

R para Ciência de Dados rmarkdown e quarto

Instituto de Matemática e Estatística
Universidade Federal da Bahia

Coordenação: Profa Carolina & Prof Gilberto

Preparando o ambiente

Durante o curso

- Usaremos nas aulas: posit.cloud.
 - Recomendamos instalar e usar R com versão pelo menos 4.1: cran.r-project.org.
 - usaremos o *framework* [tidyverse](https://www.tidyverse.org):
 - Instalação: `install.packages("tidyverse")`
-

Na sua casa

- **IDE** recomendadas: [RStudio](https://www.rstudio.com) e [VSCode](https://code.visualstudio.com).
 - Caso você queira usar o [VSCode](https://code.visualstudio.com), instale a extensão da linguagem R: [REditorSupport](https://marketplace.visualstudio.com/items?itemName=augustobelo.r-editor-support).
- Outras linguagens interessantes: [python](https://www.python.org) e [julia](https://julialang.org).
 - [python](https://www.python.org): linguagem interpretada de propósito geral, contemporânea do R, simples e fácil de aprender.
 - [julia](https://julialang.org): linguagem interpretada para análise de dados, lançada em 2012, promete simplicidade e velocidade.

Onde estudar sozinho

Este curso é apenas o começo!

Você vai ter que estudar sozinho para avançar mais...

Para usar o pacote `rmarkdown`, você precisa ter:

- conhecimento básico da linguagem R
- conhecimento básico da linguagem latex
- conhecimento básico da linguagem markdown

Onde estudar sozinho

R

- Zen do R
- R for Datascience
- ecoR

\LaTeX

- Learn \LaTeX in 30 minutes
- Detexify
- Learn \LaTeX with Wikibooks

markdown

- Tutorial de markdown da Microsoft
- Tutorial de markdown da Mozilla
- markdown Basics
- markdown Live Preview

`rmarkdown`

`rmarkdown`: The Definitive Guide

`quarto`

`quarto`

Pacotes da linguagem R deste curso

- rmarkdown
- blogdown
- bookdown
- readxl
- writexl
- janitor
- patchwork
- prettydoc
- glue
- ggthemes
- gt
- rarticles
- tidyverse
- quarto

A linguagem R

Sobre a linguagem R

A precursora da linguagem R: S.

- R é uma linguagem derivada da S.
 - S foi desenvolvido em fortran por **John Chambers** em 1976 no **Bell Labs**.
 - S foi desenvolvida para realizar análise estatística de dados.
 - Filosofia do S: permitir que usuários possam analisar dados usando estatística com pouco conhecimento de programação.
-

História da linguagem R

- Em 1991, **Ross Ihaka** e **Robert Gentleman** criaram o R na Nova Zelândia.
- Em 1996, **Ross** e **Robert** liberam o R sob a licença “GNU General License”, o que tornou o R um software livre.
- Em 1997, The Core Group é criado para melhorar e controlar o código fonte do R.

markdown

markdown

- Criado em 2004 por **John Gruber**.
- Criado inicialmente para textos para internet.
- **Ideia:** fácil de escrever, fácil de ler e entender o código, e permitir edição em forma de prosa. Foco no conteúdo e não nos detalhes da linguagem.
- `markdown` foi inspirada pela formatação permitida ao escrever e-mails.
- `markdown` é portátil.
- Não depende de versões como Microsoft Word.
- Uso amplamente disseminado, com versões adotadas em aplicativos como: WhatsApp, Notion, GitHub, Stack Overflow, entre outros.

`rmarkdown` and `quarto` usam `pandoc` para converter código `markdown` para os formatos HTML, pdf e docx.

markdown

Sintaxe básica

Seções (e subseções) são partes que dividem um texto de acordo com conteúdo afins.

Para mais detalhes, consulte [Seções e subseções de acordo com NBR 6024/2012](#).

1. SEÇÃO PRIMÁRIA

1.1 SEÇÃO SECUNDÁRIA

1.1.1 Seção Terciária

1.1.1.1 Seção quaternária

1.1.1.1.1 Seção quinária

3. MATERIAIS E MÉTODOS

3.1 ANÁLISE DESCRITIVA:
características dos grupos de estudo

3.1.1 Brasil

3.1.1.1 Grandes Regiões

3.1.1.1.1 Norte

3.1.1.1.2 Nordeste

3.1.1.1.3 Sul

3.1.1.1.4 Sudeste

3.1.1.1.5 Centro-Oeste

3. MATERIAIS E MÉTODOS

Texto, texto, texto, texto,
texto, texto, texto, texto,
texto, texto ...

3.1 ANÁLISE DESCRITIVA:

características dos grupos de estudo

Texto, texto, texto, texto,
texto, texto, texto, texto,
texto, texto ...

3.1.1 Brasil

Texto, texto, texto, texto,
texto, texto, texto, texto,
texto, texto ...

... continua.

markdown

Sintaxe básica

Podemos definir seções e subseções com #.

O caractere # precisa estar na primeira coluna da linha.

É necessário incluir um único espaço depois de #.

código markdown	código HTML	código \LaTeX
# texto	<h1>texto</h1>	\section{texto}
## texto	<h2>texto</h2>	\subsection{texto}
### texto	<h3>texto</h3>	\subsubsection{texto}
#### texto	<h4>texto</h4>	\paragraph{texto}
##### texto	<h5>texto</h5>	\subparagraph{texto}
##### texto	<h6>texto</h6>	

markdown

Sintaxe básica

Parágrafos

Para criar parágrafos, separa blocos de linhas com um (ou mais) linhas em branco.

código markdown	código HTML	código \LaTeX
Primeira linha.	<code><p>Primeira linha.</p></code>	Primeira linha.
Segunda linha.	<code><p>Segunda linha.</p></code>	Segunda linha.

Não inclua tabs ou espaços na primeira linha de parágrafos.

markdown

Sintaxe básica

Formatação de texto

Descrição	código markdown	código HTML	código \LaTeX	Resultado
Itálico	<code>*Itálico*</code>	<code>Itálico</code>	<code>\textit{Itálico}</code>	<i>Itálico</i>
Negrito	<code>**Negrito**</code>	<code>Negrito</code>	<code>\textbf{Negrito}</code>	Negrito
Tachado	<code>~~Tachado~~</code>	<code><s>Tachado</s></code>		Tachado
Sobrescrito	<code>x^2^</code>	<code>x<sup>2</sup></code>		x^2
Subscrito	<code>t~0~</code>	<code>t<sub>0</sub></code>		t_0

Podemos ter um texto em negrito e em itálico:

```
***Negrito e Itálico***
```

markdown

Sintaxe básica

Bloco de citação

Na primeira linha do parágrafo, inclua >.

