Best Practices

Kidane M. Tekle

March 2019

Disclarimer!

All pictures used are from random searches of the web and for educational purposes. They might be subject to specific licenses and should be checked before using further.

Introduction

Overall

Coding

Collaboration

Summary

Best Practices

Introduction

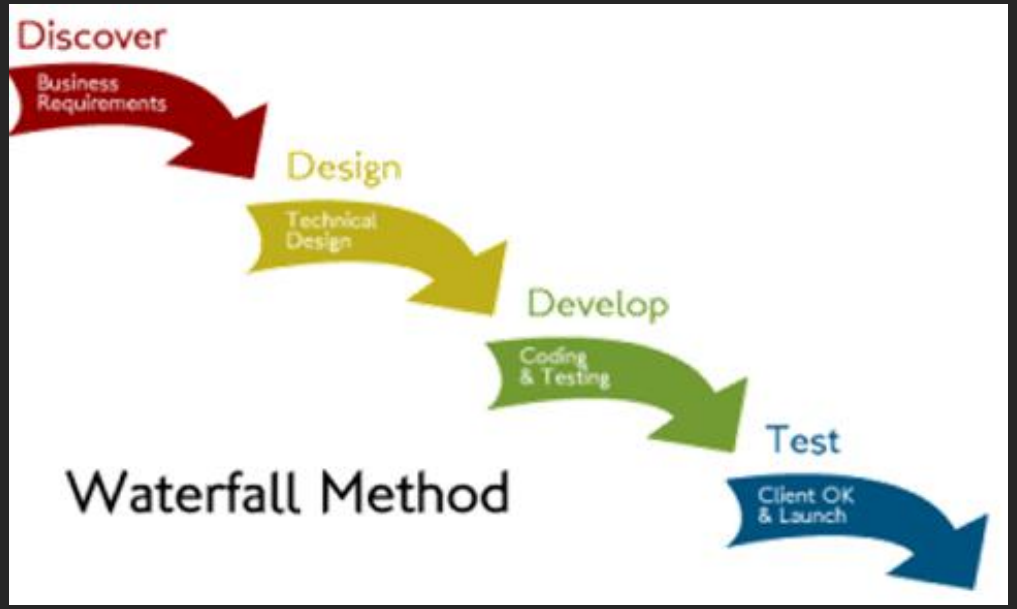
Overall

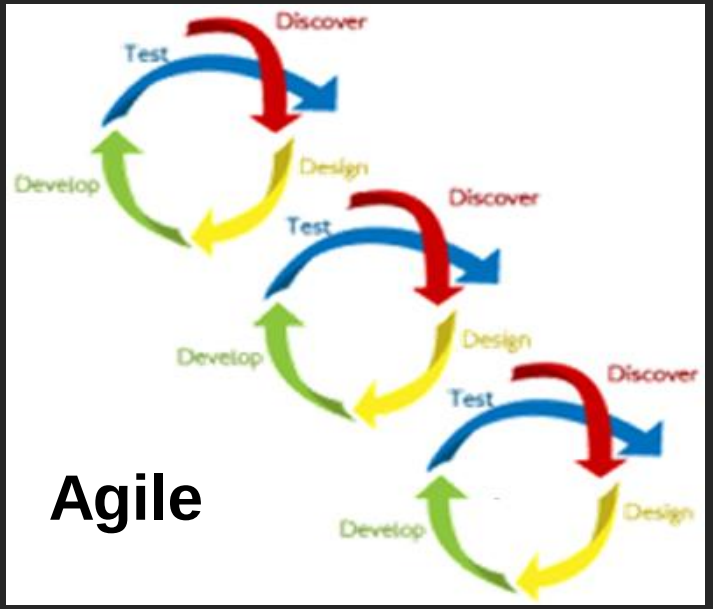
Coding

Collaboration

Summary

Overall: Be Agile





Overall : Use the most suitable tooling

1. Planning
2. Development
   - ✓ Workbench (IDE)
   - ✓ Programming Language
   - ✓ Framework vs. Library
   - ✓ Database
3. Team engagement
4. Documentation
5. Time / Schedule management
6. TODO / Task management

Overall: Have a planning habit

- Have at least 2 sessions

1. Focus / Category
   - ✓ List top level things first
   - ✓ Keep them a few (optimally <=5)
2. Details
   - ✓ Be very precise
   - ✓ What is the concrete deliverable of each?
   - ✓ Required resources & Expected challanges

