Publish by Example

Hans Petter Langtangen[1,2]

[1]Center for Biomedical Computing, Simula Research Laboratory,
[2]Deptartment of Informatics, University of Oslo

Apr 18, 2013

Contents

1 Motivation

Publish is a software for managing your own list of publications or any list of references to literature. There are numerous tools available for this purpose. The most well-known to scientists is perhaps BibTeX. Publish offers several important advantages over BibTeX:

1. the format is simpler (plain self-explanatory text),
2. multiple registrations of the same publication are detected and merged to one master registration,
3. flexibly selected parts of the list of publications can be output in different formats: BibTeX, HTML, reStructuredText, PDF, ...,

4. the list of publications can easily be manipulated in simple Python code to meet various demands for publishing or maintaining the list (e.g., exporting the list to other types of software).

Point 2 and 3 are particularly important: publication lists needs to be formatted also for the web nowadays, and many published lists suffer from double entries in BibTeX because of slightly different input by different people.

There should be no need to emphasize the importance of having beautifully formatted and absolutely correct publication data in your CV and in your papers. My claim is that unless you really like to constantly check and polish your BibTeX files, your life gets much simpler if you use Publish for all publication records and generate BibTeX or other formats from the Publish format.

2 Publish in action

The following sections takes you through the main steps of creating your first Publish database:

1. import data from another source, here BibTeX files,
2. clean up data,
3. export to other formats,
4. customize the behavior of Publish.

2.1 Importing a BibTeX file

Suppose you have some BibTeX files and want to put these in a Publish database. Technically, this means merging the BibTeX files to one text file with a simple format. You may choose to have one Publish database or many.

Here is a sample BibTeX file called `refs1.bib`:

```
@InProceedings{Mardal_et_al_2003a,
author =    {K.-A. Mardal and G. W. Zumbusch and H. P. Langtangen},
editor =    {H. P. Langtangen and A. Tveito},
title =     {Software Tools for Multigrid Methods},
booktitle = {Advanced Topics in Computational Partial Differential
            Equations -- Numerical Methods and Diffpack Programming},
publisher = {Springer},
series    = {Lecture Notes in Computational Science and Engineering},
year =      {2003},
pages =     {97-152},
}

@TechReport{Jeberg_et_al_2004,
author =    {P. V. Jeberg and H. P. Langtangen and C. B. Terp},
title =     {Optimization with {Diffpack}: Practical Example from
             Welding},
year =      {2004},
institution =  {Simula Research Laboratory},
type =    {Report 2004-12},
}

@Misc{Langtangen_talk_2007a,
author = {H. P. Langtangen},
title = {Computational Modeling of Huge Tsunamis from Asteroid Impacts},
month = {May},
year = {2007},
howpublished = {Invited keynote lecture at the \emph{International
 conference on Computational Science 2007 (ICCS'07)}, Beijing, China},
}

@Article{Mortensen_et_al_2011,
author         = {M. Mortensen and H. P. Langtangen and G. N. Wells},
title          = {A FEniCS-Based Programming Framework for Modeling
Turbulent Flow by the {Reynolds}-Averaged {Navier-Stokes} Equations},
journal        = {Advances in Water Resources},
year           = {2011},
volume         = {34},
number         = {9},
doi            = {10.1016/j.advwatres.2011.02.013},
}

@Book{Langtangen_2012,
author =    {H. P. Langtangen},
title =     {Python Scripting for Computational Science},
series =    {Texts in Computational Science and Engineering},
publisher = {Springer},
year =      {2012},
edition =   {Third},
}
```

Importing this to Publish format is done by

```
Terminal> publish import refs1.bib
```

Publish creates a database with the default name `papers.pub`. It has the following content:

```
* articles
** A FEniCS-Based Programming Framework for Modeling Turbulent Flow by the {Reynolds}-Averaged {Navier
   key:       Mortensen_et_al_2011
   author:    M. Mortensen, H. P. Langtangen, G. N. Wells
   year:      2011
   journal:   Advances in Water Resources
   volume:    34
   number:    9
   doi:       10.1016/j.advwatres.2011.02.013
   status:    published
   entrytype: article
* books
** {P}ython Scripting for Computational Science
   key:       Langtangen_2012
   author:    H. P. Langtangen
   year:      2012
   publisher: Springer
   status:    published
   series:    Texts in Computational Science and Engineering
   edition:   Third
   entrytype: book
* proceedings
** Software Tools for Multigrid Methods
   key:       Mardal_et_al_2003a
   author:    K.-A. Mardal, G. W. Zumbusch, H. P. Langtangen
   editor:    H. P. Langtangen, A. Tveito
   year:      2003
   booktitle: Advanced Topics in Computational Partial Differential Equations -- Numerical Methods and
   publisher: Springer
   pages:     97-152
   status:    published
   series:    Lecture Notes in Computational Science and Engineering
   entrytype: inproceedings
* reports
** Optimization With {Diffpack}: Practical Example From Welding
   key:         Jeberg_et_al_2004
   author:      P. V. Jeberg, H. P. Langtangen, C. B. Terp
   year:        2004
   institution: Simula Research Laboratory
   status:      published
   entrytype:   techreport
   type:        Report 2004-12
* misc
** Computational Modeling of Huge Tsunamis From Asteroid Impacts
   key:          Langtangen_talk_2007a
   author:       H. P. Langtangen
   year:         2007
   status:       published
   month:        May
   howpublished: Invited keynote lecture at the \emph{International conference on Computational Scienc
   entrytype:    misc
```