Exemplo de código

```
> Isto é uma citação.
```

Resultado

Isto é uma citação.

markdown

Sintaxe básica

Bloco de citação com múltiplos parágrafos

Adicione > em cada parágrafo e nos espaços em branco entre os parágrafos.

Exemplo de código

```
> Primeira linha do bloco de citação.  
>  
> Segunda linha do bloco de citação.
```

Resultado

Primeira linha do bloco de citação.
Segunda linha do bloco de citação.

markdown

Sintaxe básica

Bloco de citação pode ter todos os outros elementos markdown.

Exemplo

```
> ### Um lindo bloco de citação
> Primeiro parágrafo tem negrito.
>
> Segundo parágrafo tem Itálico.
>
> Terceiro Párrafo tem Negrito e Itálico.
```

Resultado

*Primeiro parágrafo tem **negrito**.*

*Segundo parágrafo tem *Itálico*.*

*Terceiro Párrafo tem ***Negrito e Itálico***.*

markdown

Sintaxe básica

Listas ordenadas

As listas ordenadas devem começar com o número 1 (ou i.).

código markdown	código HTML	código \LaTeX
1. Primeiro item	<code></code>	<code>\begin{enumerate}</code>
8. Segundo item	<code>Primeiro item</code>	<code>\item Primeiro item</code>
1. Terceiro item	<code>Segundo item Terceiro item</code>	<code>\item Segundo item \item Terceiro item</code>
	<code></code>	<code>\end{enumerate}</code>

Resultado

- 1 Primeiro item
- 2 Segundo item
- 3 Terceiro item

markdown

Sintaxe básica

Listas não ordenadas

As listas não ordenadas começam com: -, * ou +.

código markdown	código HTML	código \LaTeX
+ Primeiro item	<code></code>	<code>\begin{itemize}</code>
+ Segundo item	<code>Primeiro</code>	<code>\item Primeiro</code>
	<code>item</code>	<code>item</code>
+ Terceiro item	<code>Segundo</code>	<code>\item Segundo item</code>
	<code>item</code>	
	<code>Terceiro</code>	<code>\item Terceiro</code>
	<code>item</code>	<code>item</code>
	<code></code>	<code>\end{itemize}</code>

Resultado

- Primeiro item
- Segundo item
- Terceiro item

markdown

Sintaxe básica

Listas aninhadas

Indente as listas que estão dentro de outras listas com dois espaços.

código markdown	código HTML	código L ^A T _E X
+ Item 1		\begin{itemize}
+ Item 1 interno	Item 1	\item Item 1
+ Item 2 interno		\begin{itemize}
+ Item 2	Item 1	\item Item 1 interno
	interno	
	Item 2	\item Item 2 interno
	interno	
		\end{itemize}
	Item 2	\item Item 2
		\end{itemize}

Resultado

- Item 1
 - Item 1
 - Item 2
- Item 2

markdown

Sintaxe básica

Código fonte (sem inclusão do resultado)

Para inclusão de código *inline* (dentro de uma frase): ``print("olá mundo!")``.

Exemplo

Um texto com código ``print(1 + 2)``.

Resultado

Um texto com código `print(1 + 2)`.

Para inclusão de código *em bloco*, use `````.

Exemplo

Podemos substituir `r` por: `python`, `html`, `julia` entre outros.

```
```r
print("Olá mundo!")
print(1 + 2)
```
```

Resultado

```
print("Olá mundo")
print(1 + 2)
```

Para ver a lista de linguagens compatíveis consulte: [linguagens compatíveis com markdown](#).

markdown

Sintaxe básica

Tabelas

markdown usa tabelas conhecidas como pipe table que tem a seguinte sintaxe:

- As colunas são separadas por |.
- A primeira linha contém cabeçalho das colunas.
- A segunda linha contém o alinhamento:
 - ----- (default) valores serão alinhados à esquerda
 - -----: valores serão alinhados à direita
 - :----- valores serão alinhados à esquerda
 - :---: valores ficarão centralizados
- A partir da terceira linha, incluímos as informações
- Incluímos a legenda da tabela depois de incluímos todas as linhas

As células de uma *pipe table* **não** podem conter:

- parágrafos
- listas
- valor em múltiplas linhas

Gerador de tabelas markdown: tablesgenerator.com/markdown_tables.

Exemplo

```
default	alinhamento à esquerda	alinhamento à direita	centralizado
12	12	12	12
123	123	123	123
1	1	1	1
```

: Legenda da tabela.

Resultado

Tabela 7: Legenda da tabela.

| default | alinhamento
à esquerda | alinhamento à
direita | centralizado |
|---------|---------------------------|--------------------------|--------------|
| 12 | 12 | 12 | 12 |
| 123 | 123 | 123 | 123 |
| 1 | 1 | 1 | 1 |

markdown

Sintaxe básica

Links e imagens

A sintaxe básica para links é `[texto do link](endereço do link)`, onde:

- `texto do link` é um texto descritivo para o link
- `endereço do link` é o endereço para redirecionamento

Exemplo:

```
[Google](https://www.google.com.br)
```

Resultado:

[Google](https://www.google.com.br)

Se *texto do link* é igual a *endereço do link*, você pode usar:

```
<https://www.google.com.br>
```

markdown

Sintaxe básica

Links e imagens

A sintaxe básica para links é `![texto da imagem] (endereço da imagem)`, onde:

- texto da imagem é um texto descritivo da imagem
- endereço da imagem é *filename* da imagem

A imagem precisa estar dentro do mesmo diretório que o arquivo `.Rmd`.

Exemplo

```
![Logo da linguagem R] (figuras/r.png)
```

Resultado



markdown

Sintaxe básica

- Vamos analisar um documento simples usando `markdown`.
- Por hora, ignore as primeiras linhas delimitadas `---`.

Exemplo com sintaxe

markdown

Sintaxe básica

Exercício

Use loremipsum.io para criar um texto html.

Inclua:

- o texto precisa ter 6 parágrafos
- uma citação de sua preferência
- texto negrito, texto itálico, texto tachado, e texto tachado e negrito
- inclua uma lista de itens que você gosta
- inclua um link para o website do nosso curso ufba.netlify.app
- inclua o logo rmarkdown ao final do arquivo (o arquivo na pasta figuras)

Inclua a seguinte tabela com as colunas centralizadas.

