Templated-docs Documentation

Release 0.3.1

Alex Morozov

October 17, 2016

Contents

Templated-docs is a Django package that allows you to generate documents, including texts, spreadsheets and presentations, from templates. It utilises LibreOffice as an underlying conversion document mechanism.

CHAPTER 1

Features

- Generate any LibreOffice-supported document from within Django.
- Use Django template language inside office documents.
- Create custom generation management commands to integrate with other systems.

CHAPTER 2

Installation

To install templated-docs:

1. Make sure you have LibreOffice >= 4.3.0 installed on the same machine
2. Install a package with `pip install templated-docs`
3. Add `templated_docs` to `INSTALLED_APPS`

If you are using cpython you may need the `libffi` development package in order to compile this module's dependencies.

- **Ubuntu**: `apt-get install libffi-dev`
- **MacOS**: `brew install libffi`

CHAPTER 3

Usage

To generate a document from a template `sample.odt` you can write a view like this:

```
from templated_docs import fill_template
from templated_docs.http import FileResponse

def get_document(request):
  """
  A view to get a document filled with context variables.
  """
  context = {'user': request.user, 'foo': 'bar'}
  filename = fill_template('sample.odt', context, output_format='pdf')
  visible_filename = 'greeting.pdf'
  return FileResponse(filename, visible_filename)
```

templated_docs.fill_template(template_name, context, output_format='odt') Fill a template `template_name` using a `context` dictionary as a context, optionally converting it to the `output_format`. Returns a filename of a generated file.

CHAPTER 4

More information

4.1 Working with document templates

4.1.1 Supported formats

Important: The templates themselves **must** be in on of the OpenDocument formats: `.odt`, `.ods`, `.odp` or `.odg`.

Templated-docs can generate documents in any format LibreOffice itself supports. For the moment they are:

Text documents: doc, docx, fodt, html, odt, ott, pdf, txt, xhtml, png

Spreadsheets: csv, fods, html, ods, ots, pdf, xhtml, xls, xlsx, png

Presentations: fodp, html, odg, odp, otp, pdf, potm, pot, pptx, pps, ppt, svg, swf, xhtml, png

Drawings: fodg, html, odg, pdf, svg, swf, xhtml, png

4.1.2 Where to put templates

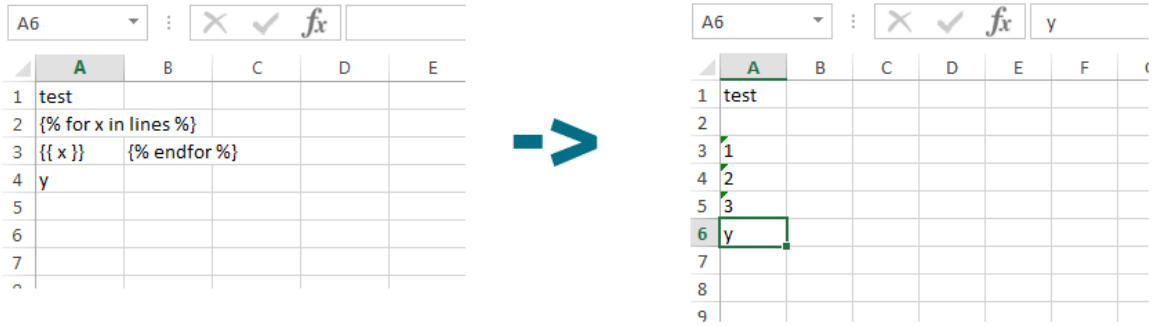
Templated-docs uses a standard Django template discovery mechanism, so you should place your documents at a path where other templates reside. Usually it's either a `templates` directory at the project level, or `templates` folders in each applications.

4.1.3 What subset of templating language is supported

Django templating language is fully supported, including filters and tags. Keep in mind, though, that in order to use custom template tags you should load them first using a standard `{% load %}` tag.

4.1.4 Looping

Generating lines in loops is supported via the standard `{% for %}` tag. Here's the picture that illustrates the concept:



4.2 Useful template tags

To use these template tags, load them first with `{% load templated_docs_tags %}`.

4.2.1 lolinebreaks

To keep newlines in a multiline text, use this tag instead of a default `linebreaks` filter:

```
{{ variable|lolinebreaks }}
```

4.2.2 image

Inserting images from models' ImageField is supported. Assuming you have a `claim` model instance with a `stamp` ImageField in it:

```
{% image claim.stamp %}
```

4.3 Serving generated documents over HTTP

A common task is to serve user a created file. To facilitate this, a handy `FileResponse` response class is available:

```
from templated_docs.http import FileResponse

def your_view(self):
    filename = get_my_document()
    return FileResponse(filename, visible_name='document.pdf')
```

A `FileResponse` constructor is a subclass of Django's `HttpResponse`, providing the following arguments:

- `actual_file` - the real filename to serve
- `visible_name` - the name a user will see in the browser
- `delete` - set this to *False* to skip file deletion

Important: The *actual_file* **is deleted by default** after being served, as usually this is the wanted behaviour. To keep the file, pass the `delete=False` argument to the `HttpResponse` constructor.

4.4 Management commands for generating documents

Todo: describe `templated_docs.management.base.DocumentGenerationCommand`.

4.5 Templated-docs settings

Some settings can be utilised to alter the package behaviour:

- **TEMPLATED_DOCS_LIBREOFFICE_PATH** - a path to the LibreOffice installation's *program* folder. Defaults to `/usr/share/libreoffice/program`.