Introduction

General

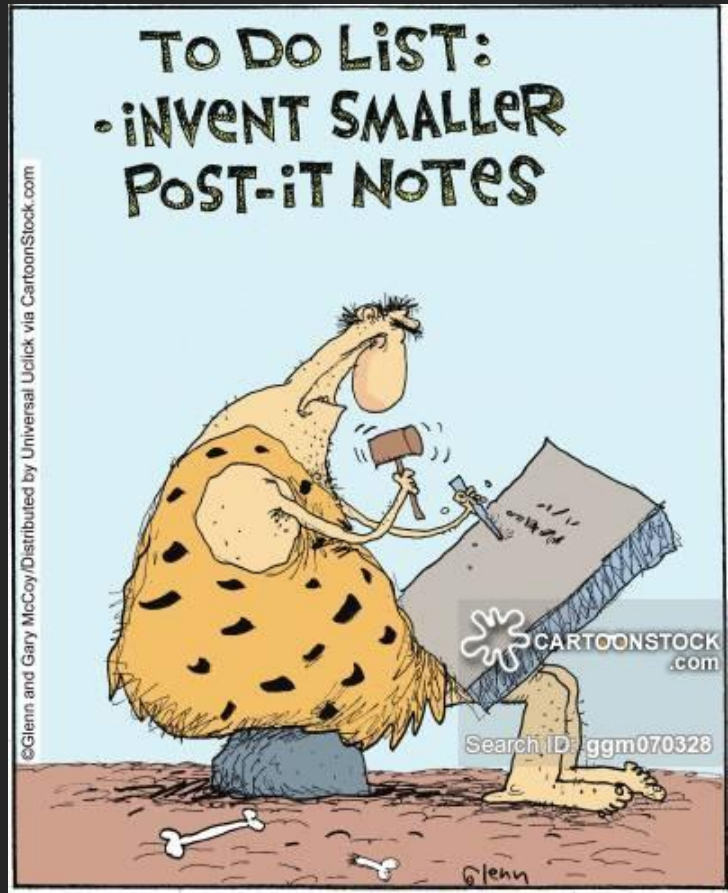
Coding

Collaboration

Summary

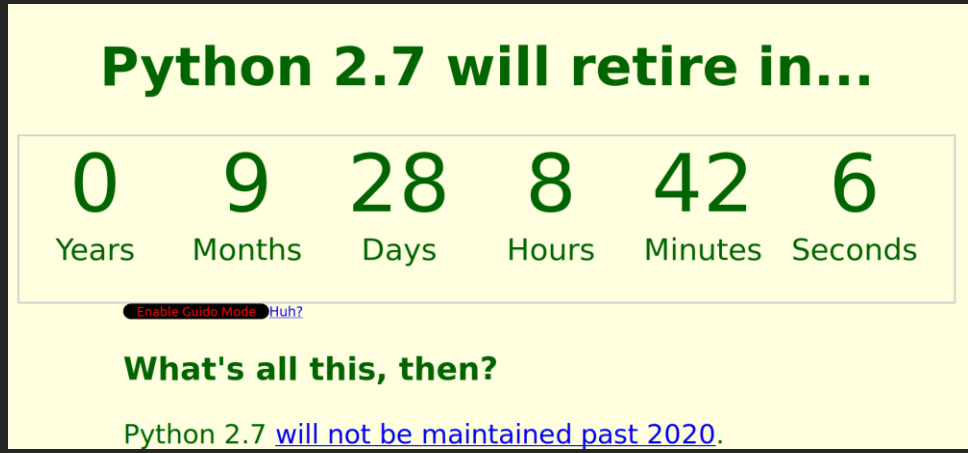
Coding: Use a modern IDE

- Subjective reasons
  - I have always used X, I am comfortable with it
  - It's what the cool kids are using
  - Tool of choice of the company / group
- Objective reasons
  - Natural fit for the problem at hand
  - Support of best practices
  - Assistance to the developer
  - License



