# Package 'pagedown'

January 7, 2025

**Type** Package

**Title** Paginate the HTML Output of R Markdown with CSS for Print

**Version** 0.22

**Description** Use the paged media properties in CSS and the JavaScript library 'paged.js' to split the content of an HTML document into discrete pages. Each page can have its page size, page numbers, margin boxes, and running headers, etc. Applications of this package include books, letters, reports, papers, business cards, resumes, and posters.

**Depends** R (>= 3.5.0)

**Imports** rmarkdown (>= 2.13), bookdown (>= 0.8), htmltools, jsonlite, later (>= 1.0.0), processx, servr (>= 0.31), httpuv, xfun, websocket

**Suggests** promises, testit, xaringan, pdftools, revealjs, covr, xml2

**License** MIT + file LICENSE

**URL** https://github.com/rstudio/pagedown

**BugReports** https://github.com/rstudio/pagedown/issues

**SystemRequirements** Pandoc (>= 2.2.3)

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** Yihui Xie [aut, cre] (<https://orcid.org/0000-0003-0645-5666>), Romain Lesur [aut, cph] (<https://orcid.org/0000-0002-0721-5595>), Christophe Dervieux [ctb] (<https://orcid.org/0000-0003-4474-2498>), Brent Thorne [aut] (<https://orcid.org/0000-0002-1099-3857>), Xianying Tan [aut] (<https://orcid.org/0000-0002-6072-3521>), Atsushi Yasumoto [ctb] (<https://orcid.org/0000-0002-8335-495X>), Posit Software, PBC [cph, fnd], Adam Hyde [ctb] (paged.js in resources/js/), Min-Zhong Lu [ctb] (resume.css in resources/css/), Zulko [ctb] (poster-relaxed.css in resources/css/)

**Maintainer** Yihui Xie <xie@yihui.name>

**Repository** CRAN

**Date/Publication** 2025-01-07 15:20:11 UTC

# Contents



---

`book_crc` *Create a book for Chapman & Hall/CRC*

---

## Description

This output format is similar to `html_paged`. The only difference is in the default stylesheets.

## Usage

```
book_crc(..., css = c("crc-page", "default-page", "default", "crc"))
```

## Arguments

| | |
|---|---|
| `..., css` | Arguments passed to `html_paged()`. |

## Value

An R Markdown output format.

## business_card *Create business cards*

### Description

This output format is based on an example in the Github repo https://github.com/RelaxedJS/ReLaXed-examples. See https://pagedown.rbind.io/business-card/ for an example.

### Usage

```
business_card(template = pkg_resource("html", "card.html"))
```

### Arguments

| | |
|---|---|
| template | The path to the Pandoc template to convert Markdown to HTML. |

### Value

An R Markdown output format.

### Examples

```
pagedown::business_card()
```

## chrome_print *Print a web page to PDF or capture a screenshot using the headless Chrome*

### Description

Print an HTML page to PDF or capture a PNG/JPEG screenshot through the Chrome DevTools Protocol. Google Chrome or Microsoft Edge (or Chromium on Linux) must be installed prior to using this function.

### Usage

```
chrome_print(
  input,
  output = xfun::with_ext(input, format),
  wait = 2,
  browser = "google-chrome",
  format = c("pdf", "png", "jpeg"),
  options = list(),
  selector = "body",
  box_model = c("border", "content", "margin", "padding"),
  scale = 1,
```

```
  work_dir = tempfile(),
  timeout = 30,
  extra_args = c("--disable-gpu"),
  verbose = 0,
  async = FALSE,
  outline = gs_available(),
  encoding
)
```

