dictionary,
  enum-indent: length,
  list-indent: length,
  header-display: bool,
  alternating-header: bool,
  first-page-header: int | float,
  header-title: str | content,
  justify: bool,
  cols: none | int,
  fig-caption-width: relative,
  header-line-stroke: length,
  content: content
) -> content
```

**lang**    `str`

Set the language of the document.

Default: `"de"`

**eq-numbering**    `str` or `none`

Set the equation numbering style.

Default: `none`

**eq-chapterwise**    `bool`

Chapterwise numbering of equations.

Default: `false`

**text-size**    `length`

Set the text size. Headings are adjusted automatically.

Default: `10pt`

**page-numbering** `none` or `str` or `function`

Set the page numbering style.

Default: `"1"`

**page-numbering-align** `alignment`

Set the page numbering alignment.

Default: `center`

**heading-numbering** `none` or `str` or `function`

Set the heading numbering style.

Default: `"1.1"`

**margins** `auto` or `relative` or `dictionary`

Set the margins of the document.

Default: (left: `25mm`, right: `25mm`, top: `30mm`, bottom: `30mm`)

**enum-indent** `length`

Set the Enum indentation.

Default: `1.5em`

**list-indent** `length`

Set the List indentation.

Default: `1.5em`

**header-display** `bool`

Set if the default header should be used.

Default: `false`

**alternating-header** `bool`

Set if the default header should be alternating.

Default: `true`

**first-page-header** `int` or `float`

Set the first page of the header.

Default: `1`

**header-title** `str` or `content`

Set the Header Titel

Default: `none`

**justify** `bool`

Set if document should be justified.

Default: `true`

**cols** `none` or `int`

Set the number of columns, that the document should have.

Default: `none`

**fig-caption-width** `relative`

Set the width of the figure captions.

Default: `75%`

**header-line-stroke** `length`

Set the width of the headerline.

Default: `.65pt`

## Constructor functions

These functions are used to create certain elements of the document. They can be called with certain arguments and output some content.

- `abstract()`
- `ctable()`
- `fig-outline()`
- `maketitle()`
- `shortcap()`
- `tab-outline()`

### **abstract**

This function will display an abstract section with a title and content.

#### Parameters

```
abstract(  
  title: string content,  
  alignment: alignment,  
  outlined: bool,  
  numbering: numbering,  
  width: length,  
  content: string content  
)
```

**title**    string or content

The title of the abstract.

Default: "Abstract"

**alignment**    alignment

The alignment of the abstract.

Default: left

**outlined**    bool

If the heading should be outlined.

Default: true

**numbering**    numbering

The numbering of the heading.

Default: none

**width**   `length`

The width of the abstract block.

Default: `auto`

**content**   `string` or `content`

The content of the abstract.

## **ctable**

This function will display a custom table. The table uses the `pillar` package under the hood to interact with the table in a similar manner as in Latex. This means, that the columns and vertical lines can be defined with a string. Furthermore, the table automatically adds 3 horizontal lines.

Example usage:

```
1 #ctable(  
2   cols:"l|cr",  
3   [A], [B], [C],  
4   ..range(1,16).map(str),  
5 )
```

**t** Typst

## **Parameters**

```
ctable(  
  ..data,  
  cols: string,  
  stroke: length,  
  middle-stroke: length,  
  vertical-stroke: length,  
  header-rows: int  
) -> content
```

**cols**   `string`

A string that defines the columns and vertical lines of the table.

Default: `"ccc"`

**stroke**   `length`

The linesyle of the table, especially the top and bottom horizontal lines.

Default: `.75pt`

**middle-stroke**    `length`

The linesyle of the middle horizontal line.

Default: `.6pt`

**vertical-stroke**    `length`

The linesyle of the vertical lines.

Default: `.6pt`

**header-rows**    `int`

The number of header rows.


Default: `1`

## fig-outline

This function is a wrapper of the `outline` function and allows for an easy way to create the table of figures.

Example usage:

```
1 #fig-outline()
```

 Typst

### Parameters

```
fig-outline(  
  title: string content,  
  target: label selector location function,  
  ..args  
) -> content
```

**title**    `string` or `content`

The title of the table of figures.

Default: `"List of Figures"`

**target**    `label` or `selector` or `location` or `function`

The Target of the table of figures.

Default: `figure.where(kind: image)`

**..args**

Additional optional arguments to the `outline` function.



## maketitle

This function will display the frontmatter of the document. This includes the title, authors, and date.

Example usage:

```
1 #maketitle(  
2   title: "The Title of the Paper",  
3   authors: ("Authors Name",),  
4   date: "2025-01-01",  
5 )
```

## Parameters

```
maketitle(  
  title: string content ,  
  authors: array ,  
  date: string content datetime ,  
  metadata: bool  
) -> content
```

**title** string or content

The title of the document.

Default: ""

**authors** array

The authors of the document.

Default: ()

**date** string or content or datetime

The date of the document.

Default: none

**metadata** bool

Use title and author information for the document metadata. This does not work when article sets columns before!!!

Default: false

## shortcap

This function will help you to provide a long caption to a figure, but a short caption to the outline.

Example usage:

```
1 #figure(  
t Typst
```

```

2   rect(),
3   caption: shortcap("Short caption", "Long caption"),
4 )

```

### Parameters

```

shortcap(
  short: string content,
  long: string content
) -> content

```

**short**    string or content

The short caption of the figure.

**long**    string or content


The long caption of the figure.

### tab-outline

This function is a wrapper of the `outline` function and allows for an easy way to create the list of tables.

Example usage:

```
1 #tab-outline()
```

 Typst

### Parameters

```

tab-outline(
  title: string content,
  target: label selector location function,
  ..args
) -> content

```

**title**    string or content

The title of the table of figures.

Default: "List of Tables"

**target**    label or selector or location or function

The Target of the table of figures.

Default: `figure.where(kind: table)`

**..args**

Additional optional arguments to the outline function.

## Utility functions

These functions can be used to perform certain tasks in the document. These functions will help you style certain elements of the document, where otherwise complicated functions would be needed.


- `balance()`

### **balance**

Balance the content of columns. Credits go to: <https://github.com/typst/typst/issues/466>

Example usage:

```
1 #balance(columns(n)[#lorem(80)])
```

 Typst

### **Parameters**

`balance(content)` -> `content`