Version: 8.x

Routes

Most routes within Gotenberg are designed to accept *multipart/form-data* requests and generate one or more PDF files. This guide will assist you in understanding their usage.

Convert with Chromium

The next routes leverage the capabilities of the Chromium browser to effectively transform a diverse range of HTML documents into PDFs.

Checkout the Chromium module configuration to tailor the Chromium behavior to your needs.

URL into PDF route

This *multipart/form-data* route converts a web page into PDF.

```
POST /forms/chromium/convert/url
```

It accepts the following specific form field:

Key	Description	
url	URL of the page you want to convert into PDF.	required

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
```

```
-o my.pdf
```

200 OK | 400 Bad Request | 403 Forbidden | 503 Service Unavailable

```
Content-Disposition: attachement; filename={output-filename.pdf}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.
- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

HTML file into PDF route

This *multipart/form-data* route converts an HTML file into PDF.

```
POST /forms/chromium/convert/html
```

It accepts the following specific form file:

Key	Description	
index.html	The HTML file to convert into PDF.	required

For instance:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>My PDF</title>
  </head>
```

```
  <body>
    <h1>Hello world!</h1>
  </body>
</html>
```

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/html \
--form files=@/path/to/index.html \
-o my.pdf
```

200 OK | 400 Bad Request | 503 Service Unavailable

```
Content-Disposition: attachement; filename={output-filename.pdf}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.
- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

You may also send additional files, like images, fonts, stylesheets, and so on.

The only requirement is that their paths in the `index.html` file are on the root level.

For instance, this will work:

```
<!DOCTYPE html>
<html lang="en">
  <head>
```

```
    <meta charset="utf-8" />
    <title>My PDF</title>
  </head>
  <body>
    <h1>Hello world!</h1>
    <img src="img.png" />
  </body>
</html>
```

But this won't:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>My PDF</title>
  </head>
  <body>
    <h1>Hello world!</h1>
    <img src="/foo/img.png" />
  </body>
</html>
```

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/html \
--form files=@/path/to/index.html \
--form files=@/path/to/img.png \
-o my.pdf
```

> **INFO**
>
> Remote paths for images, fonts (e.g., Google Fonts), etc., work too.

Markdown file(s) into PDF route

This *multipart/form-data* route converts Markdown file(s) into PDF.

```
POST /forms/chromium/convert/markdown
```

It accepts the following specific form files:

Key	Description	
index.html	The HTML file that wraps the markdown content.	required
*.md	At least one markdown file.	required

It works like the HTML route but with access to a Go template function `toHTML`. This function converts a markdown file's content into HTML.

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>My PDF</title>
  </head>
  <body>
    {{ toHTML "file.md" }}
  </body>
</html>
```

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/markdown \
--form files=@/path/to/index.html \
--form files=@/path/to/file.md \
-o my.pdf
```

200 OK | 400 Bad Request | 503 Service Unavailable

```
Content-Disposition: attachement; filename={output-filename.pdf}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.
- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

Page Properties

Each route accepts the following form fields:

Key	Description	Default
paperWidth	Paper width, in inches.	8.5
paperHeight	Paper height, in inches.	11
marginTop	Top margin, in inches.	0.39
marginBottom	Bottom margin, in inches.	0.39
marginLeft	Left margin, in inches.	0.39
marginRight	Right margin, in inches.	0.39
preferCssPageSize	Define whether to prefer page size as defined by CSS.	false
printBackground	Print the background graphics.	false

Key	Description	Default
omitBackground	Hide the default white background and allow generating PDFs with transparency.	false
landscape	Set the paper orientation to landscape.	false
scale	The scale of the page rendering.	1.0
nativePageRanges	Page ranges to print, e.g., '1-5, 8, 11-13' - empty means all pages.	All pages