Tabela 8: Cinco maiores cidades do Brasil.

| Posição | Cidade | Estado |
|---------|----------------|--------|
| 1 | São Paulo | SP |
| 2 | Rio de Janeiro | RJ |
| 3 | Brasília | DF |
| 4 | Salvador | BA |
| 5 | Fortaleza | CE |

Equações usando \LaTeX

Expressões matemáticas usando L^AT_EX

`rmarkdown` e `quarto` usam L^AT_EX para composição tipográfica de equações matemáticas.

Existem dois tipos de inclusão de equações matemáticas:

- *inline*: equação é parte de um parágrafo
 - usamos `$` para equações em modo *inline*
- *display*: equação em um linha separada com texto centralizado
 - usamos `$$` para equações em modo *display*

Expressões matemáticas usando \LaTeX

Equações em modo *inline*

Exemplo:

A equação $e^{i\pi} + 1 = 0$ foi proposta por Euler.

Resultado:

A equação $e^{i\pi} + 1 = 0$ foi proposta por Euler.

Expressões matemáticas usando \LaTeX

Equações em modo *display*

Exemplo:

A seguinte equação foi proposta por Euler:

\$\$

$$e^{i\pi} + 1 = 0$$

\$\$

Resultado:

A seguinte equação foi proposta por Euler:

$$e^{i\pi} + 1 = 0$$

Expressões matemáticas usando \LaTeX

Símbolos e funções importantes

| Descrição | Código \LaTeX | Resultado |
|--------------------------|------------------------|-------------|
| Letra alpha | <code>\alpha</code> | α |
| Letra epsilon | <code>\epsilon</code> | ϵ |
| União de conjuntos | <code>\cup</code> | \cup |
| Intersecção de conjuntos | <code>\cap</code> | \cap |
| Menor (desigualdade) | <code><</code> | $<$ |
| Espaço simples | <code>b\ a</code> | $b\ a$ |
| Espaço duplo | <code>b\quad a</code> | $b\quad a$ |
| Espaço triplo | <code>b\qquad a</code> | $b\qquad a$ |
| Maior (desigualdade) | <code>></code> | $>$ |
| Infinito | <code>\infty</code> | ∞ |
| Logaritmo | <code>\log</code> | \log |
| Multiplicação | <code>\cdot</code> | \cdot |
| Contém | <code>\subset</code> | \subset |
| Contido | <code>\supset</code> | \supset |
| Integral | <code>\int</code> | \int |
| Somatório | <code>\sum</code> | \sum |
| Produtório | <code>\prod</code> | \prod |
| Limite | <code>\lim</code> | \lim |

Lista exhaustiva de símbolos e funções.

Detexfy - aplicativo para descobrir símbolos matemáticos.

Expressões matemáticas usando \LaTeX

Sobrescrito, subscrito e fração

Sobrescrito e subscrito

Muito comum em expressões matemáticas envolvendo expoentes, índices, e em alguns proponentes especiais.

- Sobrescrito e limite superior em \int, \prod, \sum, \cap e \cup : \wedge
- Subscrito e limite inferior em \int, \prod, \sum, \cap e \cup : $\underset{_}$

Se o sobrescrito e subscrito tiver mais de um caracter, envolva a expressão em chaves $\{\}$.

Fração

| Código \LaTeX | Resultado |
|--------------------------|---------------|
| <code>\frac{a}{b}</code> | $\frac{a}{b}$ |

Expressões matemáticas usando \LaTeX

Sobrescrito e subscrito

| Código \LaTeX | Resultado |
|--------------------------------------|---------------------------|
| <code>a^{n_j}</code> | a^{n_j} |
| <code>a_{n^i}</code> | a_{n^i} |
| <code>\int_{-\infty}^{\infty}</code> | $\int_{-\infty}^{\infty}$ |
| <code>\frac{1}{1+x^2} dx</code> | $\frac{1}{1+x^2} dx$ |
| <code>\sum_{i=1}^{\infty}</code> | $\sum_{i=1}^{\infty}$ |
| <code>\frac{1}{i^2}</code> | $\frac{1}{i^2}$ |
| <code>\prod_{i=1}^n i</code> | $\prod_{i=1}^n i$ |
| <code>\cup_{i=1}^{n+12} A_i</code> | $\cup_{i=1}^{n+12} A_i$ |
| <code>\cap_{i=-\infty}^0 B_i</code> | $\cap_{i=-\infty}^0 B_i$ |

Expressões matemáticas usando \LaTeX

Parênteses, colchetes e chaves

| Descrição | Código \LaTeX | Resultado |
|----------------------|-------------------------|-------------------------|
| Parênteses | $(x + y)$ | $(x + y)$ |
| Colchetes | $[x + y]$ | $[x + y]$ |
| Chaves | $\{x + y\}$ | $\{x + y\}$ |
| Produto interno | $\langle x, y \rangle$ | $\langle x, y \rangle$ |
| Valor absoluto | $ x + y $ | $ x + y $ |
| Norma | $\ x + y\ $ | $\ x + y\ $ |
| Arredonda para baixo | $\lfloor x + y \rfloor$ | $\lfloor x + y \rfloor$ |
| Arredonda para cima | $\lceil x + y \rceil$ | $\lceil x + y \rceil$ |

Para aumentar o tamanho de parênteses, colchetes e outros: `\big`, `\Big`, `\bigg`, e `\Bigg`.

Para aumentar o tamanho de parênteses, colchetes e outros: `\left` e `\right` (na mesma linha).

Para detalhes sobre ajuste no tamanho de parênteses, colchetes e outros, consulte [parênteses](#), [colchetes](#) e [outros](#).

Use `split` para quebrar uma equação em várias linhas, e use `&` para alinhamento.

Expressões matemáticas usando L^AT_EX

Parênteses, colchetes e chaves

Exemplo

```
$$  
\begin{split}  
a &= \sum_{i=-\infty}^{\infty} \left\{ a_i + b_i \right. \\ &= \left. c_i + \int_a^b x \cdot i \, dx \right\}  
\end{split}  
$$
```

Resultado

$$\begin{aligned} a &= \sum_{i=-\infty}^{\infty} \left\{ a_i + b_i \right. \\ &= \left. c_i + \int_a^b x \cdot i \, dx \right\} \end{aligned}$$

Expressões matemáticas usando \LaTeX

Exercício

Use a plataforma www.texrendr.com para codificar em \LaTeX as seguintes equações matemáticas:

- $\sum_{i=0}^{\infty} \frac{1}{2^i} = 2$
- $\int_{-\infty}^{\infty} \exp\left(-\frac{x^2}{2}\right) dx = \sqrt{2\pi}$
- $\prod_{i=1}^n \frac{1}{2} = \frac{1}{2^n}$
- $\lfloor 1, 2 \rfloor = 1$
- $\lceil 1, 2 \rceil = 2$
- $\|(1, 2)\|^2 = 5$
-

$$\begin{aligned} \sum_{i=1}^n \alpha^i &= 1 + \alpha + \cdots + \alpha^n \\ &= \frac{1 - \alpha^{n+1}}{1 - \alpha} \end{aligned}$$

Pacote rmarkdown

rmarkdown

Documentos com extensão `.Rmd` ou `.rmd` permitem combinar:

- código fonte (`code` tag in `html` e `verbatim` em $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$)
- resultado de computações
- texto simples em prosa

Documentos com extensão `.Rmd` ou `.rmd` são renderizados (processados) pelo pacote `rmarkdown`.

