Project documentation

Why do you need documentation?

- You want yourself to understand how code written some time ago works
- You want others to understand how to (re-)use your code

For this you need to

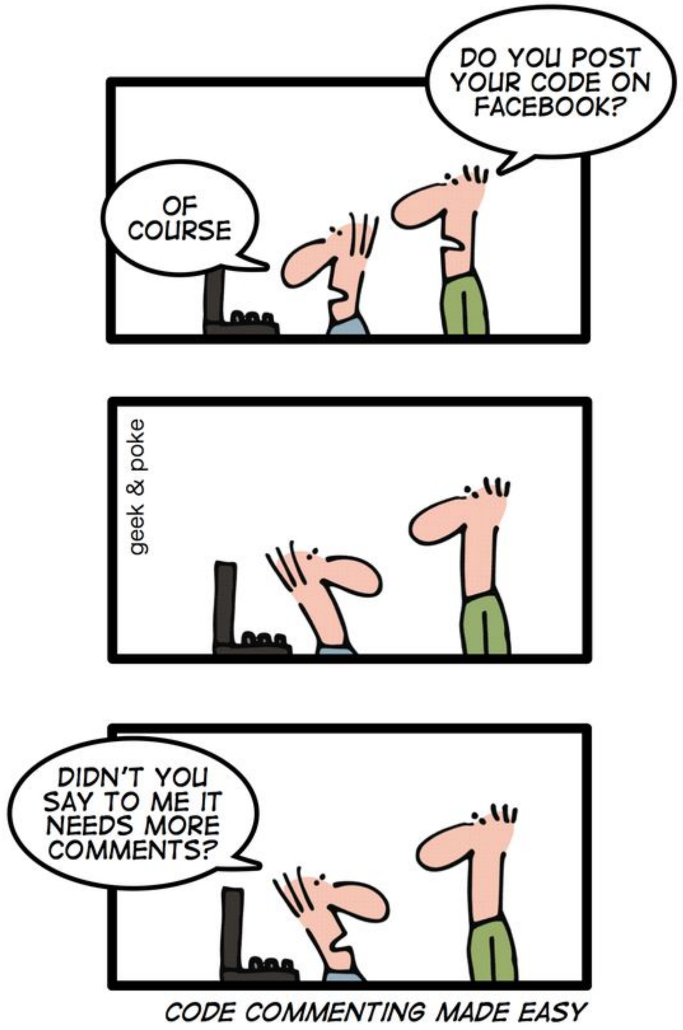
- Explain parts of your code with comments
- Explain what to install and how to get started in your readme
- Explain in-depth use of your code in a notebook

Comments

Comments are annotations you write directly in the code source.

They:

- are written for users who deal with your source code
- explain parts that are not intuitive from the code itself
- **do not** replace readable or structured code
- (in a specific structure) can be used to directly generate documentation for users.


DO YOU POST YOUR CODE ON FACEBOOK?
OF COURSE
geek & poke
DIDN'T YOU SAY TO ME IT NEEDS MORE COMMENTS?
CODE COMMENTING MADE EASY

When *not* to use comments

- ...to repeat in natural language what is written in your code

```
% Now we check if the age of a patient is greater than 18
if agePatient > 18
```

- ...to turn old code into zombie code (fine for troubleshooting, but do not leave it in!)

```
% Do not run this!!
% itDoesNotWork  =optimizeMulticoreDeepLearning(myProteins)
% if itDoesNotWork == 1444
%    connection = connectToHPC(currentUser, password)
% end
%}
```

When *not* to use comments

- ...to replace version control, like git

```
%removed on August 5
% if ...

% Now, it connects to the API with o-auth2, updated 05/05/2016
...
```

Comment lines: WHY over HOW

Comment lines are used to explain the **purpose** of some piece of code.

```
# Bug fix GH 20601
# If the data frame is too big, the number of unique index combination
# will cause int32 overflow on windows environments.
# We want to check and raise an error before this happens
num_rows = np.max([index_level.size for index_level in self.new_index_levels])
num_columns = self.removed_level.
```

From Pandas reshape.py documentation

Docstrings

- Structured comments, associated to *segments* (rather than lines) of code, can be used to generate documentation for users* of your project.
- These comments are called *docstrings*.
- Docstrings are parsed as the first statement of a module (e.g. a function or class).
- Docstrings allow you to provide documentation to a function, that is relevant to the user of that function.
- Writing docstrings makes you generate your documentation as you are generating the code: efficiently, comprehensively!

**Remember? That's probably you!*

Generating docstrings

```
function c = addme(a,b)
% ADDME  Add two values together.
%   C = ADDME(A) adds A to itself.
%
%   C = ADDME(A,B) adds A and B together.
%
%   See also SUM, PLUS.

switch nargin
    case 2
        c = a + b;
    case 1
        c = a + a;
    otherwise
        c = 0;
end
```

When you type `help addme` at the command line

```
addme  Add two values together.
    C = addme(A) adds A to itself.

    C = addme(A,B) adds A and B together.

    See also sum, plus.
```

Docstring styleguides

- Google Python Docstring
- Numpy Python Docstring
- Matlab - look at MATLAB's own functions

Example from PVMD Toolbox - Create output folder

A glimpse into code generation

Docstrings are formatted so that they can easily be turned into documentation of your package.

- http://www.doxygen.nl/ : C++ (and many more languages)
- http://www.sphinx-doc.org/ : Python, MATLAB
- https://roxygen2.r-lib.org/ : R

We will not do this today, but it is worth checking out if you want to release your code!

Example PVMD Toolbox - Docstrings and sphinx

Your turn (choose one!)

1. Comment lines

a. Do you have superfluous comments? **Remove them!**

- Remove your zombie code and version control-like comments
- See if you can replace a 'how' comment for a 'why' comment (*what is the purpose of this code?* rather than *this is how this code works*)

b. Are there elements without comments that need them? **Add them!**

- Have you found yourself staring at a piece of code for too long without understanding it? Perhaps it needs more information!
- Try to comment on the thought behind the code rather than phrasing it in English.

2. Docstrings

- Add a docstring to a function, preferably the last function you worked on (so it's fresh in your memory).
- Keep in mind: what does my user need to know when they are working with this function?

README

The README page is the first thing your user will see!

What should be included as a bare minimum in README files?

README

As a bare minimum a README file should include:

- A descriptive project title
- Motivation (why the project exists)
- How to setup
- Copy-pastable quick start code example
- Recommended citation

Example eScience Center - matchms