html-pdf-node DT

1.0.8 • Public • Published 3 years ago

Readme

Code Beta

4 Dependencies

20 Dependents

9 Versions

html-pdf-node

> Pagination plugin for converts html or url to pdf

Note: This plugin will convert html page or public URL into PDF. This will work with Node.js

Installation

```
npm install html-pdf-node
```

Usage

To convert HTML page to PDF using generatePdf method:

```
var html_to_pdf = require('html-pdf-node');
```

```
let options = { format: 'A4' };
// Example of options with args //
// let options = { format: 'A4', args: ['--no-sandbox', '--disable-se

let file = { content: "<h1>Welcome to html-pdf-node</h1>" };
// or //
let file = { url: "https://example.com" };
html_to_pdf.generatePdf(file, options).then(pdfBuffer => {
  console.log("PDF Buffer:-", pdfBuffer);
});
```

html_to_pdf.generatePdf ([file], [options], [callback])

Parameters

`file` <**Object**> File object should have one of the following properties:

- `url` <string> Any public url of the PDF.
- `content` <string> HTML file content of the PDF.

`options` <**Object**> Options object should have the following properties:

- `args` <Array<string>> Additional arguments to pass to the browser instance. The list of Chromium flags can be found here. This options will overwrite the default arguments. The default arguments are `['--no-sandbox', '--disable-setuid-sandbox']`.
- `path` <string> The file path to save the PDF to. If `path` is a relative path, then it is resolved to current working directory. If no path is provided, the PDF won't be saved anywhere.
- `scale` <number> Scale of the webpage rendering. Defaults to `1`. Scale amount must be between 0.1 and 2.
- `displayHeaderFooter` <boolean> Display header and footer. Defaults to `false`.
- `headerTemplate` <string> HTML template to print the header. Should be valid HTML markup with following classes used to inject printing values into them:
  - `date` formatted date
  - `title` document title
  - `url` document location
  - `pageNumber` current page number
  - `totalPages` total pages in the document
- `footerTemplate` <string> HTML template to print the footer. Should use the same format as the `headerTemplate`.
- `printBackground` <boolean> Print background graphics. Defaults to `false`.
- `landscape` <boolean> Paper orientation. Defaults to `false`.

- `pageRanges` <string> Paper ranges to print, e.g., '1-5, 8, 11-13'. Defaults to the empty string, which means print all pages.
- `format` <string> Paper format. If set, takes priority over `width` or `height` options. Defaults to 'Letter'.
- `width` <string|number> Paper width, accepts values labeled with units.
- `height` <string|number> Paper height, accepts values labeled with units.
- `margin` <Object> Paper margins, defaults to none.
  - `top` <string|number> Top margin, accepts values labeled with units.
  - `right` <string|number> Right margin, accepts values labeled with units.
  - `bottom` <string|number> Bottom margin, accepts values labeled with units.
  - `left` <string|number> Left margin, accepts values labeled with units.
- `preferCSSPageSize` <boolean> Give any CSS `@page` size declared in the page priority over what is declared in `width` and `height` or `format` options. Defaults to `false`, which will scale the content to fit the paper size.

Return value

Promise which resolves with PDF buffer.

html_to_pdf.generatePdfs ([files], [options], [callback])

Parameters

`files` <**Array**<**Object**>> File object should have one of the follwing properties:

- `url` <string> Any public url of the PDF.
- `content` <string> HTML file content of the PDF.

`options` <**Object**> Options object should have the following properties:

- `args` <Array<string>> Additional arguments to pass to the browser instance. The list of Chromium flags can be found here. This options will overwrite the default arguments. The default arguments are `['--no-sandbox', '--disable-setuid-sandbox']`.
- `path` <string> The file path to save the PDF to. If `path` is a relative path, then it is resolved to current working directory. If no path is provided, the PDF won't be saved anywhere.
- `scale` <number> Scale of the webpage rendering. Defaults to `1`. Scale amount must be between 0.1 and 2.
- `displayHeaderFooter` <boolean> Display header and footer. Defaults to `false`.
- `headerTemplate` <string> HTML template to print the header. Should be valid HTML markup with following classes used to inject printing values into them:
  - `date` formatted date
  - `title` document title
  - `url` document location

    - `pageNumber` current page number
    - `totalPages` total pages in the document
- `footerTemplate` <string> HTML template to print the footer. Should use the same format as the `headerTemplate` .
- `printBackground` <boolean> Print background graphics. Defaults to `false` .
- `landscape` <boolean> Paper orientation. Defaults to `false` .
- `pageRanges` <string> Paper ranges to print, e.g., '1-5, 8, 11-13'. Defaults to the empty string, which means print all pages.
- `format` <string> Paper format. If set, takes priority over `width` or `height` options. Defaults to 'Letter'.
- `width` <string|number> Paper width, accepts values labeled with units.
- `height` <string|number> Paper height, accepts values labeled with units.
- `margin` <Object> Paper margins, defaults to none.
    - `top` <string|number> Top margin, accepts values labeled with units.
    - `right` <string|number> Right margin, accepts values labeled with units.
    - `bottom` <string|number> Bottom margin, accepts values labeled with units.
    - `left` <string|number> Left margin, accepts values labeled with units.
- `preferCSSPageSize` <boolean> Give any CSS `@page` size declared in the page priority over what is declared in `width` and `height` or `format` options. Defaults to `false` , which will scale the content to fit the paper size.

Return value

Promise which resolves with array of object which contains file objects with PDF buffers.

Example:

```
let options = { format: 'A4' };
let file = [{ url: "https://example.com", name: 'example.pdf' }];

html_to_pdf.generatePdfs(file, options).then(output => {
  console.log("PDF Buffer:-", output); // PDF Buffer:- [{url: "https:
});
```

Keywords

html pdf nodejs puppeteer handlebars

Install

```
> npm i html-pdf-node
```

Repository

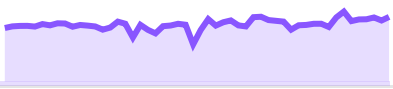
github.com/mrafiqk/html-pdf-node

Homepage

github.com/mrafiqk/html-pdf-node#readme

Weekly Downloads

29,702



Version	License
1.0.8	ISC

Unpacked Size	Total Files
56.9 kB	6

Issues	Pull Requests
64	23

Last publish

3 years ago

Collaborators

Try on RunKit

Report malware

Support

Help

Advisories

Status

Contact npm

Company

About

Blog

Press

Terms & Policies

Policies

Terms of Use

Code of Conduct

Privacy