

Software Craftsmanship Practices

or How to make your job delightful

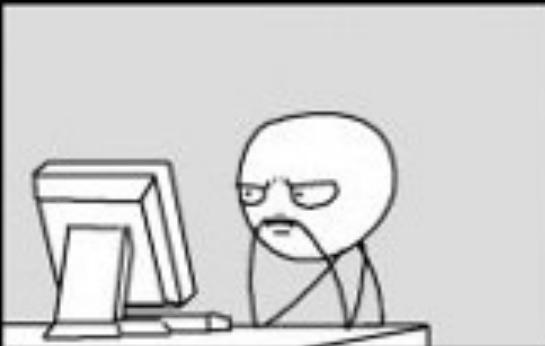
Best Practice is a technique or methodology that, through experience and research, has proven to reliably lead to the desired result.



Software Engineers



What Society Believes I Do



What My Wife Believes I Do



What My Mom Believes I Do



What My Boss Believes I Do

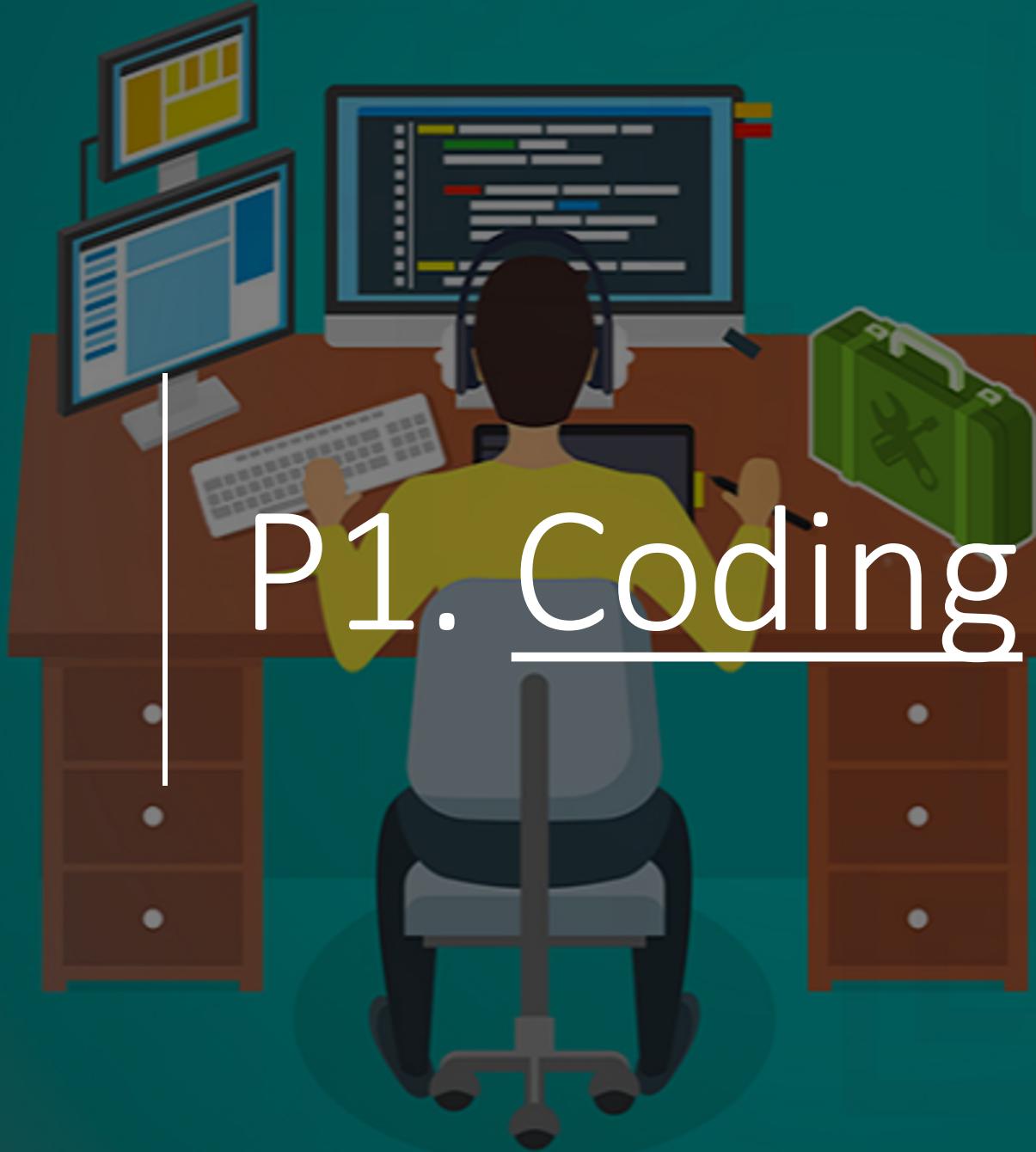


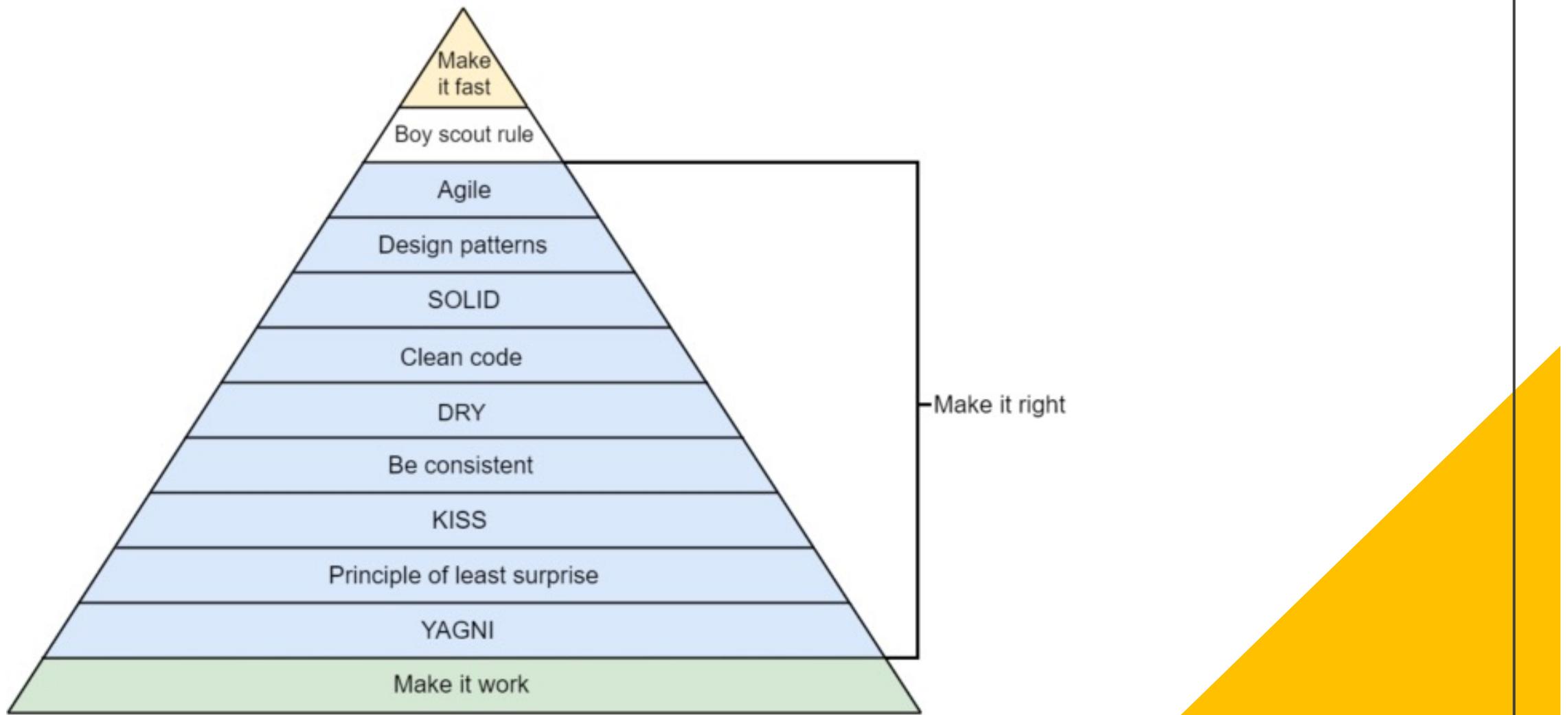
What I Believe I Do



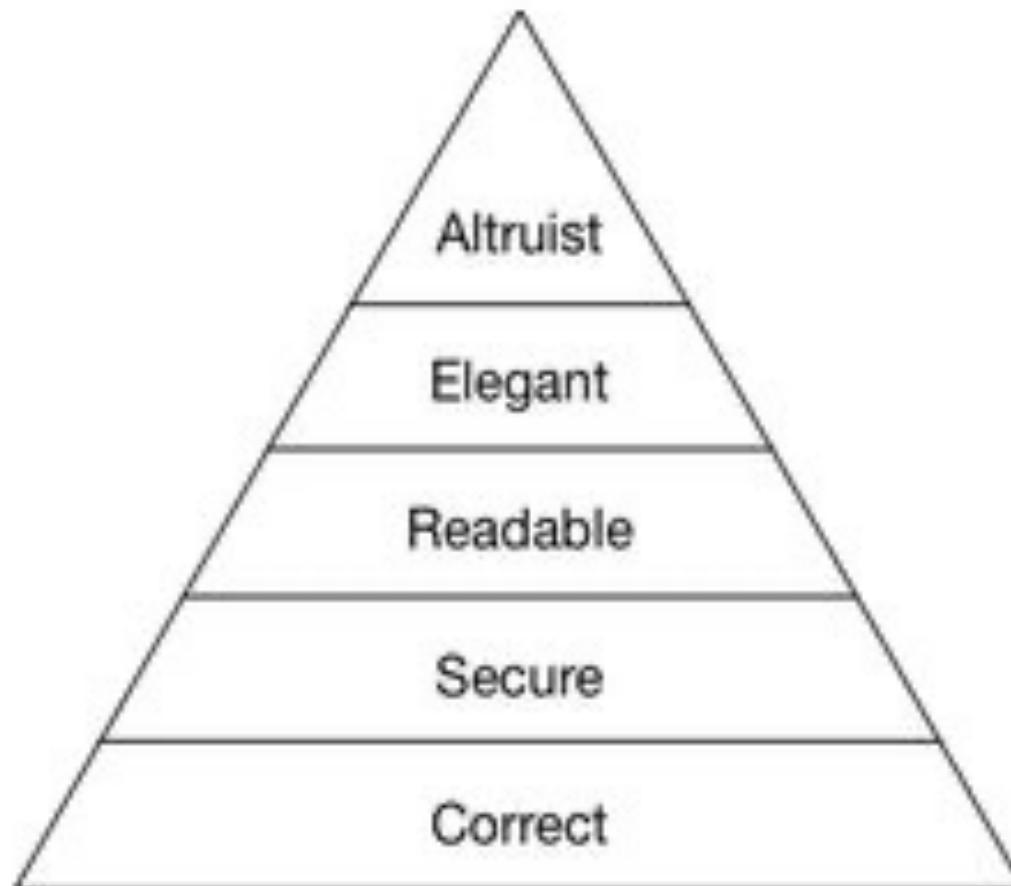
What I Really Do

P1. Coding





When you open a Pull Request, you're Saying "My
Code Is Production Ready"





Ellen Shapiro @designatednerd · 18h



there's ordering an elevator with extra buttons in case you decide to build more floors.



