

Typst 简明使用教程

卓能文

2024-01-12

目 录

1 Typst 简介	1
2 Typst 安装	1
3 Typst 使用	1
3.1 创建文件	1
3.2 章节设置	1
3.3 编译	2
3.4 注意几个特殊字符	2
3.5 显示图片	2
3.5.1 设置宽度:	2
3.5.2 居中显示:	3
3.5.3 设置标题:	3
3.5.4 多图并列	3
3.5.5 多图并列（带标题）	4
3.5.6 多图并列含间距（带标题）	4
3.5.7 多图并列（带子标题）	5
3.5.8 多图并列（带子标题、子图无编号）	5
3.6 显示表格	6
3.7 显示公式	6
3.8 显示代码	7
3.8.1 添加标题	7
3.8.2 居左显示(codly)	7
3.8.3 显示代码文件	8
3.8.4 显示代码文件(sourcerer)	8
3.8.5 显示代码文件(codly)	8
3.9 标签与引用	8
3.10 参考文献设置	9
4 写在最后	9
A 附录	9
A.1 article 模板	9
A.2 本文档源码	11
参考文献	16

图形列表

图 3.1: 玫瑰	3
图 3.2: 多图并列（带标题）	4
图 3.3: 多图并列含间距（带标题）	5
图 3.4: 多图并列（带子标题）	5
图 3.5: 玫瑰 1	5
图 3.6: 玫瑰 2	5
图 3.7: 多图并列（带子标题、子图无编号）	6
图 3.8: 玫瑰	9

表格列表

表 3.1: 示例表格	6
-------------------	---

代码列表

代码 3.1: 计算斐波纳契	7
代码 3.2: 计算斐波纳契	8
代码 3.3: 计算斐波纳契	8
代码 3.4: 计算斐波纳契	8

1 Typst 简介

Typst 是撰写任何长篇文本（如论文、文章、科学论文、书籍、报告和家庭作业）的优秀工具。此外，Typst 非常适合于编写任何包含数学符号的文档，例如在数学、物理和工程领域的论文。最后，由于其强大的风格化和自动化功能，它是任何一组具有共同风格的文件的绝佳选择，例如丛书。Typst 文档风格和 md 文档类似，所以很容易上手，同时内置了强大的脚本功能及较多的排版原语，因此，能比较轻松完成优质文档的撰写及排版工作。

2 Typst 安装

Typst 的本地安装非常简单，直接从 <https://github.com/typst/typst/releases> 下载适合自己操作系统的版本，解压到适当的地方即完成安装。另外，也可以在 <https://typst.app> 上注册账号，在线编辑 typst 文档，并下载生成的 PDF 文档。

对初学者编辑器建议采用 visual studio code，并安装 Typst LSP 和 Typst Preview 插件。老手可以安装 sumlime text 并安装 typst 插件。

⚠ Warning

当你的文档内容比较多的时候，VS code 反应将变得极其缓慢，建议对文件进行切分或换 sumlime text 编辑器。当然，最好用的编辑器还是 helix！

3 Typst 使用

3.1 创建文件

新建文本文档，以 .typ 为后缀。建议克隆 <https://github.com/soarowl/typst.git> 到本地，并将其中的 article.typ 复制到文档所在的目录，并适当进行修改。然后在文档头部添加如下内容：

```
1 #import "article.typ":*
2
3 #show: article.with(title: "Typst 简明使用教程", authors: ("卓能文",))
```

3.2 章节设置

格式有点类似 markdown，比较简单：

```
1 = 第一章
2 内容
3
4 == 第一节
5 内容
6
7 == 第二节
8 内容
9
10 == 第三节
11 内容
12
13 = 第二章
14 == 第一节
15 内容
```

```
16
17 == 第二节
18 内容
19
20 = 第三章
```

3.3 编译

```
1 typst compile filename.typ
```

如果没有任何错误，将输出 `filename.pdf` 文档。

3.4 注意几个特殊字符

字符	意义	转义
*	两个*字符之间的文字将加粗显示	*
#	表示 <code>typst</code> 命令	\#
+	段首+ 有序列表	\+
-	段首- 无序列表	\-