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form paperWidth=8.27 \
--form paperHeight=11.7 \
--form marginTop=1 \
--form marginBottom=1 \
--form marginLeft=1 \
--form marginRight=1 \
--form preferCssPageSize=false \
--form printBackground=true \
--form omitBackground=true \
--form landscape=true \
--form scale=2.0 \
--form nativePageRanges=1-5 \
-o my.pdf
```

INFO

Examples of paper size (width x height):

- `Letter` - 8.5 x 11 (default)
- `Legal` - 8.5 x 14

- `Tabloid` - 11 x 17
- `Ledger` - 17 x 11
- `A0` - 33.1 x 46.8
- `A1` - 23.4 x 33.1
- `A2` - 16.54 x 23.4
- `A3` - 11.7 x 16.54
- `A4` - 8.27 x 11.7
- `A5` - 5.83 x 8.27
- `A6` - 4.13 x 5.83

INFO

The rules regarding the `printBackground` and `omitBackground` form fields are the following:

- If `printBackground` is set to *false*, no background is printed.
- If `printBackground` is set to *true*:
  - If the HTML document has a background, that background is used.
  - If not:
    - If `omitBackground` is set to *true*, the default background is transparent.
    - If not, the default white background is used.

Header & Footer

Each route accepts the following form files:

Key	Description	Default
header.html	HTML file containing the header.	None
footer.html	HTML file containing the footer.	None

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form files=@/path/to/header.html \
--form files=@/path/to/footer.html \
-o my.pdf
```

Each of them has to be a **complete HTML document**:

```
<html>
<head>
    <style>
    body {
        font-size: 12px;
        margin: auto 20px;
    }
    </style>
</head>
<body>
<p><span class="pageNumber"></span> of <span class="totalPages"></span>
</p>
</body>
</html>
```

INFO

The following classes allow you to inject printing values:

- `date` - formatted print date.
- `title` - document title.
- `url` - document location.
- `pageNumber` - current page number.
- `totalPages` - total pages in the document.

CAUTION

There are some limitations:

- No JavaScript.
- The CSS properties are independent of the ones from the HTML document.
- `footer.html` CSS properties override the ones from `header.html`.
- Only fonts installed in the Docker image are loaded - see the fonts configuration section.
- Images only work using a base64 encoded source - i.e., `...`.
- `background-color` and `color` CSS properties require an additional `-webkit-print-color-adjust: exact` CSS property in order to work.
- Assets are not loaded (i.e., CSS files, scripts, fonts, etc.).
- Background form fields do not apply.

Wait Before Rendering

Each route accepts the following form fields:

Key	Description	Default
waitDelay	Duration (e.g, '5s') to wait when loading an HTML document before converting it into PDF.	None
waitForExpression	The JavaScript expression to wait before converting an HTML document into PDF until it returns *true*.	None

INFO

These form fields do not work if JavaScript is disabled via `--chromium-disable-javascript`.

See the Chromium module configuration for more details.

waitDelay

When the page relies on JavaScript for rendering, and you don't have access to the page's code, you may want to wait a certain amount of time to make sure Chromium has fully

rendered the page you're trying to generate.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form waitDelay=5s \
-o my.pdf
```

waitForExpression

A more reliable option than the previous form field:

```
// Somewhere in the HTML document.
var globalVar = 'notReady'
await promises()
window.globalVar = 'ready'
```

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form 'waitForExpression=window.status === '\''ready'\''' \
-o my.pdf
```

Emulated Media Type

Each route accepts the following form field:

Key	Description	Default
emulatedMediaType	The media type to emulate, either "screen" or "print" - empty means "print".	print

Some websites have dedicated CSS rules for print. Using "screen" allows you to force the "standard" CSS rules.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form emulatedMediaType=screen \
-o my.pdf
```

Custom HTTP Headers

Each route accepts the following form field:

Key	Description	Default
extraHttpHeaders	HTTP headers to send by Chromium while loading the HTML document (JSON format).	None

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form 'extraHttpHeaders={"X-Header":"value"}' \
-o my.pdf
```

Invalid HTTP Status Codes

Each route accepts the following form field:

Key	Description	Default
failOnHttpStatusCodes	Return a *409 Conflict* response if the HTTP status code from the main page is not acceptable.	[499,599]

A `X99` entry means every HTTP status codes between `X00` and `X99` (e.g., 499 means every HTTP status codes between 400 and 499).