## Arguments

| | |
|---|---|
| input | A URL or local file path to an HTML page, or a path to a local file that can be rendered to HTML via `rmarkdown::render()` (e.g., an R Markdown document or an R script). If the `input` is to be rendered via `rmarkdown::render()` and you need to pass any arguments to it, you can pass the whole `render()` call to `chrome_print()`, e.g., if you need to use the `params` argument: `pagedown::chrome_print(rmarkdown::params = list(foo = 1:10)))`. This is because `render()` returns the HTML file, which can be passed to `chrome_print()`. |
| output | The output filename. For a local web page '`foo/bar.html`', the default PDF output is '`foo/bar.pdf`'; for a remote URL '`https://www.example.org/foo/bar.html`', the default output will be '`bar.pdf`' under the current working directory. The same rules apply for screenshots. |
| wait | The number of seconds to wait for the page to load before printing (in certain cases, the page may not be immediately ready for printing, especially there are JavaScript applications on the page, so you may need to wait for a longer time). |
| browser | Path to Google Chrome, Microsoft Edge or Chromium. This function will try to find it automatically via `find_chrome()` if the path is not explicitly provided and the environment variable `PAGEDOWN_CHROME` is not set. |
| format | The output format. |
| options | A list of page options. See `https://chromedevtools.github.io/devtools-protocol/tot/Page#met` for the full list of options for PDF output, and `https://chromedevtools.github.io/devtools-protoc` for options for screenshots. Note that for PDF output, we have changed the defaults of `printBackground` (`TRUE`), `preferCSSPageSize` (`TRUE`) and when available `transferMode` (`ReturnAsStream`) in this function. |
| selector | A CSS selector used when capturing a screenshot. |
| box_model | The CSS box model used when capturing a screenshot. |
| scale | The scale factor used for screenshot. |
| work_dir | Name of headless Chrome working directory. If the default temporary directory doesn't work, you may try to use a subdirectory of your home directory. |
| timeout | The number of seconds before canceling the document generation. Use a larger value if the document takes longer to build. |
| extra_args | Extra command-line arguments to be passed to Chrome. |
| verbose | Level of verbosity: `0` means no messages; `1` means to print out some auxiliary messages (e.g., parameters for capturing screenshots); `2` (or `TRUE`) means all messages, including those from the Chrome processes and WebSocket connections. |

| | |
|---|---|
| async | Execute chrome_print() asynchronously? If TRUE, chrome_print() returns a promise value (the **promises** package has to be installed in this case). |
| outline | If not FALSE, chrome_print() will add the bookmarks to the generated pdf file, based on the table of contents informations. This feature is only available for output formats based on html_paged. It is enabled by default, as long as the Ghostscript executable can be detected by find_gs_cmd. |
| encoding | Not used. This argument is required by RStudio IDE. |

## Value

Path of the output file (invisibly). If async is TRUE, this is a promise value.

## References

https://developer.chrome.com/blog/headless-chrome/

---

# find_chrome *Find Google Chrome, Microsoft Edge or Chromium in the system*

---

## Description

On Windows, this function tries to find Chrome or Edge from the registry. On macOS, it returns a hard-coded path of Chrome under '/Applications'. On Linux, it searches for chromium-browser and google-chrome from the system's *PATH* variable.

## Usage

```
find_chrome()
```

## Value

A character string.

---

# html_letter *Create a letter in HTML*

---

## Description

This output format is similar to html_paged. The only differences are in the default stylesheets and the default value of the fig_caption parameter which is set to FALSE. See https://pagedown.rbind.io/html-letter/ for an example.

## Usage

```
html_letter(..., css = c("default", "letter"), fig_caption = FALSE)
```

### Arguments

`...`, `css`, `fig_caption`
: Arguments passed to `html_paged()`.

### Value

An R Markdown output format.

---

`html_paged` *Create a paged HTML document suitable for printing*

---

### Description

This is an output format based on `bookdown::html_document2` (which means you can use those Markdown features added by **bookdown**). The HTML output document is split into multiple pages via a JavaScript library **paged.js**. These pages contain elements commonly seen in PDF documents, such as page numbers and running headers.

### Usage

```
html_paged(
  ...,
  css = c("default-fonts", "default-page", "default"),
  theme = NULL,
  template = pkg_resource("html", "paged.html"),
  csl = NULL,
  front_cover = NULL,
  back_cover = NULL
)
```

### Arguments

`...`
: Arguments passed to `bookdown::html_document2`.