3.5 显示图片

建议将图片保存在一个特定的目录，如 `images`、`img` 之类的地方。

```
1 #image("images/rose.jpg")
```

typst

3.5.1 设置宽度:

```
1 #image("images/rose.jpg", width: 50%)
```

typst



3.5.2 居中显示:

```
1 #align(center,image("images/rose.jpg", width: 50%))
```

typst



3.5.3 设置标题:

```
1 #figure(  
2   caption: [玫瑰],  
3   image("images/rose.jpg", width: 50%)  
4 )
```

typst



图 3.1 玫瑰

i Info

放入 `#figure` 命令中的图片同时会在图形列表中出现。

3.5.4 多图并列

```
1 #grid(  
2   columns: (1fr, 1fr),  
3   image("images/rose.jpg"),  
4   image("images/rose.jpg"),  
5 )
```

typst



3.5.5 多图并列（带标题）

```
1 #figure(  
2   caption: [多图并列（带标题）],  
3   grid(  
4     columns: (1fr, 1fr),  
5     image("images/rose.jpg"),  
6     image("images/rose.jpg"),  
7   )  
8 )
```

typst



图 3.2 多图并列（带标题）

3.5.6 多图并列含间距（带标题）

```
1 #figure(  
2   caption: [多图并列含间距（带标题）],  
3   grid(  
4     columns: (1fr, 1fr),  
5     gutter: 10pt,  
6     image("images/rose.jpg"),  
7     image("images/rose.jpg"),  
8   )  
9 )
```

typst



图 3.3 多图并列含间距（带标题）

3.5.7 多图并列（带子标题）

```
1 #figure(
2   caption: [多图并列（带子标题）],
3   grid(
4     columns: (1fr, 1fr),
5     gutter: 10pt,
6     figure(
7       caption: [玫瑰 1],
8       image("images/rose.jpg")
9     ),
10    figure(
11      caption: [玫瑰 2],
12      image("images/rose.jpg")
13    ),
14  )
15 )
```

typst



图 3.5 玫瑰 1



图 3.6 玫瑰 2

图 3.4 多图并列（带子标题）

3.5.8 多图并列（带子标题、子图无编号）

```
1 #figure(
2   caption: [多图并列（带子标题、子图无编号）],
3   grid(
4     columns: (1fr, 1fr),
5     gutter: 10pt,
6     [#image("images/rose.jpg")玫瑰 1],
7     [#image("images/rose.jpg")玫瑰 2],
8   )
9 )
```

typst



玫瑰 1



玫瑰 2

图 3.7 多图并列（带子标题、子图无编号）

3.6 显示表格

```
1 #figure(
2   caption: [示例表格],
3   kind: table,
4   supplement: "表",
5   ```tbl
6     Rx   Nx
7     Rx   Nx.
8   -
9   software|version
10  -
11     AFL|2.39b
12     Mutt|1.8.0
13     Ruby|1.8.7.374
14 TeX Live|2015
15   -
16   ```
17 )
```

typst

表 3.1 示例表格

software	version
AFL	2.39b
Mutt	1.8.0
Ruby	1.8.7.374
TeX Live	2015

i Info

由于目前 Typst 中有 bug，显示表格时，必须加上 `kind` 和 `supplement` 字段。

更多用法请参考 <https://github.com/maxcrees/tbl.typ>

3.7 显示公式

```
1 勾股定理可用公式:  $a^2 + b^2 = c^2$  表示。
```

typst

勾股定理可用公式: $a^2 + b^2 = c^2$ 表示。

```
1 $ \sum_{k=1}^n k = (n(n+1)) / 2 $
```

typst

$$\sum_{k=1}^n k = \frac{n(n+1)}{2} \quad (1)$$

Typst 默认只能显示一级公式，不能按章节重新计数，可采用第三方包 `i-figured` 实现，本模板已经内置。格式请参考 `latex` 相关文档。

3.8 显示代码

i Info

目前，`codly` 显示代码有些问题，如部分代码在换页时被遮挡，超长代码不自动换行处理等。暂时换为 `sourcerer` 包进行代码显示。

代码可以很容易添加，格式和 `markdown` 一样。

```
1 ```py3
2 def fibonacci(n):
3     if n <= 1:
4         return n
5     else:
6         return fibonacci(n - 1) + fibonacci(n - 2)
7 ```
```

typst

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n - 1) + fibonacci(n - 2)
```

3.8.1 添加标题

```
1 #figure(
2     caption: [计算斐波纳契],
3     ```py3
4     def fibonacci(n):
5         if n <= 1:
6             return n
7         else:
8             return fibonacci(n - 1) + fibonacci(n - 2)
9     ```
10 )
```

typst

代码 3.1 计算斐波纳契

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n - 1) + fibonacci(n - 2)
```

3.8.2 居左显示(codly)

