django-puppeteer-pdf Documentation

Release 0.1.3

Namespace

Feb 28, 2018

Contents

1 Quickstart 3

2 Contribute 5

3 Contents 7

`django-puppeteer-pdf` allows a Django site to output dynamic PDFs. It utilises the puppeteer library, allowing you to write using the technologies you know - HTML and CSS - and output a PDF file.

CHAPTER 1

Quickstart

```
pip install django-puppeteer-pdf
```

Install the puppeteer-pdf cli puppeteer-pdf for your platform.

`settings.py`

```
INSTALLED_APPS = (
    # ...
    'puppeteer_pdf',
    # ...
)
```

`urls.py`

```
from django.conf.urls.defaults import url, patterns
from puppeteer_pdf.views import PDFTemplateView

urlpatterns = patterns('',
    url(r'^pdf/$', PDFTemplateView.as_view(template_name='my_template.html',
                                           filename='my_pdf.pdf'), name='pdf'),
)
```

CHAPTER 2

Contribute

You can fork the project on Github.

CHAPTER 3

Contents

3.1 Full Installation Notes

3.1.1 Installing the Package

From PyPI

```
pip install django-puppeteer-pdf
```

From source

```
git clone git://github.com/namespace-ee/django-puppeteer-pdf.git
cd django-puppeteer-pdf
python setup.py install
```

3.1.2 Installing the puppeteer-pdf

Find the relevant version of the `puppeteer-pdf` downloads page.

3.1.3 Setting up your Django

Add `puppeteer_pdf` to your `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    # ...
    'puppeteer_pdf',
    # ...
)
```

By default it will try to execute the `puppeteer-pdf` command from your `PATH`.

If you can't add `puppeteer-pdf` to your `PATH` or you want to use some other version, you can use the `PUPPETEER_PDF_CMD` setting:

```
PUPPETEER_PDF_CMD = '/path/to/my/puppeteer-pdf'
```

3.1.4 Display static files

Set `STATIC_ROOT` in your `settings.py`:

```
STATIC_ROOT = '/full/path/to/static/directory/'
```

Make sure your static files and directories are inside this directory.

Note: In production static files are supposed to reside outside the project folder, in a public directory. The STATIC_ROOT-setting gives the path to this directory. However, django-puppeteer-pdf requires that STATIC_ROOT is also set on your local machine.

In development the static files reside in their respective apps folder or in a cross-app directory defined by the STATIC_DIRS-setting. Refer to the django documentation for how you can move static files to the STATIC_ROOT directory through a django script.

3.2 Usage

The `PDFTemplateView` is a Django class-based view. By default, it uses `PDFTemplateResponse` to render an HTML template to PDF. It accepts the following class attributes:

`template_name` The full name of a template to use as the body of the PDF.

`header_template` Optional. The full name of a template to use as the header on each page.

`footer_template` Optional. The full name of a template to use as the footer on each page.

`filename` The filename to use when responding with an attachment containing the PDF. Default is `'rendered_pdf.pdf'`.

If `None`, the view returns the PDF output inline, not as an attachment.

`response_class` The response class to be returned by `render_to_response()` method. Default is `PDFTemplateResponse`.

`html_response_class` The response class to be returned by `render_to_response()` method, when rendering as HTML. See note below. Default is `TemplateResponse`.

`cmd_options` The dictionary of command-line arguments passed to the underlying `puppeteer-pdf` binary. Default is `{}`.

puppeteer-pdf options can be found by running `puppeteer-pdf --help`. Unfortunately they don't provide hosted documentation.

Note: For convenience in development you can add the GET arg `?as=html` to the end of your URL to render the PDF as a web page.

3.2.1 Simple Example

Point a URL at `PDFTemplateView`:

```
from django.conf.urls.defaults import *
from puppeteer_pdf.views import PDFTemplateView


urlpatterns = patterns('',
    # ...
    url(r'^pdf/$', PDFTemplateView.as_view(template_name='my_template.html',
                                           filename='my_pdf.pdf'), name='pdf'),
    # ...
)
```

3.2.2 Advanced Example

Point a URL (as above) at your own view that subclasses `PDFTemplateView` and override the sections you need to.

```
from puppeteer_pdf.views import PDFTemplateView


class MyPDF(PDFTemplateView):
    filename = 'my_pdf.pdf'
    template_name = 'my_template.html'
    cmd_options = {
        'margin-top': 3,
    }
```

3.2.3 Unicode characters

Templates containing utf-8 characters should be supported. You will need to ensure that you set the content type in your template file for *puppeteer-pdf* to interpret it properly.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

3.3 Settings

3.3.1 Available settings

Here's a full list of available settings, in alphabetical order, and their default values.

PUPPETEER_PDF_CMD

Default: `'puppeteer-pdf'`

The name of the `puppeteer-pdf` binary.

If there are no path components, this app will look for the binary using the default OS paths.

PUPPETEER_PDF_CMD_OPTIONS

Default: `{'encoding': 'utf8', 'quiet': True}`

A dictionary of command-line arguments to pass to the `puppeteer-pdf` binary. Keys are the name of the flag and values are arguments for the flag.

To pass a simple flag, for example: `puppeteer-pdf --landscape`:

```
PUPPETEER_PDF_CMD_OPTIONS = {'landscape': True}
```

To pass a flag with an argument, for example: `wkhtmltopdf --marginTop '100px'`:

```
PUPPETEER_PDF_CMD_OPTIONS = {'marginTop': '100px'}
```

PUPPETEER_PDF_DEBUG

Default: same as `settings.DEBUG`

A boolean that turns on/off debug mode.