



Search projects



[Help](#)

[Sponsors](#)

[Log in](#)

[Register](#)

pdfgen 1.0.5



Latest version

`pip install pdfgen`

Released: Aug 12, 2020

Pyppeteer-based async python wrapper to convert html to pdf

Navigation

[Project description](#)

[Release history](#)

[Download files](#)

Project links

[Homepage](#)

Statistics

GitHub statistics:

[Stars: 9](#)

[Forks: 2](#)

[Open issues/PRs: 3](#)

Project description

PDFGen-Python: HTML to PDF wrapper

build passing pypi package 1.0.5 python 3.6 | 3.7 | 3.8

Python 3.6.1+ async wrapper for [Pyppeteer](#) to convert HTML to PDF.



NOTE: All the public API functions are adapted to async coroutines, so use them with `await`!

This is adapted version of [python-PDFKit](#) library, so big thanks to them!

Installation

- Install pdfgen:

```
$ pip install pdfgen
```

View statistics for this project via [Libraries.io](#) , or by using [our public dataset on Google BigQuery](#) 

Meta

License: MIT License (MIT)

Author: [Shivansh Saini](#)

Requires: Python >=3.6.1, <4.0.0

Maintainers



[shivanshs9](#)

Classifiers

License

- [OSI Approved :: MIT License](#)

Programming Language

- [Python](#)
- [Python :: 3](#)
- [Python :: 3.6](#)
- [Python :: 3.7](#)
- [Python :: 3.8](#)

Topic

- [Text Processing](#)
 - [Text Processing :: General](#)
 - [Text Processing :: Markup](#)
 - [Text Processing :: Markup :: HTML](#)
 - [Text Processing :: Markup :: XML](#)
 - [Utilities](#)
-

- To download Chromium beforehand, run `$ pdfgen-setup`. Otherwise, it'll be downloaded on the first run of library.

Usage

For simple async tasks:

```
import pdfgen

async def f():
    await pdfgen.from_url('http://google.com', 'out.pdf')
    await pdfgen.from_file('test.html', 'out.pdf')
    await pdfgen.from_string('Hello!', 'out.pdf')
```

Sync API is also provided at `pdfgen.sync` for all the above-mentioned functions:

```
import pdfgen

pdfgen.sync.from_url('http://google.com', 'out.pdf')
pdfgen.sync.from_file('test.html', 'out.pdf')
pdfgen.sync.from_string('Hello!', 'out.pdf')
```

You can pass a list with multiple URLs or files:

```
pdfgen.sync.from_url(['google.com', 'yandex.ru', 'engadget.com']
pdfgen.sync.from_file(['file1.html', 'file2.html'], 'out.pdf')
```

Also you can pass an opened file:

```
with open('file.html') as f:
    pdfgen.sync.pdfgen(f, 'out.pdf')
```

If you wish to further process generated PDF, you can read it to a variable:

```
# Ignore output_path parameter to save pdf to a variable
pdf = pdfgen.sync.from_url('http://google.com')
```

You can specify all [Pyppeteer options](#) used for saving PDF as shown below:



Amazon Web Services is a Sustainability sponsor of the Python Software Foundation.

PSF Sponsor · Served ethically

```
options = {
    'scale': 2.0,
    'format': 'Letter',
    'margin': {
        'top': '0.75in',
        'right': '0.75in',
        'bottom': '0.75in',
        'left': '0.75in',
    },
    'pageRanges': '1-5,8',
}
```

```
pdfgen.sync.from_url('http://google.com', 'out.pdf', options=options)
```

You can also pass any options through meta tags in your HTML:

```
body = """
<html>
  <head>
    <meta name="pdfgen-format" content="Legal"/>
    <meta name="pdfgen-landscape" content="False"/>
  </head>
  Hello World!
</html>
"""

pdfgen.sync.from_string(body, 'out.pdf')
```

Configuration

Each API call takes an optional options parameter to configure print PDF behavior. However, to reduce redundancy, one can certainly set default configuration to be used for all API calls. It takes the configuration options as initial parameters. The available options are:

- `options` - the dict used by default for pypeteer `page.pdf(options)` call. `options` passed as argument to API call will take precedence over the default options.
- `meta_tag_prefix` - the prefix for `pdfgen` specific meta tags - by default this is `pdfgen-`.
- `environ` - the dict used to provide env variables to pypeteer headless browser.

```
import pdfgen

pdfgen.configuration(options={'format': 'A4'})

async def f():
    # The resultant PDF at 'output_file' will be in A4 size and
    await pdfgen.from_string(html_string, output_file, options=
```

Troubleshooting

- `InvalidSourceError`: Provided HTML source is invalid (maybe wrong URL or non-existing file)



Help

Installing packages [↗](#)
Uploading packages [↗](#)
User guide [↗](#)
FAQs

About PyPI

PyPI on Twitter [↗](#)
Infrastructure dashboard [↗](#)
Package index name retention [↗](#)
Our sponsors

Contributing to PyPI

Bugs and feedback
Contribute on GitHub [↗](#)
Translate PyPI [↗](#)
Development credits [↗](#)

Using PyPI

Code of conduct [↗](#)
Report security issue
Privacy policy [↗](#)
Terms of use

Status: All Systems Operational [↗](#)

Developed and maintained by the Python community, for the Python community.
Donate today!

[Switch to desktop version](#)

› [English](#) [español](#) [français](#) [日本語](#) [português \(Brasil\)](#) [українська](#) [Ελληνικά](#) [Deutsch](#) [中文 \(简体\)](#) [中文 \(繁體\)](#) [русский](#)
[עברית](#) [esperanto](#)

FACEBOOK


AWS
Cloud computing

Datadog
Monitoring

**Facebook /
Instagram**
PSF Sponsor

Fastly
CDN

Google
Object Storage and
Download Analytics

Huawei
PSF Sponsor



Microsoft
PSF Sponsor



NVIDIA
PSF Sponsor

Pingdom
Monitoring

Salesforce
PSF Sponsor

Sentry
Error logging

StatusPage
Status page