因为 `figure` 命令会导致代码居中显示，添加 `align(start)` 命令让代码居左：

```
1 #figure(
2     caption: [计算斐波纳契],
3     align(start)[
4         ```py3
5         def fibonacci(n):
6             if n <= 1:
7                 return n
8             else:
9                 return fibonacci(n - 1) + fibonacci(n - 2)
```

typst

```
10 ``
11 ]
12 )
```

代码 3.2 计算斐波纳契

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n - 1) + fibonacci(n - 2)
```

3.8.3 显示代码文件

在 Typst 文档中添加太多代码，导致可读性降低，也不便于后续采用相应的工具进行编辑、更新、管理与维护，建议将代码组织在一个文件夹中。

```
1 #figure(
2     caption: [计算斐波纳契],
3     raw(read("src/fibonacci.py"), lang: "py3", block: true)
4 )
```

typst

代码 3.3 计算斐波纳契

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n - 1) + fibonacci(n - 2)
6
```

3.8.4 显示代码文件(sourcerer)

```
1 #figure(
2     caption: [计算斐波纳契],
3     code(raw(read("src/fibonacci.py"), lang: "py3", block: true), lang: "python")
4 )
```

3.8.5 显示代码文件(codly)

```
1 #figure(
2     caption: [计算斐波纳契],
3     align(start, raw(read("src/fibonacci.py"), lang: "py3", block: true))
4 )
```

typst

代码 3.4 计算斐波纳契

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     else:
5         return fibonacci(n - 1) + fibonacci(n - 2)
6
```

3.9 标签与引用

在被引用的图表等地方用<name>设置标签，在打算引用的地方输入@name 即可。name 后面如果是中文，添加一个空格可避免编译错误。在 i-figured 中，需要在引用的地方添加 fig:、tbl:、lst: 等，形成@fig:name 形式。如：图 3.8 所示。



图 3.8 玫瑰

3.10 参考文献设置

参考文献设置也比较简单，只需在文件尾部加入 `#bibliography("example.yml", style: "gb-7714-2015-numeric")` 即可。yml 格式如下：

```
1 audio-descriptions:
2   affiliated:
3     - names: Taylor, Dallas
4       role: narrator
5   author: Barrows, Miellyn Fitzwater
6   date: 2017-02-07
7   issue: 8
8   parent:
9     author: Taylor, Dallas
10    title: Twenty Thousand Hertz
11    type: Audio
12  title: Audio Descriptions
13  type: Audio
14  url: https://www.20k.org/episodes/audio
15 barb:
16   author: Günther-Haug, Barbara
17   date: 2020
18   language: de-DE
19   location: München
20   publisher: MVG
21   title: Den Boden unter den Füßen verlieren
22   type: Book
```

在文章适当的地方插入 `@audio-descriptions[1]` 或 `@barb[2]` 这类的键。

4 写在最后

Typst 相对来说还比较新，功能和 latex 相比稍弱，同时还存在一些 bug。如果使用过程中有任何建议或模板上有什么问题，请到 <https://github.com/soarowl/typst.git> 提要求。

A 附录

A.1 article 模板

```
1 #import "@preview/gentle-clues:0.6.0":*
2 #import "@preview/i-figured:0.2.3"
3 #import "@preview/sourcerer:0.2.1": code
4 #import "@preview/tbl:0.0.4"
```

```

5
6 #let article(
7   title: "",
8   authors: (),
9   date: datetime.today().display(),
10  logo: none,
11  body,
12 ) = {
13   set document(title: title, author: authors.join(" "))
14   set heading(numbering: "1.1")
15   set text(font: ("Times New Roman", "SimSun"), lang: "zh")
16
17   //***** 图形、代码及表格列表设置
18   // this `level: 2` instructs the figure counters to be reset for every
19   // level 2 section, so at every level 1 and level 2 heading.
20   show heading: i-figured.reset-counters.with(level: 1)
21   // this `level: 2` instructs the figure numbering to include the first
22   // two levels of the current heading numbering.
23   // how this should behave with zeros can be set using `zero-fill`.
24   // e.g., setting `zero-fill: false` and `leading-zero: false` assures
25   // there is never a `0` in the numbering.
26   show figure: i-figured.show-figure.with(level: 1)
27   // master 版本不能编译
28   // show math.equation: i-figured.show-equation
29
30   // set figure(numbering: "1-1") // don't work, maybe a typst bug
31   set figure.caption(position: top, separator: [#h(1em)])
32   show figure.where(kind: image): set figure.caption(position: bottom)
33   //*****
34
35   //***** 代码框设置
36   show raw.where(block: true): it => {
37     code(it)
38   }
39   //*****
40
41   //***** 表格设置
42   show: tbl.template.with(box: false, breakable: true, tab: "|")
43   //*****
44
45   //***** 标题页设置
46   // The page can contain a logo if you pass one with `logo: "logo.png"`.
47   if logo != none {
48     v(0.4fr)
49     align(right, image(logo, width: 26%))
50     v(9.6fr)
51   }
52
53   set align(center)
54
55   v(20fr, weak: true)
56   text(2em, weight: 700, title)
57
58   // 作者
59   v(1.5em, weak: true)
60   let by = authors.map(author => [#strong(author)]).join(" ")
61   text(1.2em, by)
62
63   v(70fr)
64   text(1.1em, date)
65
66   set align(left)
67   pagebreak()
68   //*****
69
70   //***** 页眉、页脚

```