Gerador de texto aleatório para este curso: loremipsum.io.

Cheatsheet do pacote `rmarkdown`: [cheatsheet do pacote rmarkdown](#).

`rmarkdown` envia o documento `.Rmd` para o pacote `knitr` que converte o documento para `markdown`, e em seguida `pandoc` converte este arquivo `.md` para formato adequado.



rmarkdown

RStudio

- 1 Para criar um arquivo `.Rmd` no IDE Rstudio:
`file > New File > R Markdown.`
- 2 Em seguida, escolha a opção adequada para o seu texto, incluindo:
 - a. *document* (documentos em prosa).
 - b. *presentation* (apresentação).
 - c. *template* (documentos usando *templates*).

Estrutura básica de documentos .Rmd

```
1- ---
2- title: "Um belo documento"
3- date: 2024/08/02
4- output: html_document
5- ---
6-
7- ```{r}
8- #| label: setup
9- #| echo: false
10-
11- library(tidyverse)
12- ```
13-
14- # Um seção
15-
16- Texto texto texto.
17-
18- ```{r}
19- #| label: grafico
20- ggplot(mtcars, aes(cyl)) + geom_bar()
21- ```
22-
23- ## Uma subseção
24-
25- Texto texto texto.
```

rmarkdown

YAML

- Criada em 2001.
- YAML (Yet Another Markup Language) é uma linguagem desenvolvida para armazenar dados.
- Fácil de ler e escrever.
- YAML é delimitada por `---` e sempre está nas primeiras linhas do documento `.Rmd`.
- YAML controla a formatação do documento `.Rmd`.
- YAML usa dupla *chave valor*: `chave: valor`:
 - `valor` não pode ter espaço em branco. Se `valor` tiver espaço em branco, use aspas ou `|`.
 - `chave`: nome do campo

```
title: "Um título que faz sentido"
```

ou

```
title: |  
  Um título que faz sentido
```

valor pode ter subcampos, e usamos indentação para neste subcampo

```
output:  
  html_document:  
    highlight: "haddock"  
    includes:  
      in_header: header.html  
      before_body: before_body.html
```

Podemos usar código R, prefixando o campo com `!r`.

```
date: !r lubridate::today()
```

Campos comuns em YAML: `title`, `date` e `output`.

rmarkdown

YAML

Citações e bibliografia

- rmarkdown usa `bibtex` para incluir referências e citações.
- **Google Scholar** e maioria das revistas científicas incluem citações `bibtex`.
- Para especificar um formato de bibliografia, use CSL (*Citation Style Language*).

```
bibliography: refs.bib  
csl: apa.csl
```

No texto, use @ + identificador da referência no arquivo .bib.

- Citação direta no texto: @wickham2023r.
- Citação entre parênteses: [@wickham2023r].
- Cite apenas o ano entre parênteses: [-@wickham2023r]
- Citação entre parênteses com comentário: [veja @wickham2023r para mais detalhes].
- Separe múltiplas citações com ;: [@wickham2023r; @wickham2019advanced; @xie2016bookdown]

rmarkdown

YAML

Parâmetros

- Valores que passamos para os documentos.
- Útil para produzir documentos em série, onde apenas alguns valores são modificados.
- Valores ficam disponível na lista `params`.

```
params:  
  nome: "Gilberto Pereira Sassi"  
  idade: 22
```

No texto, use ``r params$nome`` e ``r params$idade``.

Geralmente `params` é usado junto com a função `render` do pacote `rmarkdown`.

rmakrdown YAML

Exemplo

```
1 ---
2 title: "Titulo"
3 date: 2024/08/02
4 output:
5   html_document:
6     params:
7       nome: "Nome da pessoa"
8       vinculo: professor
9 ---
10
11 # Primeira seção
12
13 Texto da primeira seção.
14
15 * Nome: `r params$nome`.
16 * Vínculo: `r params$vinculo`.
```

Para produzir documento, use o seguinte código R

```
render(  
  "filename.Rmd",  
  render_options = list(  
    params = list(nome = "Um nome", data = "01/01/1900")  
  )  
)
```

rmarkdown *chunk*

- Permite executar código da linguagem R dentro do texto do documento `.Rmd`.
- Permite incluir o código e o resultados da execução.
- Coloque o código R entre ````${r}``` e ```.`
- Customize o processamento de *chunk* através de `#| chave: valor` nas linhas imediatamente depois de ````${r}```.`

```
```${r}
```

```
#| label: nome_lindo
```

```
inclua seu código aqui
```

```
```
```

Se você incluir a opção `label` em *chunk*, as figuras que eventualmente forem geradas no código do *chunk* serão salvas como `label`.

rmarkdown

opções *chunk* úteis

- `echo`: inclusão de código R no texto
 - Campo booleano (apenas `true` e `false`)
 - Se `true`, o código do *chunk* será incluído no texto
 - Valor default: `true`
- `eval`: avaliação do código R
 - Campo booleano (apenas `true` e `false`)
 - Se `false`, o código do *chunk* não será executado
 - Valor default: `true`
- `results`: determina como o resultado da execução será incluída no texto
 - Valores possíveis: `hide`, `hold`, `asis`, e `markup`
 - `hide`: o resultado da execução não será mostrado no texto
 - `hold`: o resultado da execução será incluído ao final do texto
 - `asis`: o resultado da execução será incluído sem reformatação
 - `markup`: rmarkdown formatará o resultado da execução antes de incluir no texto
 - Valor default: `markup`