4



3



34

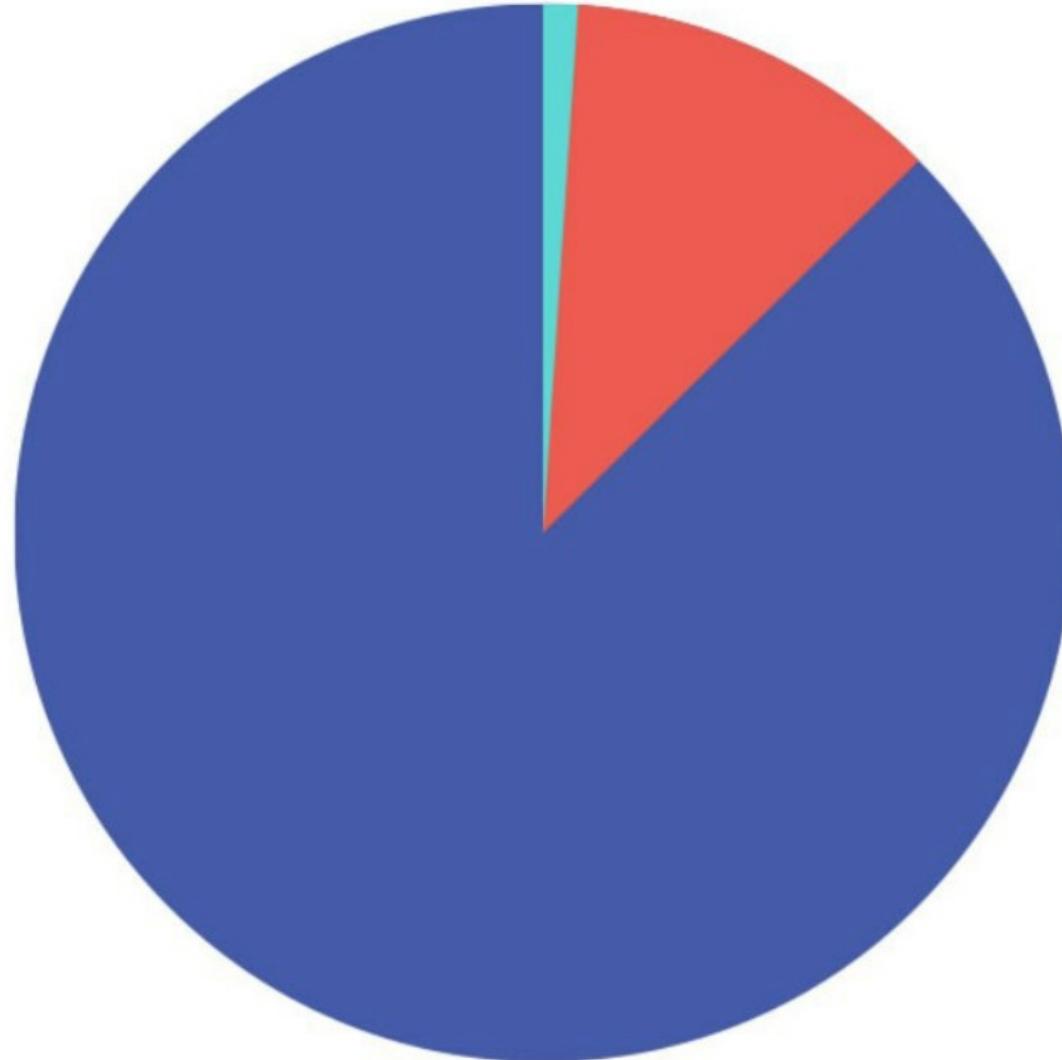


Maybe a ticket got merged in from the backlog



Principle Of Least Surprise

CLEAN
VARIABLE
NAMES



Coding



Debugging



Deciding what this
variable should be
named



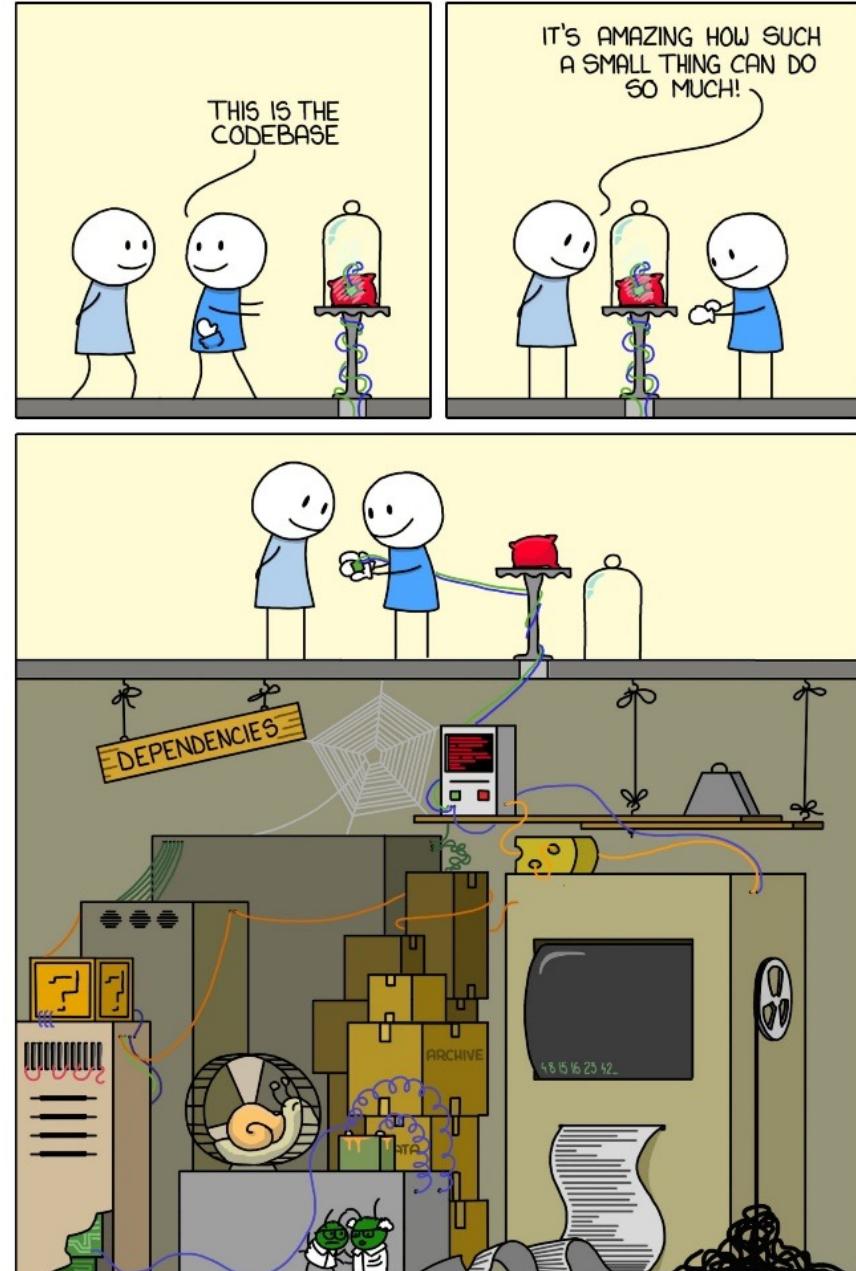
“ ”