```

71 set page(
72   header: [#h(1fr)#title#h(1fr)#line(length: 100%, stroke: 2pt)],
73   number-align: center,
74 )
75 //*****
76
77 //***** 目录
78 set page(numbering: "I")
79 counter(page).update(1)
80
81 show outline: it => {
82   show heading: set align(center)
83   it
84 }
85 outline(title: [目#h(2em)录], indent: true, depth: 3)
86 i-figured.outline(title: [图形列表])
87 i-figured.outline(target-kind: table, title: [表格列表])
88 i-figured.outline(target-kind: raw, title: [代码列表])
89 // master 版本不能编译
90 // i-figured.outline(target: math.equation, title: [公式列表])
91 pagebreak()
92 //*****
93
94 //***** 正文
95 set page(numbering: "1")
96 counter(page).update(1)
97 set par(first-line-indent: 2em, justify: true)
98 show par: set block(spacing: 0.65em)
99 // Workaround 3: Automatically add empty paragraph after heading
100 show heading: it => {
101   it
102   par(text(size: 0.35em, h(0.0em)))
103 } // Only works for paragraphs directly after heading
104
105 body
106 //*****
107 }
108

```

A.2 本文档源码

```

1  #import "article.typ":*
2
3  #show: article.with(title: "Typst 简明使用教程", authors: ("卓能文",))
4
5  // 加入公式编号
6  #set math.equation(numbering: "(1.1)")
7
8  #show raw.where(block: true, lang: "typst-ex"): it => {
9    let txt = it.text
10    code(raw(txt, lang: "typc", block: true), lang: "typst")
11    eval(txt, mode: "markup")
12  }
13
14  = Typst 简介
15  Typst 是撰写任何长篇文本（如论文、文章、科学论文、书籍、报告和家庭作业）的优秀工具。此外，Typst 非常适合于编写任何包含
  数学符号的文档，例如在数学、物理和工程领域的论文。最后，由于其强大的风格化和自动化功能，它是任何一组具有共同风格的文件的
  绝佳选择，例如丛书。Typst 文档风格和 md 文档类似，所以很容易上手，同时内置了强大的脚本功能及较多的排版原语，因此，能比较
  轻松完成优质文档的撰写及排版工作。
16
17  = Typst 安装
18  Typst 的本地安装非常简单，直接从 #link("https://github.com/typst/typst/releases") 下载适合自己操作系统的版本，解
  压到适当的地方即完成安装。另外，也可以在 #link("https://typst.app") 上注册账号，在线编辑 typst 文档，并下载生成的 PDF
  文档。

```