- **error**: inclusão de mensagens de erro no texto
 - Campo booleano (apenas true e false)
 - Se true, o erro de execução será incluído no texto
 - Valor default: true
- **message**: inclusão de mensagens no texto
 - Campo booleano (apenas true e false)
 - Se true, a mensagem no texto será incluído no texto
 - Valor default: true
- **warning**: inclusão de mensagens de warning no texto
 - Campo booleano (apenas true e false)
 - Se true, a mensagem no texto será incluído no texto
 - Valor default: true
- **comment**: caractere incluído na primeira coluna de cada linha de resultado
 - Valor do tipo caractere. Precisa incluir aspas.
 - Valor default: "##"
- **highlight**: destaque do código no texto
 - Campo booleano (apenas true e false)
 - Se true, a mensagem será destaque com cores no texto
 - Valor default: true

- `prompt`: adiciona o `>` na primeira coluna em cada linha de código R
 - Se `true`, `>` será incluído na primeira coluna de cada linha de código
 - Valor default: `false`
- `fig.align`: alinhamento de figuras criadas pelo código R no *chunk*
 - Valores possíveis: `left`, `right`, e `center`
 - `left`: alinha figura à esquerda
 - `right`: alinha figura à direita
 - `center`: alinha figura ao centro
 - Valor default: `center`
- `fig.ext`: extensão que as figuras criadas pelo código R no *chunk* serão salvas
 - Valores possíveis: `png`, `jpg`, `pdf`, entre outros
 - Valor default: `NULL`
- `fig.show`: como incluir figuras criadas pelo código R
 - Valores possíveis: `hide`, `hold`, `animate`, e `asis`
 - `hide`: não inclui as figuras no documento
 - `hold`: inclui todas as figuras criadas no final do documento
 - `animate`: combina todas as figuras em uma animação
 - `asis`: inclui as figuras serão inseridas sem formatação na localização do *chunk*
 - Valor default: `asis`

- `out.width` e `out.height`: largura e altura da figura no texto
 - Unidades comumente usadas: %, in, cm, `\\linewidth` (apenas para beamer e pdf)

Para mais opções, consulte: [opções para *chunk*](#).

rmarkdown

exemplo

Exemplo

- Vamos analisar um documento simples usando `rmarkdown`.

Exemplo com sintaxe

rmarkdown

exercício

Exercício

Crie um documento `.Rmd`, com as seguintes especificações:

- No cabeçalho YAML, inclua um campo `title` com o seu nome
- No cabeçalho YAML, inclua um campo `date` com sua data de nascimento
- No cabeçalho YAML, inclua um campo `output: html_document`
- Inclua 3 parágrafos (use loremipsum.io)
- Inclua um chunk que carrega o pacote `tidyverse` e com as opções `echo: true` e `message: false`
- Inclua um chunk que realiza um sumário de `iris` com as opções `comment: "#"` e `prompt: true`

rmarkdown documentos deste curso

Tipos mais comuns de documentos produzidos pelo pacote Rmarkdown:

- `pdf_document`: documentos com extensão `.pdf` (usa \LaTeX para criar o documento)
- `html_document`: documentos com extensão `.html`
- `word_document`: documentos com extensão `.docx`
- `ioslide_document`: apresentações com extensão `.html`
- `beamer_presentation`: apresentações com extensão `.pdf` (usa \LaTeX para criar o documento)
- `html_pretty`: documentos com extensão `html` e formatação *estilosa* (precisa do pacote `prettydoc`)
- `rticles`: documentos com formatação de revistas científicas

Veremos também como usar os seguintes pacotes:

- `blastula`: envio automatizado de emails
- `blogdown`: crie websites estáticos usando R (e [hugo](#))
- `bookdown`: editoração de livros

rmarkdown

pdf_document

rmarkdown

pdf_document

Para criar .pdf, especifique output: pdf_document no cabeçalho YAML.

```
---  
title: "Hello word!"  
date: 01/01/1900  
author: "Fulano de Tal"  
output: pdf_document  
---
```

Neste caso, você pode usar código \LaTeX dentro do texto.

Necessário ter \LaTeX instalado em sua máquina.

rmarkdown

pdf_document

Algumas configurações disponíveis no cabeçalho YAML:

- `toc`: inclusão de sumário
 - Campo booleano (apenas `true` e `false`)
 - Valor default: `false`
- `toc_depth`: limite de nível de # para inclusão no sumário
 - Valores inteiros de 1 a 6
 - Valor default: 2 (# e ## será incluído no sumário)
- `number_sections`: inclusão de numeração nas seções
 - Campo booleano (apenas `true` e `false`)
 - Valor default: `false`
- `fig_caption`: inclusão de legenda na figura
 - Campo booleano (apenas `true` e `false`)
 - Valor default: `true`
- `lang`: especificação da linguagem do documento
 - para português brasileiro use `pt-br`
- `documentclass`: L^AT_EX document class (valor default é `article`)
- `classoption`: opções para document class (`oneside` por exemplo)

- `highlight`: formatação do código incluído no texto
 - Valores possíveis: `default`, `tango`, `pygments`, `kate`, `monochrome`, `espresso`, `zenburn`, `haddock`, `breezedark`, `arrow`, e `rstudio`
 - Valor default: `default`
- `fontsize`: tamanho de fonte
 - Valores possíveis: `10pt`, `11pt`, `12pt`, e outros
 - Valor default: `12pt`
- `linkcolor`: cor para links internos dentro do documento
 - Valores possíveis: consulte [pacote xcolor](#)
- `urlcolor`: cor para link externo dentro do documento
 - Valores possíveis: consulte [pacote xcolor](#)
- `citecolor`: cor para citações dentro do texto
 - Valores possíveis: consulte [pacote xcolor](#)
- `citation_package`: processamento das citações dentro do documento
 - Valores possíveis: `pandoc-citeproc`, `natbib`, e `biblatex`
 - Valor default: `pandoc-citeproc`
- `keep_tex`: mantenha código fonte \LaTeX ?
 - Campo booleano (apenas `true` e `false`. Se `true`, mantenha o código fonte \LaTeX)
 - Valor default: `false`
- `geometry`: opções do [pacote geometry](#)

Podemos incluir código \LaTeX para customização adicional do documento com `includes`:

- `in_header`: inclusão de código no preâmbulo (entre `\documentclass{article}` e `\begin{document}`)
- `before_body`: inclusão de código imediatamente depois `\begin{document}`
- `after_body`: inclusão de código imediatamente antes de `\end{document}`

```
output:  
pdf_document:  
  includes:  
    in_header: preambulo.tex  
    before_body: prefixo.tex  
    after_body: sufixo.tex
```

rmarkdown
pdf_document

Exemplo

Vamos analisar um exemplo!

exemplo-3

rmarkdown

pdf_document

Exercício

Crie um documento chamado `documento.pdf`, e inclua os seguintes campos no cabeçalho YAML:

- `title` - inclua o seguinte título:
 - Documento pdf do curso R para ciência de Dados - rmarkdown e quarto
- `date` - inclua sua data de nascimento
- `lang` - inclua o idioma português brasileiro - `pt-br`
- `fontsize` - formate o texto para 14pt
- `geometry` - inclua margens de 1cm
- `includes` - inclua o pacote `enumerate` como preambulo
- Inclua duas seções, três parágrafos e o seguinte código (com seu resultado) em seu documento:

```
summary(iris)
ggplot(iris) + geom_bar(aes(x = Species))
```

rmarkdown

html_document

rmarkdown html_document

markdown foi desenvolvida para criar documentos html, por isso `html_document` tem o conjunto de características mais ricas.

