Open in app

Aayush Ostwal Follow
Jul 20, 2020 · 3 min read

HTML and Text to PDF using Python

The Data science world has evolved with visualization techniques, but it's of no use if we can not generate reports at the end of the day. We can generate patterns but now its time to store them in the more representable form for the coming generations.

Photo by Isaac Smith on Unsplash

2

A more data we are gathering, more efficient ways are needed to visualize and display

During my internship at an Edu-tech startup, we need to automatize the process of report generation. We were equipped with all the plots, graphs, CSV tables that we need to present in that report but do not have a tool that binds all of them.

Anyone into typesetting or LaTeX or tools like “wkhtmltopdf”, “pdfkit”, they know that they can be hard to operate by web devs. This is especially worse when it comes to dynamic content like user customizable papers, background graphics.

There is another easy way of generating pdf that is using pdf crowd which provides APIs and costs a lot for an individual or small scale startups. We can also use ANGULAR or JAVA, but they too require APIs which are costly too.

But, there is a solution.

PDFGEN

Pdfgen is an adapted version of pdfkit which uses wkhtmltopdf but very easy to use, of course, more efficient and free of cost. It makes use of the Google Puppeteer API to utilize Chromium to fulfill the task.

Installation and Requirements

The installation is very easy using pip, we need just to type the following command in the Python terminal

```
pip install pdfgen
```

The requirements are

- Node.js (>=7)
- GNU Make (development) (>=4.2.1)

Usage

As stated about its usage, just a two-line code can covert webpages, HTML files (string format) and text files to pdfs

Open in app

```
    ## FOR WEBITES
    await pdfgen.from_url('http://google.com', 'out.pdf')

    ## FOR HTML FILES
    await pdfgen.from_file('test.html', 'out.pdf')

    ## FOR TEXT FILES
    await pdfgen.from_string('Hello!', 'out.pdf')
```

The file is saved at the same location where the code file is stored but we can also change the path output file just by providing the path.

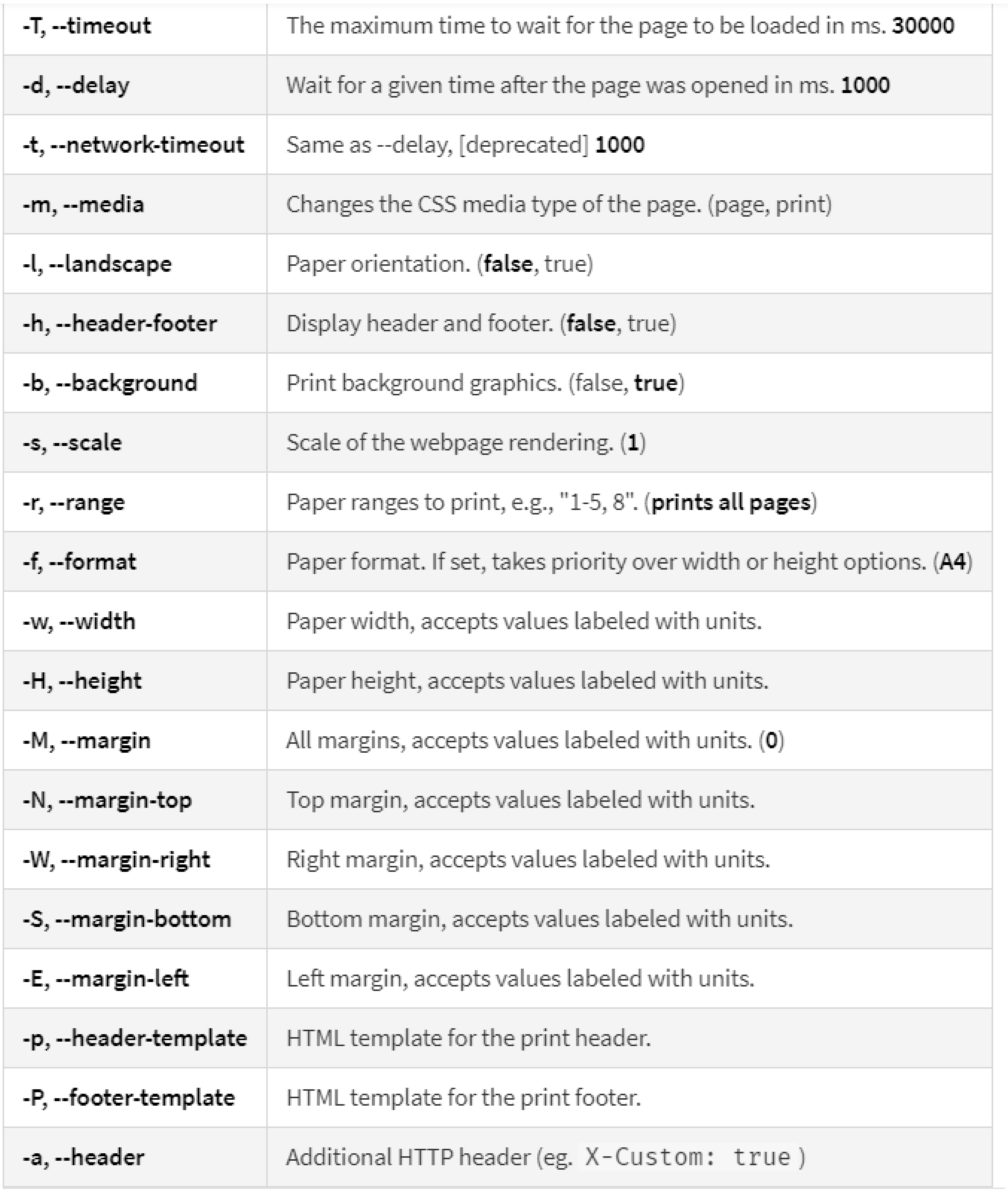
```
import pdfgen

path = 'Path where you want to store output file'
await pdfgen.from_string('Hello!', path)
```