```

19
20 对初学者编辑器建议采用`visual studio code`，并安装`Typst LSP`和`Typst Preview`插件。老手可以安装`sumlime
    text`并安装`typst`插件。
21
22 #warning[当你的文档内容比较多的时候，VS code 反应将变得极其缓慢，建议对文件进行切分或换`sumlime text`编辑器。当然，
    最好用的编辑器还是 helix! ]
23
24 = Typst 使用
25
26 == 创建文件
27 新建文本文档，以`.typ`为后缀。建议克隆 #link("https://github.com/soarowl/typst.git")到本地，并将其中的
    `article.typ`复制到文档所在的目录，并适当进行修改。然后在文档头部添加如下内容：
28 ```typc
29 #import "article.typ":*
30
31 #show: article.with(title: "Typst 简明使用教程", authors: ("卓能文",))
32 ```
33
34 == 章节设置
35 格式有点类似 markdown，比较简单：
36 ```typc
37 = 第一章
38 内容
39
40 == 第一节
41 内容
42
43 == 第二节
44 内容
45
46 == 第三节
47 内容
48
49 = 第二章
50 == 第一节
51 内容
52
53 == 第二节
54 内容
55
56 = 第三章
57 ```
58
59 == 编译
60
61 ```sh
62 typst compile filename.typ
63 ```
64
65 如果没有任何错误，将输出`filename.pdf`文档。
66
67 == 注意几个特殊字符
68
69 ```tbl
70     C LX LX
71     C LX LX.
72 _
73 字符 | 意义 | 转义
74 _
75 \* | 两个\*字符之间的文字将加粗显示 | \\*
76 \# | 表示`typst`命令 | \\#
77 \+ | 段首\+ 有序列表 | \\+
78 \- | 段首\- 无序列表 | \\-
79 _

```



```

80  ``
81
82  == 显示图片
83  建议将图片保存在一个特定的目录，如`images`、`img`之类的地方。
84  ```typst-ex
85  #image("images/rose.jpg")
86  ```
87
88  === 设置宽度:
89  ```typst-ex
90  #image("images/rose.jpg", width: 50%)
91  ```
92
93  === 居中显示:
94  ```typst-ex
95  #align(center, image("images/rose.jpg", width: 50%))
96  ```
97
98  === 设置标题:
99  ```typst-ex
100 #figure(
101   caption: [玫瑰],
102   image("images/rose.jpg", width: 50%)
103 )
104 ```
105
106 #info[放入`#figure`命令中的图片同时会在图形列表中出现。]
107
108 === 多图并列
109 ```typst-ex
110 #grid(
111   columns: (lfr, lfr),
112   image("images/rose.jpg"),
113   image("images/rose.jpg"),
114 )
115 ```
116
117 === 多图并列（带标题）
118 ```typst-ex
119 #figure(
120   caption: [多图并列（带标题）],
121   grid(
122     columns: (lfr, lfr),
123     image("images/rose.jpg"),
124     image("images/rose.jpg"),
125   )
126 )
127 ```
128
129 === 多图并列含间距（带标题）
130 ```typst-ex
131 #figure(
132   caption: [多图并列含间距（带标题）],
133   grid(
134     columns: (lfr, lfr),
135     gutter: 10pt,
136     image("images/rose.jpg"),
137     image("images/rose.jpg"),
138   )
139 )
140 ```
141
142 === 多图并列（带子标题）
143 ```typst-ex
144 #figure(
145   caption: [多图并列（带子标题）],

```

```

146   grid(
147     columns: (1fr, 1fr),
148     gutter: 10pt,
149     figure(
150       caption: [玫瑰 1],
151       image("images/rose.jpg")
152     ),
153     figure(
154       caption: [玫瑰 2],
155       image("images/rose.jpg")
156     ),
157   )
158 )
159 ```
160
161 == 多图并列（带子标题、子图无编号）
162 ```typst-ex
163 #figure(
164   caption: [多图并列（带子标题、子图无编号）],
165   grid(
166     columns: (1fr, 1fr),
167     gutter: 10pt,
168     [#image("images/rose.jpg")玫瑰 1],
169     [#image("images/rose.jpg")玫瑰 2],
170   )
171 )
172 ```
173
174 == 显示表格
175 ```typst-ex
176 #figure(
177   caption: [示例表格],
178   kind: table,
179   supplement: "表",
180   ```tbl
181     Rx    Nx
182     Rx    Nx.
183   _
184   software|version
185   _
186     AFL|2.39b
187     Mutt|1.8.0
188     Ruby|1.8.7.374
189   TeX Live|2015
190   _
191   ```
192 )
193 ```
194
195 #info[由于目前 Typst 中有 bug，显示表格时，必须加上`kind`和`supplement`字段。]
196
197 更多用法请参考 #link("https://github.com/maxcree/tbl.typ")
198
199 == 显示公式
200 ```typst-ex
201 勾股定理可用公式:  $a^2 + b^2 = c^2$  表示。
202 ```
203
204 ```typst-ex
205 $ \sum_{k=1}^n k = (n(n+1)) / 2 $
206 ```
207
208 Typst 默认只能显示一级公式，不能按章节重新计数，可采用第三方包`i-figured`实现，本模板已经内置。格式请参考`latex`相关文档。
209
210 == 显示代码

```