Coding: update your language version

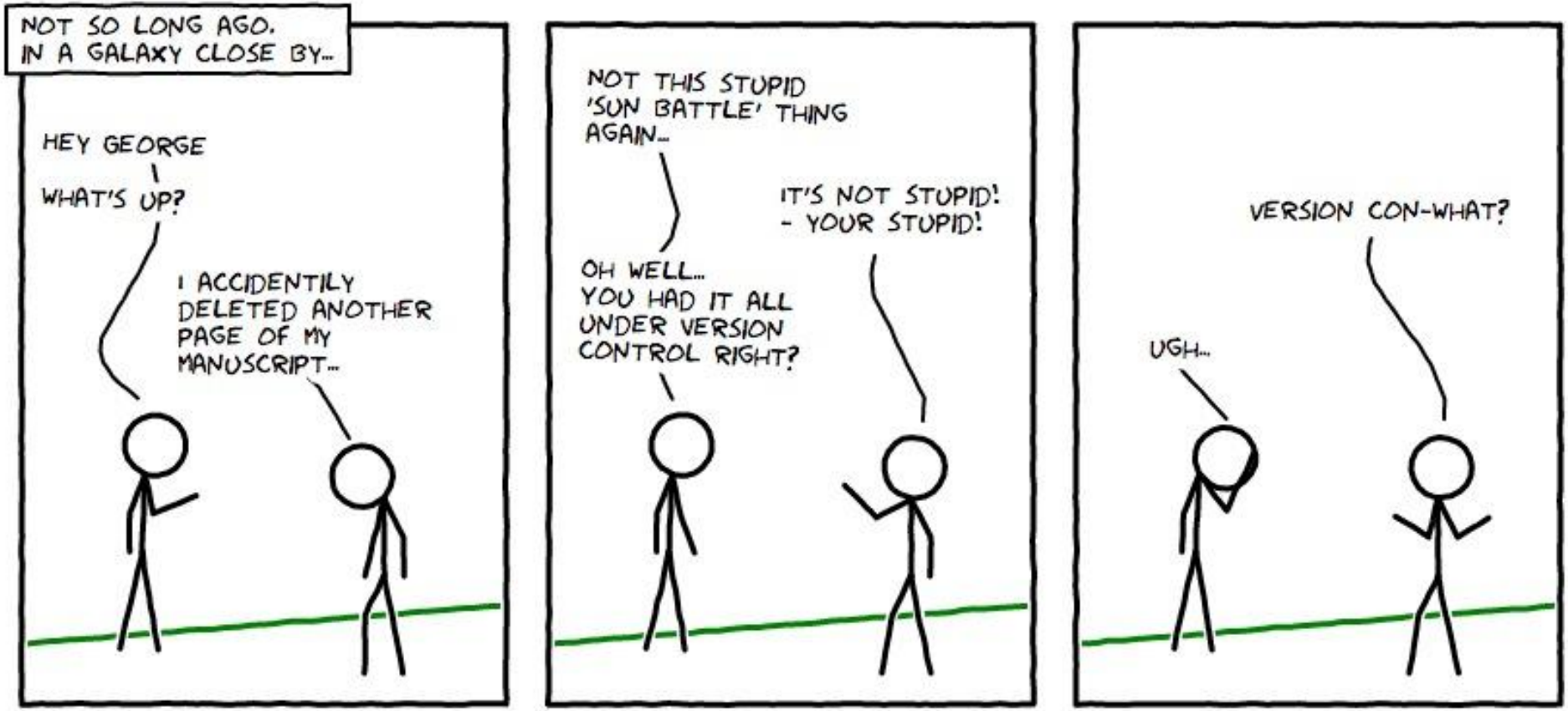
- Python >= 3.5
  - Duck typing
- Javascript => Typescript
- Java >= 8
  - Functional programming



https://pythonclock.org/

https://learntocodewith.me/programming/python/python-2-vs-python-3/
https://www.graycelltech.com/why-typescript/
http://blog.codelv.com/2018/04/python-27-vs-36-in-2018.html