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://httpstat.us/400 \
--form 'failOnHttpStatusCodes=[499]'
```

409 Conflict

```
Content-Type: text/plain; charset=UTF-8
Gotenberg-Trace: {trace}
Body:

Invalid HTTP status code from the main page: 400: Bad Request
```

Console Exceptions

Each route accepts the following form field:

Key	Description	Default
failOnConsoleExceptions	Return a *409 Conflict* response if there are exceptions in the Chromium console.	false

> **INFO**
>
> This form field does not work if JavaScript is disabled via `--chromium-disable-javascript`.
>
> See the Chromium module configuration for more details.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/html \
--form files=@/path/to/index.html \
--form failOnConsoleExceptions=true
```

409 Conflict

```
Content-Type: text/plain; charset=UTF-8
Gotenberg-Trace: {trace}
Body:

Chromium console exceptions:

exception "Uncaught" (17:10): Error: Exception 1
at file:///tmp/db09d2c8-31e3-4058-9923-c2705350f2b3/index.html:18:11;
exception "Uncaught" (20:10): Error: Exception 2
at file:///tmp/db09d2c8-31e3-4058-9923-c2705350f2b3/index.html:21:11:
```

Performance Mode

Each route accepts the following form field:

Key	Description	Default
skipNetworkIdleEvent	Do not wait for Chromium network to be idle.	false

Gotenberg, by default, waits for the network idle event to ensure that the majority of the page is rendered during conversion. However, this often significantly slows down the conversion process. Setting this form field to true can greatly enhance the conversion speed.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/html \
--form files=@/path/to/index.html \
--form skipNetworkIdleEvent=true \
-o my.pdf
```

PDF/A & PDF/UA

Each route accepts the following form fields:

Key	Description	Default
pdfa	Convert the resulting PDF into the given PDF/A format.	None
pdfua	Enable PDF for Universal Access for optimal accessibility.	false

At present, the following PDF/A formats are available:

- `PDF/A-1b`
- `PDF/A-2b`
- `PDF/A-3b`

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--form url=https://my.url \
--form pdfa=PDF/A-1b \
--form pdfua=true \
-o my.pdf
```

Screenshots `route`

You can capture full-page screenshots using the following three routes, which function similarly to their PDF equivalents:

```
POST /forms/chromium/screenshot/url
```

```
POST /forms/chromium/screenshot/html
```

```
POST /forms/chromium/screenshot/markdown
```

These routes accept the following form fields:

Key	Description	Default
format	The image compression format, either "png", "jpeg" or "webp".	png
quality	The compression quality from range 0 to 100 (jpeg only).	100
omitBackground	Hide the default white background and allow generating screenshots with transparency.	false
optimizeForSpeed	Define whether to optimize image encoding for speed, not for resulting size.	false

The following features are also available:

- Wait Before Rendering
- Emulated Media Type
- Custom HTTP headers
- Invalid HTTP Status Codes
- Console Exceptions
- Performance Mode

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/screenshot/html \
--form files=@/path/to/index.html \
--form format=jpeg \
--form quality=100 \
--form optimizeForSpeed=true \
-o my.jpeg
```

Convert with LibreOffice

The next route leverage the capabilities of LibreOffice to effectively transform a diverse range of Office documents (Word, Excel, PowerPoint, etc.) into PDFs.

Checkout the LibreOffice module configuration to tailor the LibreOffice behavior to your needs.

Office documents into PDFs `route`

This *multipart/form-data* route convert one or more Office documents into PDF.