```

211 #info[目前, `codly` 显示代码有些问题, 如部分代码在换页时被遮挡, 超长代码不自动换行处理等。暂时换为 `sourcerer` 包进行
    代码显示。]
212
213 代码可以很容易添加, 格式和 markdown 一样。
214 ```typst-ex
215 ```py3
216 def fibonacci(n):
217     if n <= 1:
218         return n
219     else:
220         return fibonacci(n - 1) + fibonacci(n - 2)
221 ```
222 ```
223
224 === 添加标题
225 ```typst-ex
226 #figure(
227     caption: [计算斐波纳契],
228     ```py3
229     def fibonacci(n):
230         if n <= 1:
231             return n
232         else:
233             return fibonacci(n - 1) + fibonacci(n - 2)
234     ```
235 )
236 ```
237
238 === 居左显示(codly)
239 因为 `figure` 命令会导致代码居中显示, 添加 `align(start)` 命令让代码居左:
240 ```typst-ex
241 #figure(
242     caption: [计算斐波纳契],
243     align(start)[
244     ```py3
245     def fibonacci(n):
246         if n <= 1:
247             return n
248         else:
249             return fibonacci(n - 1) + fibonacci(n - 2)
250     ```
251 ]
252 )
253 ```
254
255 === 显示代码文件
256 在 Typst 文档中添加太多代码, 导致可读性降低, 也不便于后续采用相应的工具进行编辑、更新、管理与维护, 建议将代码组织在一个
    文件夹中。
257 ```typst-ex
258 #figure(
259     caption: [计算斐波纳契],
260     raw(read("src/fibonacci.py"), lang: "py3", block: true)
261 )
262 ```
263
264 === 显示代码文件(sourcerer)
265 ```typc
266 #figure(
267     caption: [计算斐波纳契],
268     code(raw(read("src/fibonacci.py"), lang: "py3", block: true), lang: "python")
269 )
270 ```
271
272 === 显示代码文件(codly)
273 ```typst-ex
274 #figure(

```

```

275   caption: [计算斐波纳契],
276   align(start, raw(read("src/fibonacci.py"), lang: "py3", block: true))
277 )
278 ```
279
280 == 标签与引用
281 在被引用的图表等地方用`<name>`设置标签，在打算引用的地方输入`@name`即可。name 后面如果是中文，添加一个空格可避免编译
    错误。在`i-figured`中，需要在引用的地方添加`fig:`、`tbl:`、`lst:`等，形成`@fig:name`形式。如：`@fig:rose` 所示。
282
283 #figure(caption: [玫瑰], image("images/rose.jpg", width: 50%))
284
285 == 参考文献设置
286 参考文献设置也比较简单，只需在文件尾部加入`#bibliography("example.yml", style: "gb-7714-2015-numeric")`即可。
    yml 格式如下：
287 ```yaml
288 audio-descriptions:
289   affiliated:
290     - names: Taylor, Dallas
291       role: narrator
292   author: Barrows, Miellyn Fitzwater
293   date: 2017-02-07
294   issue: 8
295   parent:
296     author: Taylor, Dallas
297     title: Twenty Thousand Hertz
298     type: Audio
299   title: Audio Descriptions
300   type: Audio
301   url: https://www.20k.org/episodes/audio
302 barb:
303   author: Günther-Haug, Barbara
304   date: 2020
305   language: de-DE
306   location: München
307   publisher: MVG
308   title: Den Boden unter den Füßen verlieren
309   type: Book
310 ```
311
312 在文章适当的地方插入`@audio-descriptions`@audio-descriptions 或`@barb`@barb 这类的键。
313
314 = 写在最后
315 Typst 相对来说还比较新，功能和 latex 相比稍弱，同时还存在一些 bug。如果使用过程中有任何建议或模板上有什么问题，请到
    #link("https://github.com/soarowl/typst.git")提要求。
316
317 #counter(heading).update(0)
318 #set heading(numbering: "A.1")
319 = 附录
320
321 == article 模板
322 #raw(read("article.typ"), lang: "typc", block: true)
323
324 == 本文档源码
325 #raw(read("article_tutor.typ"), lang: "typc", block: true)
326
327 #bibliography("basic.yml", style: "gb-7714-2015-numeric")
328

```

参考文献

- [1] BARROWS M F. Audio Descriptions[Z/OL]. (2017-02-07). <https://www.20k.org/episodes/audio>.
- [2] GÜNTHER-HAUG B. Den Boden unter den Füßen verlieren[M]. München: MVG, 2020.