The format, referred to as the *pub* format, should be self-explanatory. Note the following:

- Multiple-line values in the BibTeX files are (and must be) single line in the pub format.
- Keys in the BibTeX format become the attribute `key` in the pub format.
- Most other attributes in the BibTeX format have the same names in the pub format.
- Capitalization, as in 'Navier-Stokes, is also preserved in the pub format.
- Capitalization forgotten in the BibTeX file, here in the word `Python`, is fixed in the pub format (`{P}ython`).
- Special LaTeX formatting, such as `\emph{...}`, is also valid in the pub format.

Publish treated the `refs1.bib` file without any remarks. That is the exception from the rule: usually Publish will complain about incomplete or inconsistent information in the BibTeX file, as the next two sections illustrate.

2.2 Adding information

Imagine we also have another BibTeX file, `refs2.bib`, that we would like to add to the Publish database `papers.pub`. We just run `publish import refs2.bib` to accomplish this task. However, the `refs2.bib` contains a reference

```
@Article{Haga_et_al_2011a,
author  = {J. B. Haga and H. Osnes and H. P. Langtangen},
title   = {On the causes of pressure oscillations in low-permeable
           and low-compressible porous media},
journal = {International Journal of Analytical and Numerical
           Methods in Geomechanics},
year    = {2011},
doi     = {10.1002/nag.1062},
url = {http://onlinelibrary.wiley.com/doi/10.1002/nag.1062/abstract}
}
```

Publish has not registered the journal in this reference. It therefore prompts the user and suggests that maybe a journal with a similar name, in this case *International Journal for Numerical Methods in Engineering*, is what was meant.

```
Validating paper: (Haga_et_al_2011a) - On the causes of pressure
oscillations in low-permeabl...
  Status is not defined, assuming status is "published".

  Unknown journal: "International Journal of Analytical and
  Numerical Methods in Geomechanics"
  Suggested journal: "International Journal for Numerical
  Methods in Engineering"

  Unknown journal, what should I do?
  [1] Replace journal.
  [2] Add journal.
  [3] Skip paper.
Please enter 1, 2 or 3 (or press return to choose [1]):
```

We need to add the journal and hence choose `[2]`. Added journals, conference proceedings, and other venues are placed in a database file with default name `venues.list` in the current working directory. The next time we import some entry with this journal name, the name will be known to Publish.

The next entry in the `refs2.bib` reads

```
@Article{Rahman_et_al_2006b,
author  = {S. Rahman and J. Gorman and C. H. W. Barnes
           and D. A. Williams and H. P. Langtangen},
title   = {Numerical Investigation of a Piezoelectric Surface
           Acoustic Wave Interaction with a One-Dimensional Channel},
journal = {Physical Review B},
note    = {035308},
volume  = {74},
year    = {2006},
}
```

Publish finds the journal name of this entry incomplete and therefore says

```
Validating paper: (Rahman_et_al_2006b) - Numerical Investigation
  of a Piezoelectric Surface A...
  Status is not defined, assuming status is "published".

  Unknown journal: "Physical Review B"
  Suggested journal: "Physical Review B: Condensed Matter
  and Materials Physics"
  Unknown journal, what should I do?
  [1] Replace journal.
  [2] Add journal.
  [3] Skip paper.
Please enter 1, 2 or 3 (or press return to choose [1]):
```

The right answer here is the complete name of the journal: `[1]`.

2.3 Merging publication records

When publish continues with the `refs2.bib` file it hits the entry

```
@Book{Langtangen_2011,
author =    {H. P. Langtangen},
title =     {Python Scripting for Computational Science},
series =    {Texts in Computational Science and Engineering},
publisher = {Springer},
year =      {2011},
edition =   {Second},
}
```

This is the same book as in `refs1.bib`, just a previous edition. Publish detects that the entries are very similar and prompts the user about what to do:

- Keep both references?
- Ingore both?
- Keep the first reference and ignore the second? Or the other way around?
- Merge the two reference into one by choosing a conflicting attribute from the first reference or the second?