Coding: Use version control



Coding: Keep just enough design / documentation

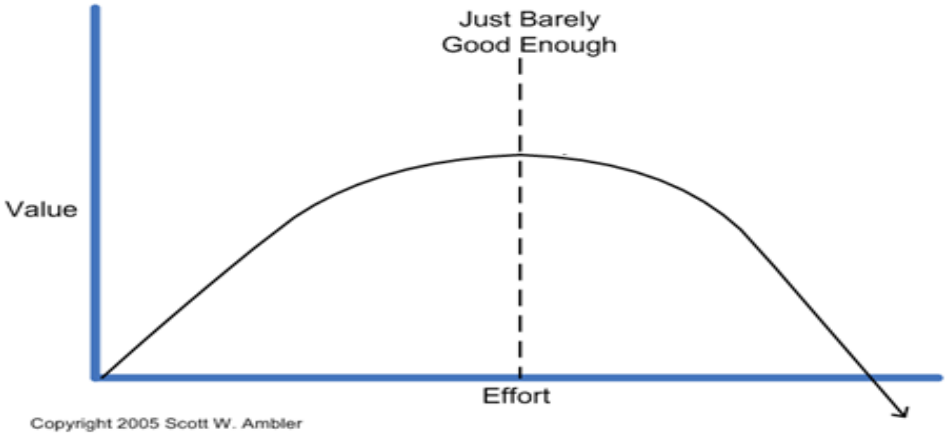


Fig 1. Ideal Curve

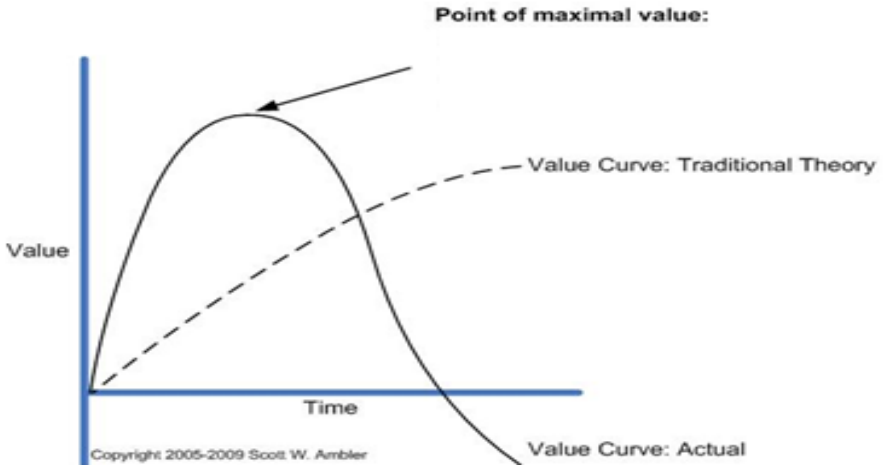


Fig 2. A more realistic plot

http://agilemodeling.com/essays/barelyGoodEnough.html

Coding: consistency (control logic example)

```
if c1:
        work1t
        if c2:
                work2t
                if c3:
                        work3t
                else:
                        work3f
        else:
                work2f
else:
        work1f
```

(3 if, 3 else, 12 lines)

```
if ! c1:
        work1f
        return
work1t

if ! c2:
        work2f
        return
work2t

if ! c3:
        work3f
        return
work3t
```

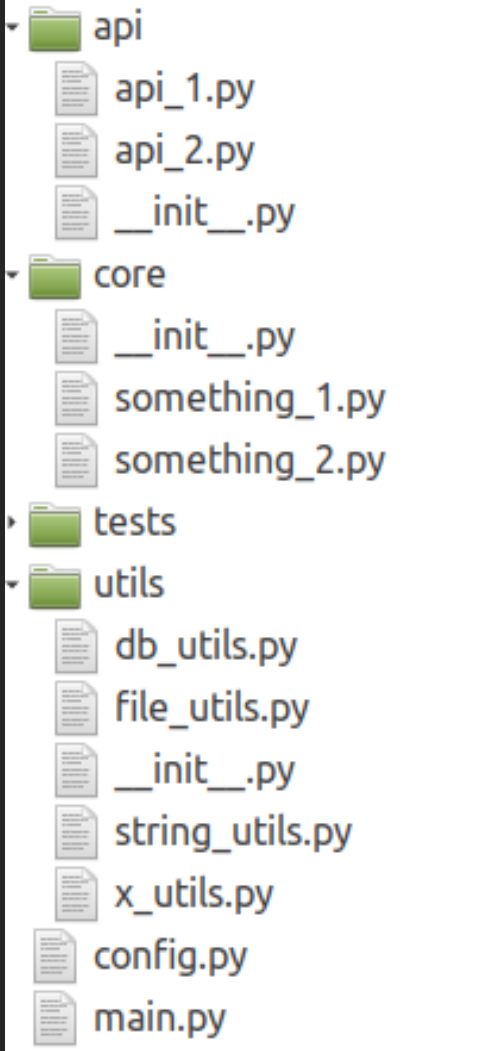
(3 not if, 3 return,12 lines)

```
if c1:
        work1t

        if ! c2:
                work2f
        else:
                work2t

                if c3:
                        work3t
                else:
                        work3f
else:
        work1f
```

(2 if,1 not if, 3 else, 12 lines)

Coding: modularize

- DRY : Do not Repeat Yourself
- SRP : Single Responsibility Principle
  - Responsibility = Reason for change
- OCP : Open Closed Principle
  - Open for extension, Closed for modification



Coding: have empathy !