Master programmers think of
systems as stories to be told
rather than programs to be
written

-Robert C. Martin

Keep It Simple

IMPLEMENTATION



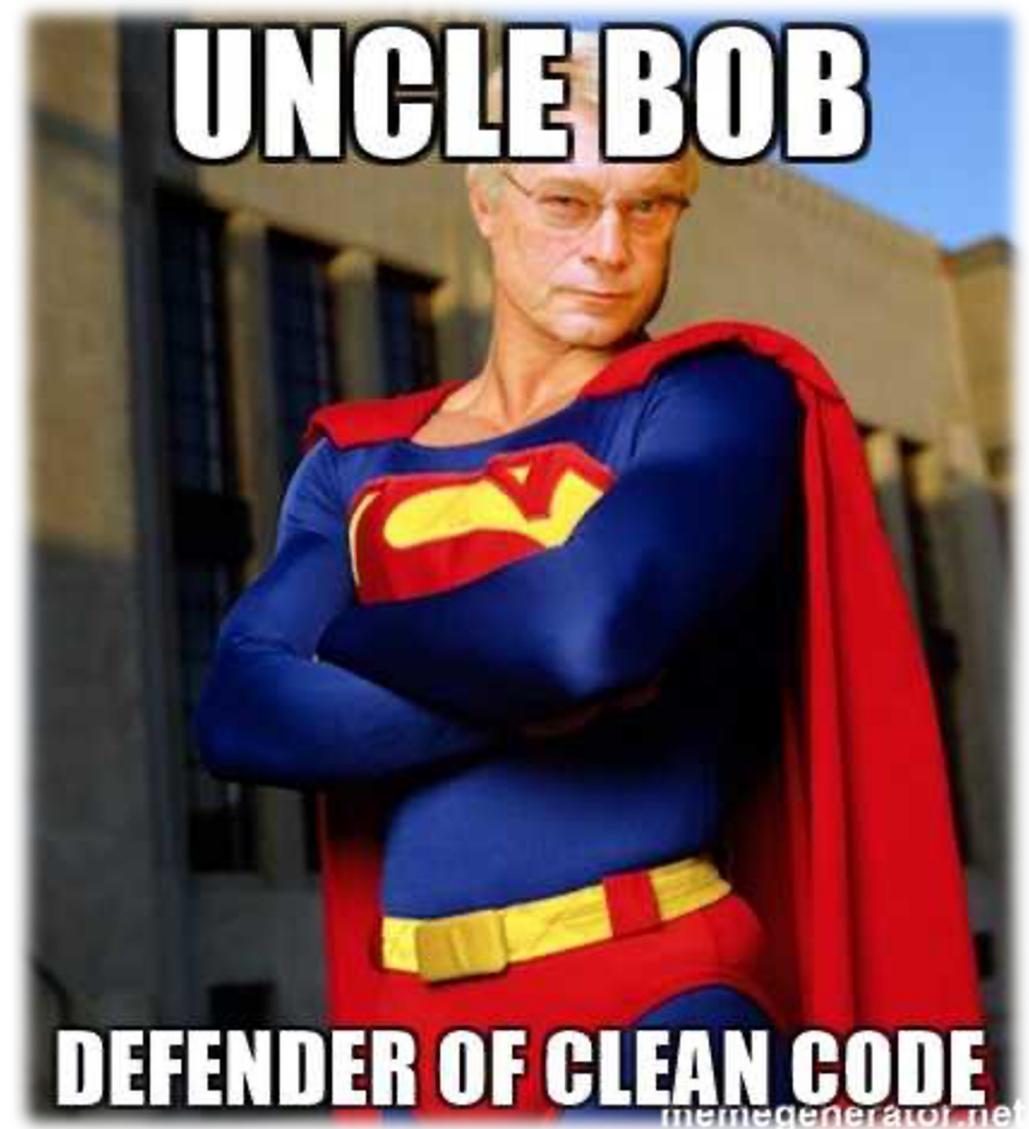
Do not
Repeat
Yorself

I will not repeat myself.
I will not repeat myself.
I will not repeat myself.
I will not repeat myself.



"Clean code is simple and direct. Clean code reads like well-written prose. Clean code never obscures the designer's intent but rather is full of crisp abstractions and straightforward lines of control."

Robert "Uncle Bob" Martin



Single
Responsibility

Liskov's
Substitution
Principle

Dependency
Inversion

Open/Closed
Principle

Interface
Segregation

S.O.L.I.D.