`css`
: A character vector of CSS and Sass file paths. If a path does not contain the '`.css`', '`.sass`', or '`.scss`' extension, it is assumed to be a built-in CSS file. For example, `default-fonts` means the file`pagedown:::pkg_resource('css', 'default-fonts.css')`. To see all built-in CSS files, run `pagedown:::list_css()`.

`theme`
: The Bootstrap theme. By default, Bootstrap is not used.

`template`
: The path to the Pandoc template to convert Markdown to HTML.

`csl`
: The path of the Citation Style Language (CSL) file used to format citations and references (see the Pandoc documentation).

`front_cover`, `back_cover`
: Paths or urls to image files to be used as front or back covers. Theses images are available through CSS variables (see Details).

### Details

When a path or an url is passed to the `front_cover` or `back_cover` argument, several CSS variables are created. They are named `--front-cover` and `--back-cover` and can be used as value for the CSS property `background-image`. For example, `background-image: var(--front-cover);`. When a vector of paths or urls is used as a value for `front_cover` or `back_cover`, the CSS variables are suffixed with an index: `--front-cover-1`, `--front-cover-2`, etc.

### Value

An R Markdown output format.

### References

https://pagedown.rbind.io

---

## html_resume *Create a resume in HTML*

---

### Description

This output format is based on Min-Zhong Lu's HTML/CSS in the Github repo https://github.com/mnjul/html-resume. See https://pagedown.rbind.io/html-resume/ for an example.

### Usage

```
html_resume(
  ...,
  css = "resume",
  template = pkg_resource("html", "resume.html"),
  number_sections = FALSE,
  fig_caption = FALSE
)
```

### Arguments

`...`, `css`, `template`, `number_sections`, `fig_caption`
See `html_paged()`.

### Value

An R Markdown output format.

## jss_paged *Create an article for the Journal of Statistical Software*

### Description

This output format is similar to `html_paged`.

### Usage

```
jss_paged(
  ...,
  css = c("jss-fonts", "jss-page", "jss"),
  template = pkg_resource("html", "jss_paged.html"),
  csl = pkg_resource("csl", "journal-of-statistical-software.csl"),
  highlight = NULL,
  pandoc_args = NULL
)
```

### Arguments

`...`, `css`, `template`, `csl`, `highlight`, `pandoc_args`
: Arguments passed to `html_paged()`.

### Value

An R Markdown output format.

## poster_relaxed *Create posters in HTML*

### Description

The output format `poster_relaxed()` is based on an example in the Github repo https://github.com/RelaxedJS/ReLaXed-examples. See https://pagedown.rbind.io/poster-relaxed/ for an example.

The output format `poster_jacobs()` mimics the style of the "Jacobs Landscape Poster LaTeX Template Version 1.0" at https://www.overleaf.com/gallery/tagged/poster. See https://pagedown.rbind.io/poster-jacobs/ for an example.

### Usage

```
poster_relaxed(
  ...,
  css = "poster-relaxed",
  template = pkg_resource("html", "poster-relaxed.html"),
  number_sections = FALSE
)

poster_jacobs(
  ...,
  css = "poster-jacobs",
  template = pkg_resource("html", "poster-jacobs.html")
)
```

### Arguments

`...`, `css`, `template`, `number_sections`
: See `html_paged()`.

### Value

An R Markdown output format.

---

## thesis_paged *Create a paged HTML thesis document suitable for printing*

---

### Description

This output format is similar to `html_paged`. The only difference is in the default stylesheets and Pandoc template. See https://pagedown.rbind.io/thesis-paged/ for an example.

### Usage

```
thesis_paged(
  ...,
  css = c("thesis"),
  template = pkg_resource("html", "thesis.html")
)
```

### Arguments

`...`, `css`, `template`
: Arguments passed to `html_paged()`.

### Value

An R Markdown output format.

# Index