In this particular case, we are notified about the `edition` attribute, should it be `Third` or `Second`? Let us say that we just want the newest one, but decide to examine every attribute, so we choose `Third` as the correct edition (alternative 5). Then we are asked about the key, do we want `Langtangen_2012` or `Langtangen_2011` as the key in the merged item? The consistent choice is `Langtangen_2012` (alternative 5). Also the year attribute differs, and we have to choose `2012`. The questions are asked again. The resulting item in the Publish database reads

```
* books
** {P}ython Scripting for Computational Science
   key:       Langtangen_2012
   author:    H. P. Langtangen
   year:      2012
   publisher: Springer
   status:    published
   series:    Texts in Computational Science and Engineering
   edition:   Third
   entrytype: book
```

2.4 Exporting references

We can export all entries in the database to "BibTex": "/doc/tutorial/src/refs.bib", PDF (via LaTeX), HTML, and reStructuredText:

```
Terminal> publish export refs.bib   # BibTeX
Terminal> publish export refs.html  # HTML
Terminal> publish export refs.rst   # reStructuredText
```

2.5 Capitalization

Sometimes words in titles need to be capitalized. One example is *Navier-Stokes*. For some output formats such as HTML and reStructuredText the word Navier-Stokes will appear correctly if it is correctly capitalized in the Publish database. For output to PDF via LaTeX, one relies on BibTeX, and different BibTeX styles may treat capitalization of titles differently. The common style called `plain` will typically typeset titles in lower case. Writing just Navier-Stokes in the title then results in navier-stokes. This is a well-known issue for BibTeX users. Publish adopts the same capitalization technique as BibTeX: the capitalization of words or characters appearing inside curly braces is preserved. It means that if we write `{Navier-Stokes}` in the title, or `{N}avier-{S}tokes`, we are guaranteed that the output to LaTeX and PDF will be Navier-Stokes and not navier-stokes.

As with BibTeX, Publish users must carefully read titles in the database and ensure that curly braces are used whenever capitalization requires it. There is a way out of this described in the next section.

2.6 Configuration

The behavior of Publish can be configured through a *configuration file* called `publish_config.py`. Suppose we want to change the database name from `papers.pub` to a hidden file with a more descriptive name: `.publish_references.pub`. Similarly, we want to rename the `venues.list` file to `.publish_venues.txt`. To this end, we make a file `publish_config.py` with the content:

```
from publish.config.defaults import *

database_filename = '.publish_references.pub'
local_venues_filename = '.publish_venues.txt'
```

The `defaults.py` file in the directory `publish/config` of the Publish source code tree contains all variables that can be configured by the user in a local `python_config.py` file. These variables include measures of the distance between strings, used to decide if two titles or other reference attributes are sufficiently close to prompt the user for information.

The dictionary `uppercase` in `defaults.py` ensures that its listed words are properly capitalized when Publish data is exported. Looking closely at the Publish database, we realize that one entry has a title starting with ”A FEniCS-based Programming ...”. The word FEniCS should be capitalized properly, but the title lacks curly braces (because the BibTeX file from which the entry was imported also lacks proper braces). Instead of inserting braces in the problematic word `FEniCS-based`, we can add this word to the `uppercase` dictionary. A key in this directory is the lowercase version of a word, while the value is the correct capitalization specified via curly braces. In our case we extend this dictionary with one item,

```
uppercase.update({
    "fenics-based": "{FEniCS}-based",
    })
```

Publish will then always ensure that any version of the word FEniCS-based will appear with capital FE and CS in various output formats. A lot of words are already registered in `uppercase`, such as FEM, FDM, Python, Cython, FORTRAN, MATLAB, and FEniCS. If you prefer `Fortran` over the official `FORTRAN`, you can either write `{F}ortran` yourself in the database, or adjust the `uppercase` dictionary in the configuration file:

```
uppercase.update({
    "fortran": "{F}ortran",
    })
```

Publish reads your configuration file by `import python_config`. This means that you can just place your `python_config.py` file in the directory where you have the Publish database and where you run Publish from. (Larger projects dealing with publication management may want to operate with different configuration files. Steering which file to use is then done via the rules for import in Python. Normally, one sets the `PYTHONPATH` environment variable appropriately to control the import of the desired configuration file.)

3 Programming Publish

- New output not in categories.
- New output formatting.