Create PDFs from Markup with Python

Lorna Mitchell

Meet `rst2pdf`

- https://rst2pdf.org
- Open source, Python project
- Accepts ReStructured Text format, outputs PDF
- Useful for: reports, slide decks, ... all sorts

ReStructured Text

ReStructured Text

A text-based markup format (like markdown, but better).

Use your favourite text editor!

Use source control.

A note on source control

Source control is for people who:

- don't like losing work
- sometimes make mistakes
- might want to collaborate with others some day

ReStructured Text Examples

Slide with bullets:

```
Meet ``rst2pdf``
==============

* http://rst2pdf.github.io
* Open source, Python project
* Accepts ReStructured Text format, outputs PDF
* Useful for: reports, slide decks, ... all sorts

.. raw:: pdf

    PageBreak titlePage
```

ReStructured Text Examples

Code block:

```
.. code-block:: python

    def inbound_sms():
        if request.is_json:
            pprint(request.get_json())
```

Image:

```
.. image:: images/postman.png
:width: 26cm
```

Portable Document Format

Portable Document Format

PDF is my preferred file format for things that:

- should be well presented
- need not be edited

It is a self-contained, open format, fonts can be embedded, and there are loads of tools that work with it.

PDF Tools

The best thing about PDF? The tools!

- **`pdftk`** for manipulating PDFs (including the surprise later in this slide deck)
- **`pdfjam`** for utterly painless handouts
- **`pdfpc`** for presenting from PDF including speaker notes, timer etc

Create PDFs from Markup with Python

Lorna Mitchell

Meet `rst2pdf`

- https://rst2pdf.org
- Open source, Python project
- Accepts ReStructured Text format, outputs PDF
- Useful for: reports, slide decks, ... all sorts

ReStructured Text

ReStructured Text

A text-based markup format (like markdown, but better).

Use your favourite text editor!

Use source control.

A note on source control

Source control is for people who:

- don't like losing work
- sometimes make mistakes
- might want to collaborate with others some day

ReStructured Text Examples

Slide with bullets:

```
Meet ``rst2pdf``
==============

* http://rst2pdf.github.io
* Open source, Python project
* Accepts ReStructured Text format, outputs PDF
* Useful for: reports, slide decks, ... all sorts

.. raw:: pdf

    PageBreak titlePage
```

ReStructured Text Examples

Code block:

```
.. code-block:: python

    def inbound_sms():
        if request.is_json:
            pprint(request.get_json())
```

Image:

```
.. image:: images/postman.png
:width: 26cm
```

Portable Document Format

Portable Document Format

PDF is my preferred file format for things that:

- should be well presented
- need not be edited

It is a self-contained, open format, fonts can be embedded, and there are loads of tools that work with it.

PDF Tools

The best thing about PDF? The tools!

- **`pdftk`** for manipulating PDFs (including the surprise later in this slide deck)
- **`pdfjam`** for utterly painless handouts
- **`pdfpc`** for presenting from PDF including speaker notes, timer etc

`rst2pdf`

How to `rst2pdf`

Everything you need is here: https://rst2pdf.org/

```
pip install rst2pdf
```

Or on GitHub https://github.com/rst2pdf/rst2pdf

rst2pdf

How to `rst2pdf`

Everything you need is here: https://rst2pdf.org/

```
pip install rst2pdf
```

Or on GitHub https://github.com/rst2pdf/rst2pdf

How to rst2pdf

Everything you need is here: https://rst2pdf.org/

```
pip install rst2pdf
```

Or on GitHub https://github.com/rst2pdf/rst2pdf

How to rst2pdf

Use **rst2pdf** like this:

```
rst2pdf awesomeness.rst
```

You can supply styles, layouts, output file name but ... this is basically it

Separate Content and Style

I love this approach!

- Style is always consistent throughout document.
- Styles can be re-used between documents.
- Content can be re-used between documents.
- Documents can VERY easily be re-styled (is my contrast OK?).
- You can grep for content.

Extra bonus: tool itself is also separate

Code Samples

It's python, and uses pygments:

```python
def inbound_sms():
    if request.is_json:
        pprint(request.get_json())
    else:
        data = dict(request.form) or dict(request.args)
        pprint(data)

    return ('', 204)
```

https://github.com/nexmo-community/nexmo-python-quickstart

The `rst2pdf` Project

The `rst2pdf` Project

Current version: 0.94 Released: 24th January 2019

Previous version: 0.93 Released: 19th December 2012

Project Status

- Python 2 only at this moment
- Working test suite and decent test coverage
- Good documentation, new website
    - includes examples, open an issue if you have a question

How do you test a PDF?

1. Generate a PDF that is "right"
2. Create a hash of that file
3. All future tests generate and hash the PDF for this test and compare it
    - it's a selection of hashes; different tools make different outputs

rst2pdf

Use a text editor. Make a PDF.

https://rst2pdf.org

Resources

`rst2pdf`

- https://rst2pdf.org
- https://github.com/rst2pdf/rst2pdf
- https://github.com/ralsina/rst-cheatsheet

Me

- https://lornajane.net
- **`@lornajane`**