P.A.P.E.R.

Python in a Smalltalk way

Everybody is a programmer

- Computer literacy is catching up
- Everybody is learning to program
  - Everybody is learning to program **Python**

Traditional software

- Backend
  - Encapsulates state in the form of one or more **objects**
- Frontend
  - Exposes the specific functionality of backend objects

Issues with traditional software

- Slow speed to expose new functionality in the backend
- Shackle users to specific workflows in the interface
- Users are restricted by the imagination and skills of the devs

The Smalltalk philosophy

- All users are developers
- There is no distinction between programming and using the software
- Everything happens in the integrated environment

Smalltalk philosophy and Python

- Python is a language rooted on the interpreter
- Docstrings, help, dir and related commands make learning new libraries through interpreter exploration a common pattern
- This has been further extended in the Jupyter notebooks framework, allowing for UI elements

Jupyter UI elements

- In 2016, right before leaving Montreal, I gave a talk at Montreal Python on using UI widgets for quick prototyping of UIs
  - https://montrealpython.org/en/2016/01/mp56/
- This talks build on that
  - It goes a step further: rather than prototyping UIs, I realized this is the best UI for this tool

The tool

keeping track of what you read using Jupyter notebooks

We read and forget a lot

- Why read it if you are going to forget it?
- Keep **metadata** about what you read.
  - Breadcrumbs to find it again.
- **Known-item search** has become hideously difficult with Web search engines.
  - A wall of spammers hide your item!

Memory hooks

- Many times I don’t remember the title, nor the authors.

- But I remember:
  - Where I read it (physical place)
  - How I read it (physical device)
  - How I found it
  - Approximate date when I read it.

P.A.P.E.R.

- Since 2012, I have been developing a paper management solution cater to my needs.
  - Available at https://github.com/DrDub/PAPER

- End user presentation: https://github.com/DrDub/PAPER/blob/main/docs/20200930_learnds_lighting.pdf
  - https://www.youtube.com/watch?v=BQkIl6pACKw

Basics

- **Data model**: hierarchical attribute-value pair DAG in a YAML file (BibTeX+Mind Map).
- **File repository**: a digital assets manager that copies PDFs and other source paper files into a unified folder-balanced hash-based space in your file system.
- **Search engine**: for files in the paper file repository, the text is extracted and stored in an index that allows queries, including query-by-example, to find similar papers to existing ones.

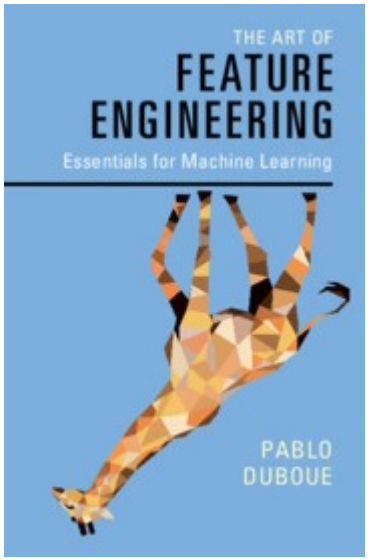
Brief demo

https://github.com/DrDub/PAPER

Proof is in the pudding

- I use this tool to keep track of the material for the book I just published.
  - The Art of Feature Engineering
    - Cambridge University Press
    - ISBN 978-1108709385

      `http://artoffeatureengineering.com/`



- 103 citations in-tool.
  - 200+ out of tool (oh, well).

Thanks

- This tool started at Les Laboratoires Foulab
  - Thanks to Rupert Brooks for help and encouragement
- Presented Jupyter for quick prototyping at Montréal Python #56
  - Thanks also to George Peristerakis for help and discussions

`https://github.com/DrDub/PAPER`

More links

- The feature engineering book comes with 10,000 lines of Jupyter notebooks:
  - https://github.com/DrDub/artfeateng
- A chapter on mining AirBnB data using PySpark:
  - https://github.com/DrDub/artfeateng/tree/master/tourism
- Homomorphic encryption for machine learning, in Python:
  - https://github.com/Textualization/riiaa21_ws11_ml_over_encrypted_data