- Try not to:
  - Re-invent the wheel
  - Leave a mess
  - Over-engineer
- Try to:
  - Use sensible names
  - Keep things modularized
  - Keep things understandable

Introduction

General

Coding

Collaboration

Summary

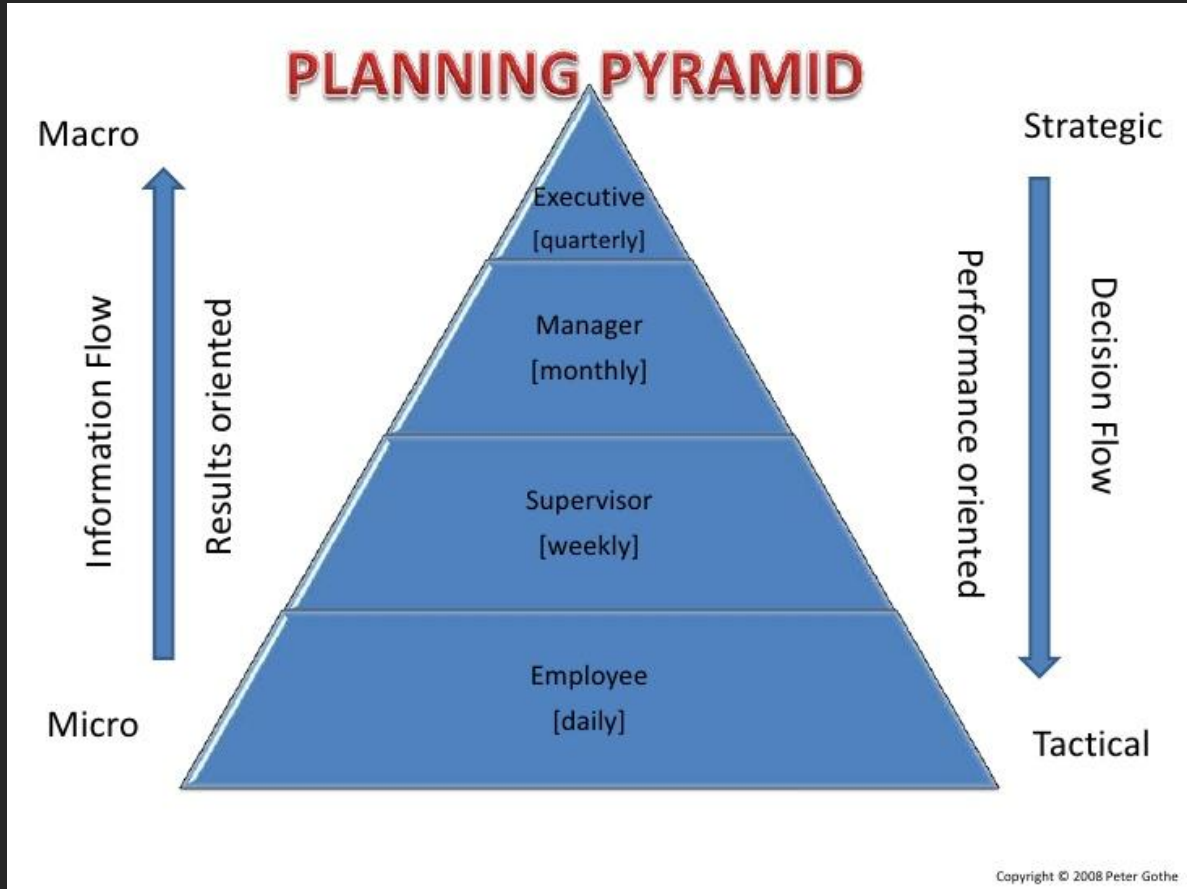
Collaboration: use your collaboration tool wisely!

- A collaboration tool is not
  - A traditional project management system
  - A documentation tool
- Boards
  - Specific purpose
  - State based columns/lists
- Cards
  - Simple & precise
  - Only one assignee per card
  - Checklist for verification
- Continuously reflect on tool usage

Collaboration: optimize information flow

Strategy should flow from top to bottom

Information should flow bottom to top



Introduction

General

Coding

Collaboration

Summary

Best Practices