Options

Pdfgen has lots of options for customization which makes PDFs more customizable and that the reason this library performs much up to the expectations than any other libraries dealing with file conversion.

> *This is a list of options that Pdfgen provide the user:*



-T, --timeout	The maximum time to wait for the page to be loaded in ms. **30000**
-d, --delay	Wait for a given time after the page was opened in ms. **1000**
-t, --network-timeout	Same as --delay, [deprecated] **1000**
-m, --media	Changes the CSS media type of the page. (page, print)
-l, --landscape	Paper orientation. (**false**, true)
-h, --header-footer	Display header and footer. (**false**, true)
-b, --background	Print background graphics. (false, **true**)
-s, --scale	Scale of the webpage rendering. (**1**)
-r, --range	Paper ranges to print, e.g., "1-5, 8". (**prints all pages**)
-f, --format	Paper format. If set, takes priority over width or height options. (**A4**)
-w, --width	Paper width, accepts values labeled with units.
-H, --height	Paper height, accepts values labeled with units.
-M, --margin	All margins, accepts values labeled with units. (**0**)
-N, --margin-top	Top margin, accepts values labeled with units.
-W, --margin-right	Right margin, accepts values labeled with units.
-S, --margin-bottom	Bottom margin, accepts values labeled with units.
-E, --margin-left	Left margin, accepts values labeled with units.
-p, --header-template	HTML template for the print header.
-P, --footer-template	HTML template for the print footer.
-a, --header	Additional HTTP header (eg. `X-Custom: true`)

Source: https://www.npmjs.com/package/pdfgen

It just not provide wide ranges of options but also provides a variety of inputs for each option, such as

- A0: 33.1in x 46.8in
- A2: 16.5in x 23.4in
- A4: 8.27in x 11.7in
- Legal: 8.5in x 14in
- Ledger: 17in x 11in
- A1: 23.4in x 33.1in
- A3: 11.7in x 16.5in
- A5: 5.83in x 8.27in

and details about other options can be found at:

> *https://www.npmjs.com/package/pdfgen#format-options*

And these options can be used as

```
## Initializing an option dictionary
options = {
    'format': 'Letter',         ## format of the paper
    'margin': {                 ## margin in the pdf
        'top': '0.75in',
        'right': '0.75in',
        'bottom': '0.75in',
        'left': '0.75in',
    },
    'pageRanges': '1-5,8',      ## pages which needed to be printed
}

## "option" used in sytax :
pdfgen.sync.from_url('http://google.com', 'out.pdf', options=options)
```

Just a Little more appreciation

Pdfgen is an awesome library to use and will simplify the tasks of many in a variety of fields. As a closing statement, I would like to say

> [illegible]