Software Design Patterns

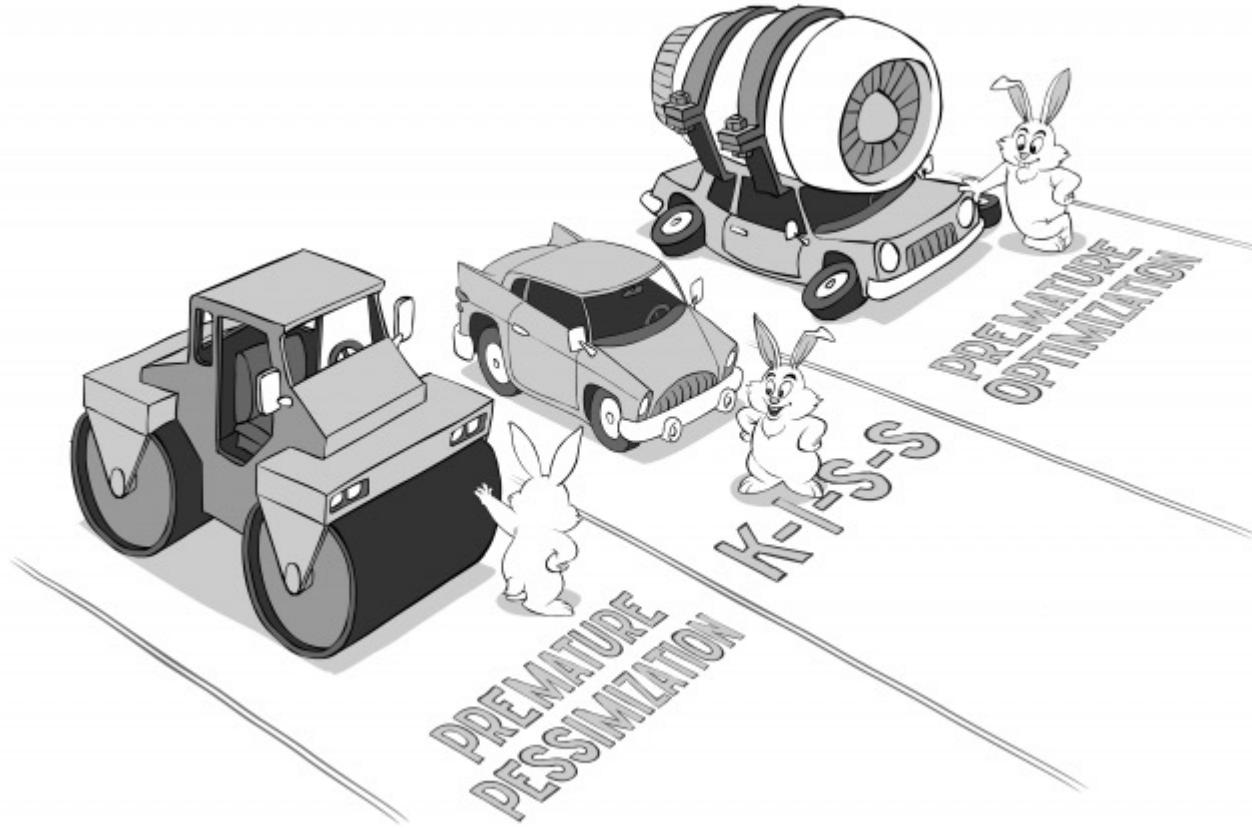
THE BOY SCOUT RULE



ALWAYS
LEAVE
CODE
CLEANER
THAN YOU
FOUND IT!

The Boy
Scout Rule

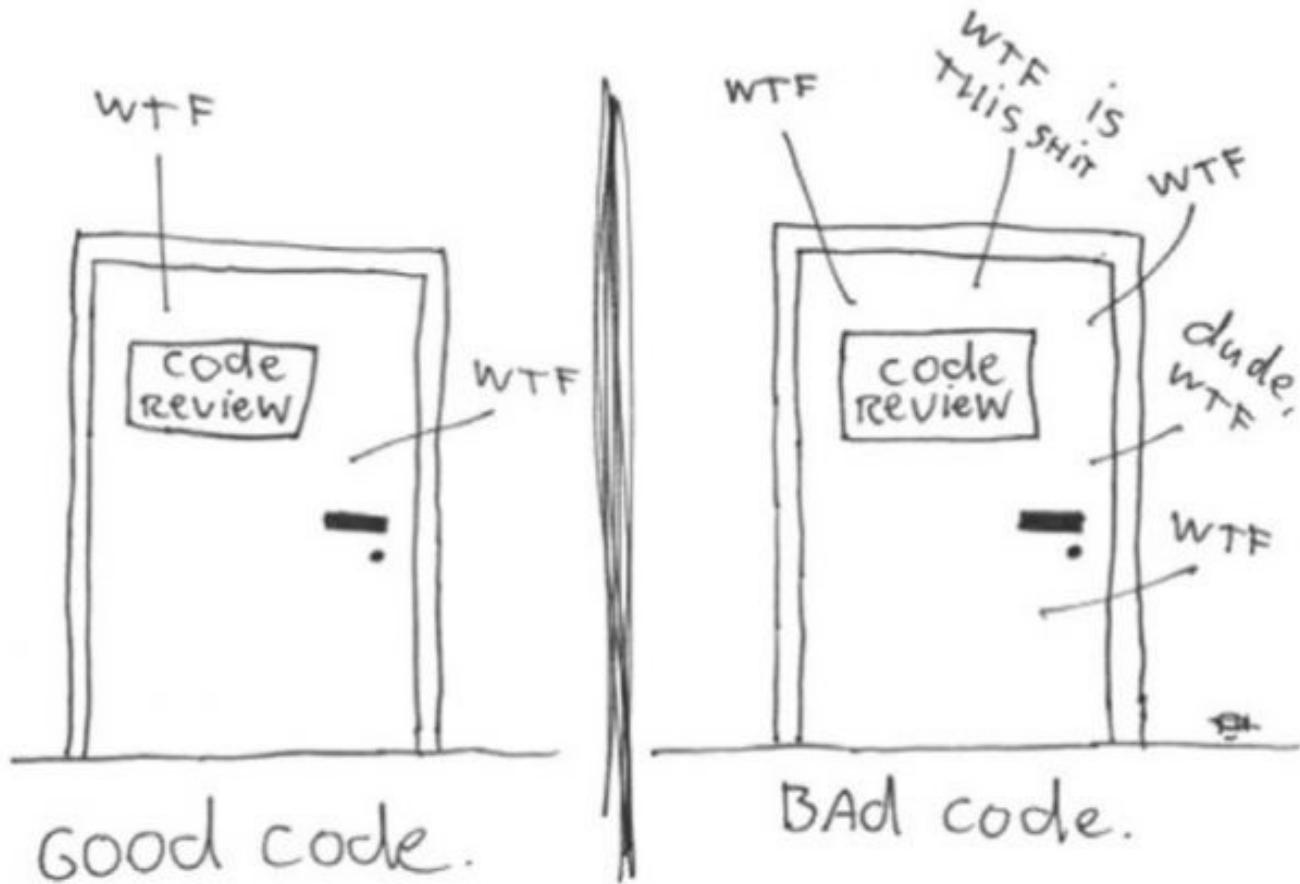
Avoid Premature Optimization



Code review.