```
POST /forms/libreoffice/convert
```

Currently, the following extensions are supported:

`.bib` `.doc` `.xml` `.docx` `.fodt` `.html` `.ltx` `.txt` `.odt` `.ott` `.pdb` `.pdf` `.psw` `.rtf` `.sdw` `.stw` `.sxw` `.uot` `.vor` `.wps` `.epub` `.png` `.bmp` `.emf` `.eps` `.fodg` `.gif` `.jpg` `.met` `.odd` `.otg` `.pbm` `.pct` `.pgm` `.ppm` `.ras` `.std` `.svg` `.svm` `.swf` `.sxd` `.sxw` `.tiff` `.xhtml` `.xpm` `.fodp` `.potm` `.pot` `.pptx` `.pps` `.ppt` `.pwp` `.sda` `.sdd` `.sti` `.sxi` `.uop` `.wmf` `.csv` `.dbf` `.dif` `.fods` `.ods` `.ots` `.pxl` `.sdc` `.slk` `.stc` `.sxc` `.uos` `.xls` `.xlt` `.xlsx` `.tif` `.jpeg` `.odp` `.odg` `.dotx` `.xltx`

cURL

```
curl \
--request POST http://localhost:3000/forms/libreoffice/convert \
--form files=@/path/to/file.docx \
-o my.pdf
```

cURL

```
curl \
--request POST http://localhost:3000/forms/libreoffice/convert \
--form files=@/path/to/file.docx \
--form files=@/path/to/file.xlsx \
-o my.zip
```

200 OK **400 Bad Request** **503 Service Unavailable**

```
Content-Disposition: attachement; filename={output-filename.ext}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.

- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

Page Properties

The route also accepts the following form fields:

Key	Description	Default
landscape	Set the paper orientation to landscape.	false
nativePageRanges	Page ranges to print, e.g., '1-4' - empty means all pages.	All pages

> **INFO**
>
> If multiple files are provided, the page ranges will be applied independently to each file.

cURL

```
curl \
--request POST http://localhost:3000/forms/libreoffice/convert \
--form files=@/path/to/file.docx \
--form landscape=true \
--form nativePageRanges=1-5 \
-o my.pdf
```

Merge

The route also accepts the following form fields:

Key	Description	Default
merge	Merge the resulting PDFs.	false

cURL

```
curl \
--request POST http://localhost:3000/forms/libreoffice/convert \
--form files=@/path/to/file.docx \
--form files=@/path/to/file.xlsx \
--form merge=true \
-o my.pdf
```

PDF/A & PDF/UA

The route also accepts the following form fields:

Key	Description	Default
pdfa	Convert the resulting PDF into the given PDF/A format.	None
pdfua	Enable PDF for Universal Access for optimal accessibility.	false

At present, the following PDF/A formats are available:

- `PDF/A-1b`
- `PDF/A-2b`
- `PDF/A-3b`

cURL

```
curl \
--request POST http://localhost:3000/forms/libreoffice/convert \
--form files=@/path/to/file.docx \
--form pdfa=PDF/A-1b \
--form pdfua=true \
-o my.pdf
```

Convert into PDF/A & PDF/UA `route`

This *multipart/form-data* route transform one or more PDF files into the requested PDF/A format and/or PDF/UA.

```
POST /forms/pdfengines/convert
```

It accepts the following form files and form fields:

Key	Description	Default
*.pdf	At least one PDF file.	required
pdfa	Convert the resulting PDF into the given PDF/A format.	None*
pdfua	Enable PDF for Universal Access for optimal accessibility.	false*

Note that at least one of the form field must be provided.*

At present, the following PDF/A formats are available:

- `PDF/A-1b`
- `PDF/A-2b`
- `PDF/A-3b`

cURL

```
curl \
--request POST http://localhost:3000/forms/pdfengines/convert \
--form files=@/path/to/pdf.pdf \
--form pdfa=PDF/A-1b \
--form pdfua=true \
-o my.pdf
```

cURL

```
curl \
--request POST http://localhost:3000/forms/pdfengines/convert \
--form files=@/path/to/pdf1.pdf \
--form files=@/path/to/pdf2.pdf \
--form files=@/path/to/pdf3.pdf \
--form files=@/path/to/pdf4.pdf \
--form pdfa=PDF/A-1b \
--form pdfua=true \
-o my.zip
```

200 OK | 400 Bad Request | 503 Service Unavailable

```
Content-Disposition: attachement; filename={output-filename.ext}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.
- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

Merge PDFs `route`

This *multipart/form-data* route accepts PDF files and merges them alphabetically.