Para criar documentos `.html`, especifique `output: html_document` no cabeçalho YAML.

```
---  
title: "Um lindo título"  
author: "Fulano de Tal"  
date: 01/01/1900  
output: html_document  
---
```

rmarkdown

html_document

Algumas configurações disponíveis no cabeçalho YAML:

- `toc`: inclusão de sumário
 - Campo booleano (apenas `true` e `false`)
 - Valor default: `false`
- `toc_depth`: limite de nível de # para inclusão no sumário
 - Valores inteiros de 1 a 6
 - Valor default: 2
- `toc_float`: sumário sempre visível?
 - Campo booleano (apenas `true` e `false` - `true` torna o sumário mais visível)
 - Valor default: `false`

- `number_sections`: inclusão de numeração nas seções
 - Campo booleano (apenas `true` e `false` - `true` numera as seções)
 - Valor default: `false`

```
# Primeira seção
```

```
Um parágrafo.
```

```
## Primeira Subseção
```

```
Outro parágrafo.
```

rmarkdown

html_document

Subseções em abas

Todas as subseções serão organizadas em abas se colocarmos `{.tabset}` ao final de uma seção.

- `.tabset-pills`: destaca a aba (seção) ativa

```
# Primeira seção {.tabset .tabset-pills}
```

Parágrafo 1 da primeira seção.

```
## Subseção 1
```

Parágrafo 1 da primeira seção.

```
## Subseção 2
```

Parágrafo 1 da segunda seção.

Primeira seção

Parágrafo 1 da primeira seção.

Subseção 1

Subseção 2

Parágrafo 1 da primeira seção.

rmarkdown html_document

- `theme`: temas disponíveis para documentos html
 - Valores possíveis: `default`, `bootstrap`, `cerulean`, `cosmo`, `darkly`, `flatly`, `journal`, `lumen`, `paper`, `readable`, `sandstone`, `simplex`, `spacelab`, `united`, `yeti`
 - Valor default: `default`
- `highlight`: formatação do código incluído no texto
 - Valores possíveis: `default`, `tango`, `pygments`, `kate`, `monochrome`, `espresso`, `zenburn`, `haddock`, `breezedark`, `arrow`, e `rstudio`
 - Valor default: `default`
- `fig_width` e `fig_height`: largura e altura das figuras no documento html
 - Unidades possíveis: `px`, `in`, `cm`, `%`
- `fig_caption`: inclusão de legendas nas figuras
 - Campo booleano (apenas `true` e `false` - `true` para incluir legendas)
 - Valor default: `false`
- `code_folding`: inclusão de botão de *alternação* entre mostrar (*show*) e esconder (*hide*) o código
 - Valores possíveis: `hide` e `show`

Podemos incluir código html para customização adicional do documento com `includes`:

- `in_header`: inclusão de código html entre `<head>` e `</head>`
- `before_body`: inclusão de código html imediatamente depois de `<body>`
- `after_body`: inclusão de código html imediatamente antes de `</body>`

```
includes:
```

```
  in_header: header.html
```

```
  before_body: before_body.html
```

```
  after_body: after_body.html
```

rmarkdown
html_document

Exemplo

Vamos analisar um exemplo!

Exemplo 4

rmarkdown html_document

Exercício

Crie um documento chamado `documento.html`, e inclua os seguintes campos no cabeçalho YAML:

- `title` - inclua o seguinte título:
 - Documento pdf do curso R para ciência de Dados - rmarkdown e quarto
- `date` - inclua sua data de nascimento
- `number_sections` - numere as seções
- `code_folding` - inclua botões para mostrar e/ou esconder código R
- `includes` - o seguinte código entre `<header>` e `</header>`

```
<meta name="description" content="Olá mundo.">  
<meta name="keywords" content="rmarkdown">  
<meta name="author" content="seu nome">
```

- Inclua duas seções, três parágrafos e o seguinte código (com seu resultado) em seu documento:

```
summary(iris)
ggplot(iris) + geom_bar(aes(x = Species))
```


rmarkdown

word_document

rmarkdown word_document

Use `pdf_document` e `html_document` sempre que possível.

`word_document` é útil por causa do monopólio da Microsoft em produtos de processamento de texto (e planilha).

Para customizar customizações, use *template*:

- Crie um documento simples (usando `rmarkdown`) `template.docx`
- Modifique o estilo e as configurações de margens deste documento
- Modifique o cabeçalho YAML para:

```
1 - - -
2 title: "Título"
3 author: "Gilberto"
4 date: 2024/08/02
5 output: word_document
6 - - -
7
8 # Primeira seção
9
10 ```{r}
11 summary(mtcars)
12 ```
```

rmarkdown
word_document

Vamos analisar um exemplo!

Exemplo 5

rmarkdown

word_document

Crie um documento chamado `documento.docx`, e inclua os seguintes campos no cabeçalho YAML:

- `title` - inclua o seguinte título:
 - Documento word gerado pelo pacote `rmarkdown`
- `date` - inclua sua data de nascimento
- `author` - inclua seu nome
- Inclua um documento de template:
 - Modifique o estilo de *título do documento*:
 - fonte: `times new roman`
 - Negrito e itálico
 - tamanho: `30pt`
- Inclua três seções com dois parágrafos cada.

rmarkdown

outras extensões

Algumas pacotes úteis que usam rmarkdown:

- pacote para automatização de envio de emails: [blastula](#)
- pacote para criação de blogs e websites usando o [gerador de site estatístico](#) – Hugo: [blogdown](#)
- pacote para redação de livros: [bookdown](#)

quarto

quarto

- Próxima geração do pacote rmarkdown desenvolvida por [posit](#) (sucessor de rstudio).
- quarto usa markdown e \LaTeX para produzir documentos.
- quarto permite que usemos código das seguintes linguagens:
 - [python](#)
 - [R](#)
 - [julia](#)
 - [Observable JS](#)
- Se você sabe usar rmarkdown, você sabe usar quarto.
- Para a linguagem R, quarto usa [knitr](#).
- De forma semelhante ao pacote rmarkdown, quarto usa [pandoc](#).
- Os arquivos tem extensão `.qmd`.
- Para criar documento `.pdf`, é necessário tem \LaTeX instalada (por exemplo, [miktex](#) e [texlive](#)).