Quality measurement

The ONLY VALID MEASUREMENT
OF CODE QUALITY: WTFs/MINUTE



Reproduced with the kind permission of Thom Holwerda.
http://www.osnews.com/story/19266/WTFs_m



Self-documented
code: What, Why
and How

Write Tests

Software Testing Best Practices



- Test Early, Test Often
- Test Coverage and Code Coverage
- Automate Testing
- Testable Requirements
- End to End Testing
- Bug Prevention

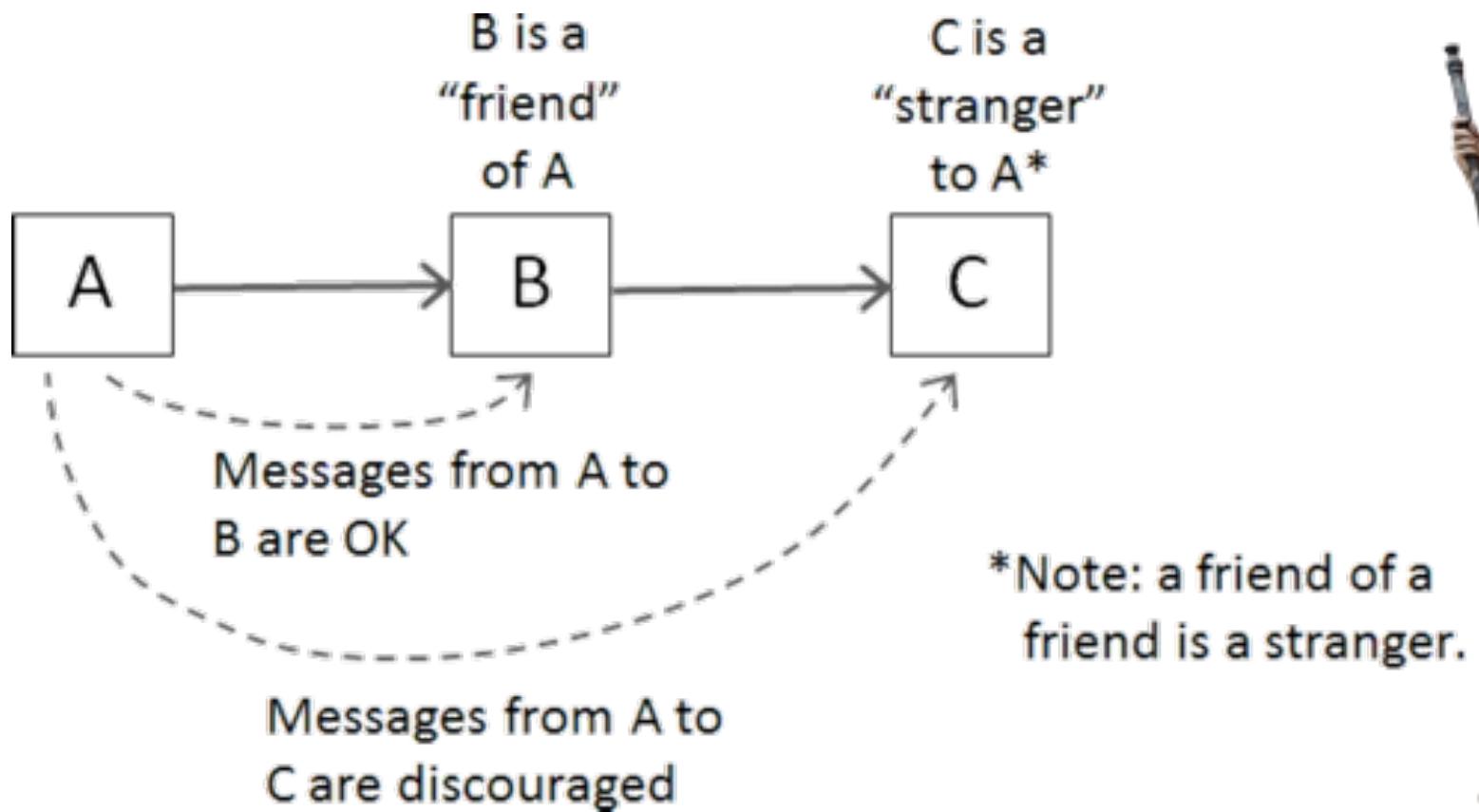




```
try {  
    something  
} catch(e) {  
    window.location.href =  
        "http://stackoverflow.com/search?q=[js] + "  
        + e.message;  
}
```

