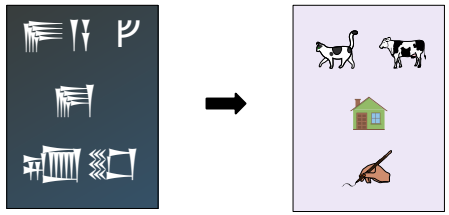
Eigenskills

Productive Programming / Cheat Sheet

[ver 1.1.0]
https://www.eigenskills.se/courses/prod-prog/

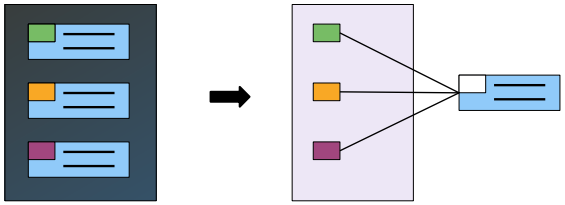
Core Principles

NAMES



Names should be meaningful and communicate intent.

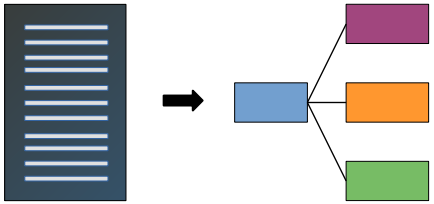
DRY



”Every piece of knowledge must have a single, unambiguous, and authoritative representation within a system.”

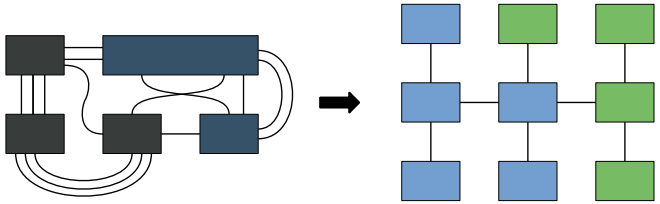
The Pragmatic Programmer - Hunt & Thomas

STORY



Code should be a human-readable description of a process, splitting the story on different levels.

DECOUPLE



Software should be built from self-contained components having a single concern and inter-coupled minimally.

Modular Design

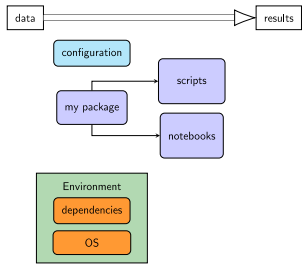
1. Start with the STORY, follow Core Principles
2. Think a few steps ahead and code defensively
3. Aim for a minimum viable product (MVP)
4. Test critical components, and test them well
5. Get early feedback: are you solving the right problem?
6. Iterate: repeat from 1. according to feedback

Modular Programming

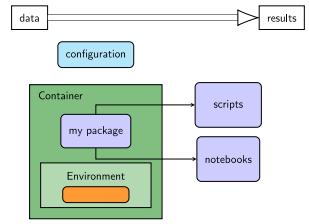
- Avoid global variables
- Keep configuration at high levels
- Stick to immutable data patterns whenever possible
- Give good NAMES
- Aim for simple connections between components
- Let modules handle their own internal logic
- Move back-end and helper code from your notebooks to external packages
- Set random number generator seeds explicitly

Deployment

Local (ad-hoc) dev



Remote dev



Testing Psychology

Assumption is the mother of all screw-ups

- Treat tests as critical thinking about code behavior
- Mind your blind spots: we have a tendency to test for expected behavior
  - Have counterexamples in your tests
  - Include edge cases and extreme or atypical values

Testing Complex Stuff

Treat your code as a subject of study

- Formulate hypotheses and try to falsify them
- Be explicit about assumptions
- Identify pre- and post-conditions, invariants, and relationships between components
- Test qualitative properties of stochastic code