Para detalhes, consulte [quarto](#).

quarto

html

quarto html

Documentos em formato html.

Para criar, especifique format: html no cabeçalho YAML.

```
---  
title: "Título magnífico"  
author: "Fulano de Tal"  
date: 01/01/1900  
format: html  
---
```

Neste caso, podemos usamos código html no texto.

Algumas configurações disponíveis no cabeçalho YAML.

- `toc`: inclusão de sumário
 - Campo booleano (`true` inclui o sumário)
 - Valor default: `false`
- `toc-depth`: nível de seção para inclusão no sumário
 - Valores possíveis: números inteiros de 1 a 6
 - Valor default: 6
- `toc-location`: localização do sumário. Essa opção pode estar desabilitada dependendo do tema.
 - Valores possíveis: `left`, `right` e `body`
 - Valor default: `right`

- `number-sections`: numeração das seções
 - Campo booleano (`true` - as seções serão numeradas)
 - Valor default: `false`
- `number-depth`: nível máximo de seção para numeração
 - Valores possíveis: números inteiros entre 1 a 6
 - Valor default: 6

Para retirar a numeração de uma seção, use o seguinte `{.unnumbered}`.

```
## Seção {.unnumbered}
```

Você pode esconder todos os códigos (`echo: false`) com a seguintes opções:

```
execute:  
  echo: true # inclusão do código de todos os chunks
```

Valores possíveis: `false`, `true`, e `fenced`. `fenced` mostra o *chunk* completamente incluindo ````${r}```` e `````.

- `embed-resources`: inclusão de figuras, css, js e outros elementos diretamente no código do arquivo html
 - Campo booleano (`true` - inclusão das figuras e outros elementos no código)
 - Valor default: `false`
- `anchor-sections`: link para seções do documento
 - Campo booleano (`true` - inclusão de links para as seções)
 - Valor default: `false`
- `link-external-icon`: mostre um ícone para indicar que o link é externo.
 - Campos booleano (`true` - inclusão do ícone nos links externos)
 - Valor default: `false`

- `link-external-newwindow`: abrir o link externo em uma nova aba?
 - Campo booleano (`true` - o link é aberto em nova aba)
 - Valor default: `false`
- `include-in-header`: inclusão de código `html` imediatamente antes de `</header>`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta
- `include-before-body`: inclusão de código `html` imediatamente depois de `<body>`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta
- `include-after-body`: inclusão de código `html` imediatamente antes de `</body>`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta
- `code-fold`: botão para mostrar/esconder o código
 - Valores possíveis:
 - `false`: não inclui o botão
 - `true`: inclui o botão com o código escondido
 - `show`: inclui o botão com o código a mostra
 - Valor default: `false`
- `code-summary`: nome para inclusão no botão
 - Valor especial: texto entre aspas

```
code-tools:  
  source: true  
  toggle: true  
  caption: "Nome do botão"
```

or

```
code-tools: true
```

`code-tools` inclui o botão para esconder ou mostrar o código de todos os *chunks*.

Subopções de `code-tools`:

- `source`: mostrar o código da página? `true` - inclui o botão para mostrar o código da página.
- `toggle`: mostrar o botão de mostrar/esconder o código de todos os *chunks*? `true` - mostra o botão para mostrar o código.
- `caption`: nome do botão. Valores possíveis: `none` ou texto.

- `code-copy`: inclusão de botão de copiar.
 - Valores possíveis:
 - `hover` - inclusão do botão de copiar ao passar o mouse em cima - default
 - `true` - inclusão do botão de copiar
 - `false` - nunca inclua o botão de copiar
- `code-line-number`: numeração nos blocos de código.
 - Campo booleano (`true` - inclua o numeração nos blocos de código)
 - Valor default: `false`
- `theme`: tema da página. Por padrão, quarto usa *Bootstrap 5*.
 - Valores possíveis: [temas disponíveis para documentos html](#)
- `highlight-style`: tema para formatação dos blocos de código.
 - Valores possíveis: [temas disponíveis para formatação dos blocos de código](#)

Tabela 13: Algumas opções de formatação.

Código	Descrição	Exemplo
max-width	Largura máxima da página	max-width: 1400px
mainfont	Fonte do documento	mainfont: monospace
fontsize	Tamanho da fonte	fontsize: 12px
fontcolor	Cor da fonte (letras)	fontcolor: #ff9522
monofont	Fonte das linhas de códigos	monofont: math
linestretch	Correponde a propriedade CSS line-height	linestretch: 1.7
backgroundcolor	Cor do fundo do documento	background: "#9596b4"
margin-*	Correponde a propriedade CSS margin	margin-top: 1em

* pode ser: left, right, top, e bottom.

quarto
html

Vamos analisar um exemplo!

Exemplo

quarto html

Crie um documento chamado `documento.html`, e inclua:

- `title` - inclua o seguinte título: Documento html
- `date` - inclua a sua data de nascimento
- inclua três seções com texto *dummy*
- inclua o sumário e deixe ele a esquerda
- numere as seções
- inclua figuras, css, js e outros elementos diretamente no código
- mude o tema do documento html
- muda o estilo de formatação dos blocos de código
- mude o tamanho da fonte
- mude a largura da linha
- mude a fonte do documento

quarto

pdf

quarto pdf

Necessário ter \LaTeX instalado (por exemplo, `miktex` e `texlive`).

Para criar, especifique `format: pdf` no cabeçalho YAML.

```
---  
title: "Título do documento .pdf"  
author: "Fulano de Tal"  
date: 01/01/1900  
format: pdf  
---
```

Neste caso, podemos usar código \LaTeX no texto.

Opções semelhantes ao formato `output: pdf_document` do pacote `rmarkdown`.