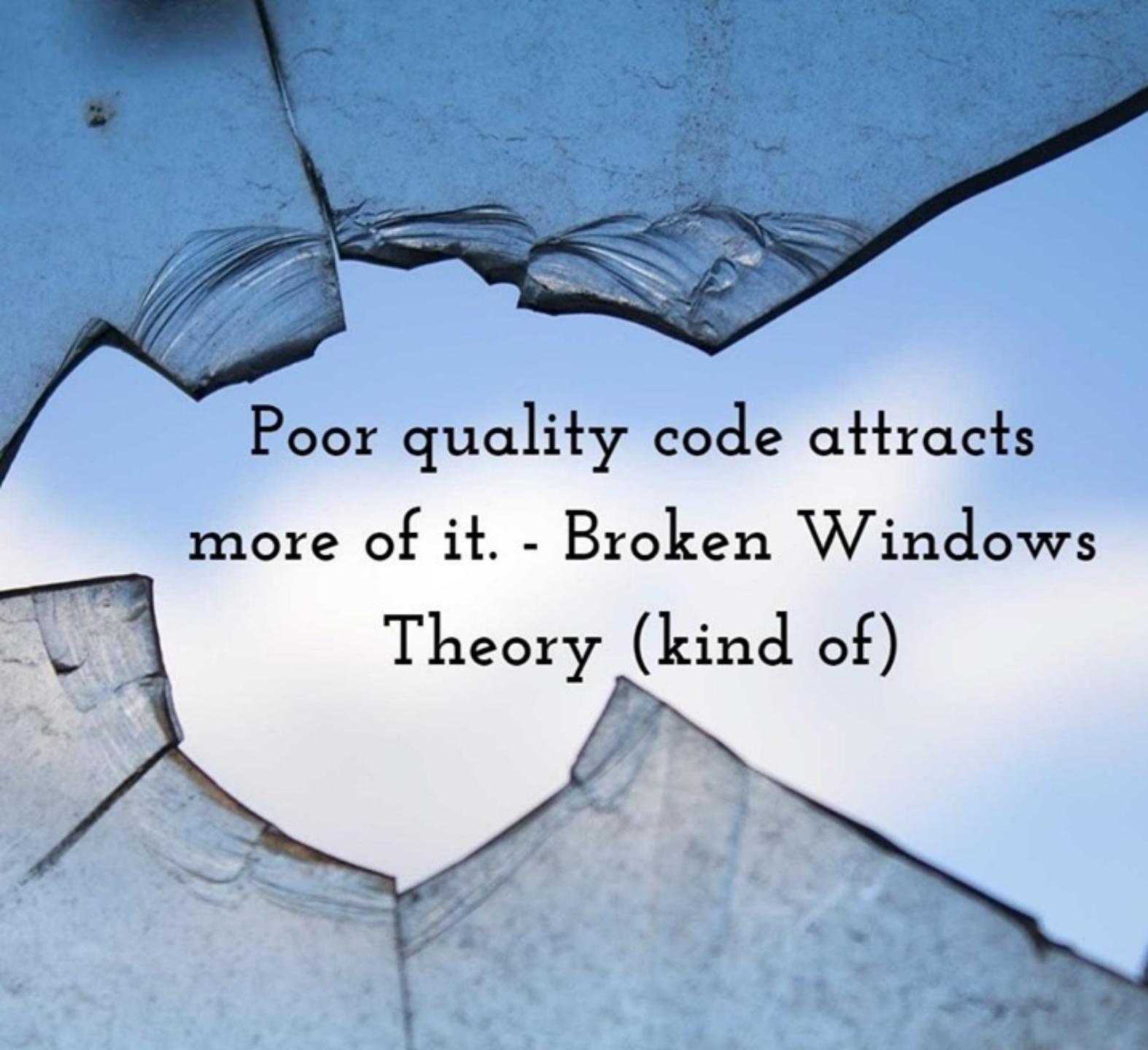
Copy and Paste carefully

Law of Demeter





App logging



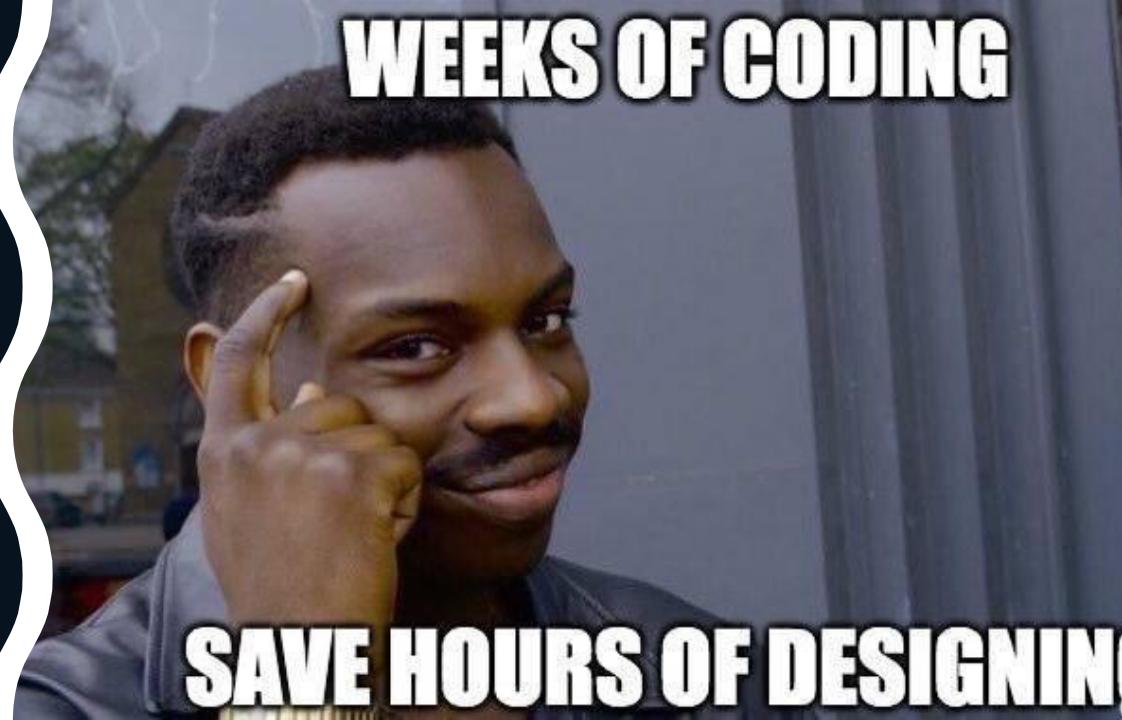
Poor quality code attracts
more of it. - Broken Windows
Theory (kind of)

**Broken
windows
theory**

P2. Thinking



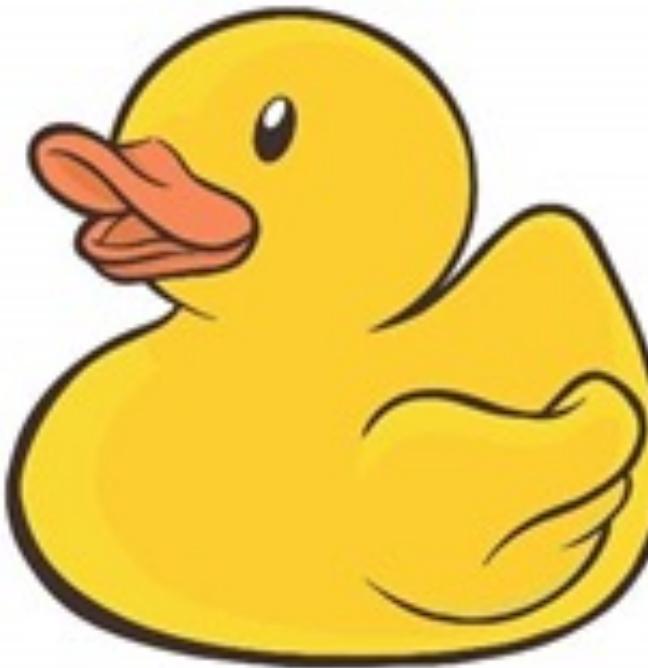
Think
twice,
do once



SAVE HOURS OF DESIGNING



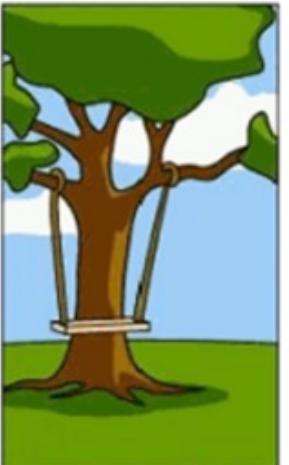
Stuck?
Talk to
your
duck!