```
POST /forms/pdfengines/merge
```

Key	Description	Default
*.pdf	PDF files.	required
pdfa	Convert the resulting PDF into the given PDF/A format.	None
pdfua	Enable PDF for Universal Access for optimal accessibility.	false

At present, the following PDF/A formats are available:

- `PDF/A-1b`
- `PDF/A-2b`
- `PDF/A-3b`

cURL

```
curl \
--request POST http://localhost:3000/forms/pdfengines/merge \
--form files=@/path/to/pdf1.pdf \
--form files=@/path/to/pdf2.pdf \
--form files=@/path/to/pdf3.pdf \
--form files=@/path/to/pdf4.pdf \
-o my.pdf
```

200 OK **400 Bad Request** **503 Service Unavailable**

```
Content-Disposition: attachement; filename={output-filename.pdf}
Content-Type: {content-type}
Content-Length: {content-length}
Gotenberg-Trace: {trace}
Body: {output-file}
```

- See the Request Tracing section for more information about the `Gotenberg-Trace` header.

- See the Output Filename section for more information about the `Gotenberg-Output-Filename` header.

Health Check `route`

```
GET /health
```

cURL

```
curl --request GET http://localhost:3000/health
```

200 OK 503 Service Unavailable

```
{
    "status": "up",
    "details": {
        "chromium": {
            "status": "up",
            "timestamp": "2021-07-01T08:05:14.603364Z"
        },
        "uno": {
            "status": "up",
            "timestamp": "2021-07-01T08:05:14.603364Z"
        }
    }
}
```

The `details` entry gathers the health checks from modules:

- The Chromium module checks if the Chromium browser is heallthy.
- The LibreOffice module checks if the LibreOffice instance is healthy.

Metrics `route`

This route exposes the metrics from other modules using the Prometheus format.

```
GET /prometheus/metrics
```

Currently, the metrics include:

Metric	Description
{namespace}_chromium_requests_queue_size	Current number of Chromium conversion requests waiting to be treated.
{namespace}_chromium_restarts_count	Current number of Chromium restarts.
{namespace}_libreoffice_requests_queue_size	Current number of LibreOffice conversion requests waiting to be treated.
{namespace}_libreoffice_restarts_count	Current number of LibreOffice restarts.

See the Prometheus module configuration for more information.

Request Tracing

A trace, or request ID, serves to identify a specific request in the logs.

By default, the API assigns a unique UUID trace to every request. However, you also have the option to specify the trace for each request using the `Gotenberg-Trace` header.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--header 'Gotenberg-Trace: debug' \
--form url=https://my.url \
-o my.pdf
```

The API also incorporates a `Gotenberg-Trace` header into each response. If you're utilizing the webhook feature, this header will also be added to each request made to your callbacks.

> **INFO**
>
> The `--api-trace-header` flag allows you to configure the header key. See the API module configuration for more details.

Output Filename

By default, for *multipart/form-data* endpoints, the API generates a response with a UUID filename. However, you have the option to specify the filename for each request using the `Gotenberg-Output-Filename` header.

cURL

```
curl \
--request POST http://localhost:3000/forms/chromium/convert/url \
--header 'Gotenberg-Output-Filename: my_filename' \
--form url=https://my.url \
-O -J
```

> **INFO**
>
> The API automatically appends the file extension, so there's no need for you to set it manually.

Edit this page