Verifique as opções em: `opções para format: pdf`.

quarto pdf

- `cite-method`: método de produção de citação
 - Valor possíveis: `biblatex`, `natbib`, e `citeproc`
 - Valor default: `citeproc`
- `biblio-title`: texto da seção de bibliografia
- `include-in-header`: inclusão de código latex imediatamente antes de `\begin{document}`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta
- `include-before-body`: inclusão de código latex imediatamente depois de `\begin{document}`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta
- `include-after-body`: inclusão de código latex imediatamente antes de `\end{document}`
 - Valor possíveis: *filename* indicando a localização do arquivo dentro da pasta

quarto

pdf

- pdf-engine: aplicativo usado para gerar os documentos
 - Valores possíveis: xelatex, pdflatex, lualatex, tectonic, latexmk, context, wkhtmltopdf, prince, weasyprint, e pdffroff
 - Valor default: xelatex
- lof: inclua a lista de figuras
 - Campo booleano (true - inclusão da lista de figuras)
 - Valor default: false
- lot: inclua a lista de tabelas
 - Campos booleano (true - inclusão da lista de tabelas)
 - Valor default: false

quarto
pdf

Vamos analisar um exemplo!

exemplo-11

Exercício

Crie um documento chamado `documento.pdf`, e inclua os seguintes campos no cabeçalho YAML:

- `title` - inclua o seguinte título: “Documento pdf gerado com quarto”
- `date` - inclua sua data de nascimento
- `lang` - inclua o idioma português brasileiro - pt-br
- `fontsize` - formate o texto para 12pt
- inclua 3 cm de margem superior e a esquerda, e 2 cm para margem inferior e a direita
- inclua o pacote `enumerate` como preâmbulo
- Inclua duas seções, três parágrafos e o seguinte código (com seu resultado) em seu documento:

```
summary(iris)
ggplot(iris) + geom_bar(aes(x = Species))
```

quarto

docx

quarto docx

Use `format: pdf` e `format: html` sempre que possível.

`format: docx` é útil por causa do monopólio da Microsoft em produtos de processamento de texto (e planilha).

Para customizar customizações, use `template`:

- Crie um documento simples `template.docx`:

```
quarto pandoc -o relatorio/template.docx \  
  --print-default-data-file reference.docx
```

- Modifique o estilo e as configurações de margens deste documento
- Modifique o cabeçalho YAML para:

```
---  
title: "Um título lindo"  
author: "fulnao de tal"  
date: 01/01/1900  
output:  
  docx:  
    reference-doc: template.docx  
---
```

quarto
docx

Vamos analisar um exemplo!

Exemplo

quarto docx

Crie um documento chamado `documento.docx`, e inclua os seguintes campos no cabeçalho YAML:

- `title` - inclua o seguinte título: Documento word usando quarto
- `date` - inclua sua data de nascimento
- `author` - inclua seu nome
- Inclua um documento de template:
 - Modifique o estilo de título do documento:
 - fonte: times new roman
 - Negrito e itálico
 - tamanho: 30pt
- Inclua três seções com um parágrafo cada.

quarto
website

quarto website

Crie um website *estático* usando temas *Bootstrap*.

RStudio local

- No RStudio (localmente): New Project > Quarto Website
 - Escolha a localização do código do Website em Browse
 - Dê um nome ao Website: Directory Name
-

RStudio na nuvem

- Instale o pacote quarto: `install.packages('quarto')`.
 - Carregue o pacote quarto: `library(quarto)`.
 - Crie um projeto: `quarto_create_project('Nome do site', type="website")`.
 - Mova os arquivos dentro da pasta para a raiz do projeto.
-

Arquivos `.qmd` com format: `html` serão novas páginas do website.

quarto website

Você pode carregar o website nas plataformas netlify.com e github.com.

Netlify

- 1 Faça login em netlify.com.
- 2 Clique em *sites* na barra lateral.
- 3 Arraste a pasta `_site` até o quadrado em destaque como apresentado na figura abaixo.



Figura 2: Colocando o seu website on-line usando netlify.

Em configurações do website, você pode atualizar o nome do site.

GitHub Pages

- 1 Faça login em github.com.
- 2 Crie um repositório público chamado `<username>.github.io`.
 - Se o *username* é `fulano-tal`, você criará o repositório `fulano-tal.github.io`.
- 1 Adicione os arquivos dentro da pasta `_site` neste repositório.
- 2 Depois de alguns minutos `⌘⌘⌘`, o arquivo está disponível em `https://<username>.github.io`.
 - Se o *username* é `fulano-tal` e o arquivo tem nome `arquivo.html`, o arquivo está on-line em `https://fulano-tal.github.io`.

Quarto Pub

- 1 Faça login em quartopub.com
- 2 No terminal do RStudio, digite: `quarto publish quarto-pub.`
- 3 Responda as perguntas interativas no terminal.

Outras opções de hospedagem: [publishing](#).

quarto website

Exercício:

- Crie um site simples com `quarto_create_project` (não edite por enquanto).
- Hospede o website na plataforma <https://quartopub.com>.

quarto configuração do website

Todas as configurações do website estão em `_quarto.yml`.

```
project:
  type: website

website:
  title: "Título"
  navbar: # barra de navegação
    left:
      - text: Home
        href: index.qmd # home do website

format:
  html:
    lang: pt-BR
    theme: cosmo
    css: styles.css
    toc: true
```

quarto configuração do website

Opções para website:

- title: "Título" - título do website no navbar
- page-footer - rodapé com duas opções:
 - center: "texto markdown" - pode ser left e right no lugar de center
 - border: true - inclusão de borda no rodapé
- reader-mode: true - inclusão do botão de modo leitura
- site-url: <https://site.com> - endereço de hospedagem
- favicon: imagem.png: ícone que aparece na aba do navegador
- search: true - inclusão do menu de busca
- navbar: páginas que serão incluídas na barra de navegação
 - logo: imagem.png - imagem que vai aparecer ao do título na barra de navegação
 - right: (ou left ou center) liste as páginas com duplas de href e text

Opções para website:

Podemos colocar um barra de navegação lateral com sidebar: com as seguintes opções:

- logo: imagem.png - logo acima da barra de navegação lateral
- contents: referência das páginas
 - section: divisão de páginas por seções
 - incluímos páginas com duplas href e text

Opções para format:

```
format:  
  html:  
    lang: pt-BR  
    theme:  
      light: flatly # escolher tema claro  
      dark: darkly # escolher tema escuro  
css: styles.css  
toc: true  
toc-title: Título
```


quarto
website

Vamos analisar um exemplo:

Exemplo

quarto website

Crie sua página pessoal com três páginas:

- 1 Home: descrição pessoal
- 2 Curriculum vitae: currículo
- 3 Contato: links das suas redes sociais
- 4 Inclua um nota de rodapé
- 5 Hospede na plataforma <https://quartopub.com>