**HAVE
YOU
TRIED
EXPLAINING
IT TO THE
RUBBER DUCK?**



How the customer explained it



How the project leader understood it



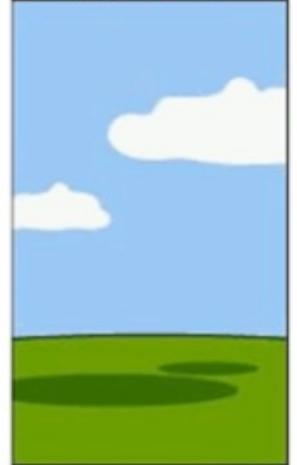
How the engineer designed it



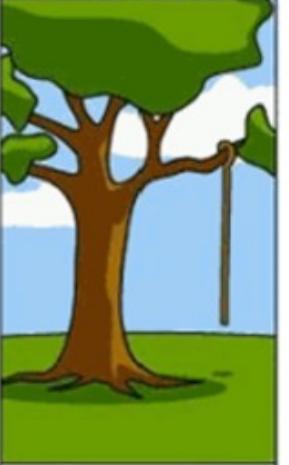
How the programmer wrote it



How the sales executive described it



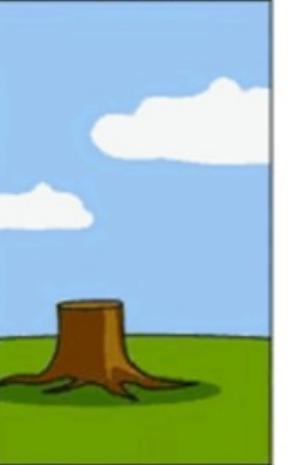
How the project was documented



What operations installed



How the customer was billed



How the helpdesk supported it



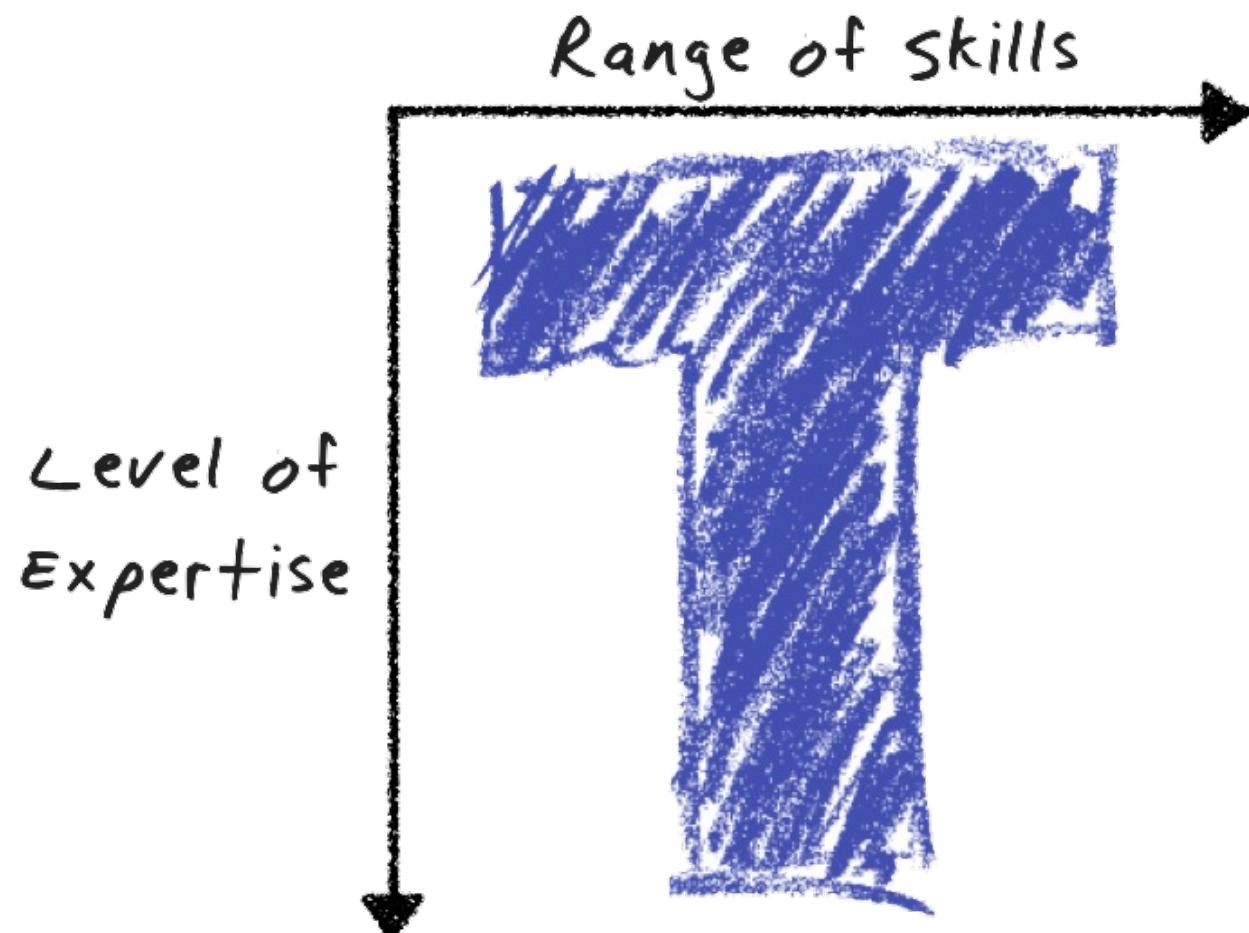
What the customer really needed

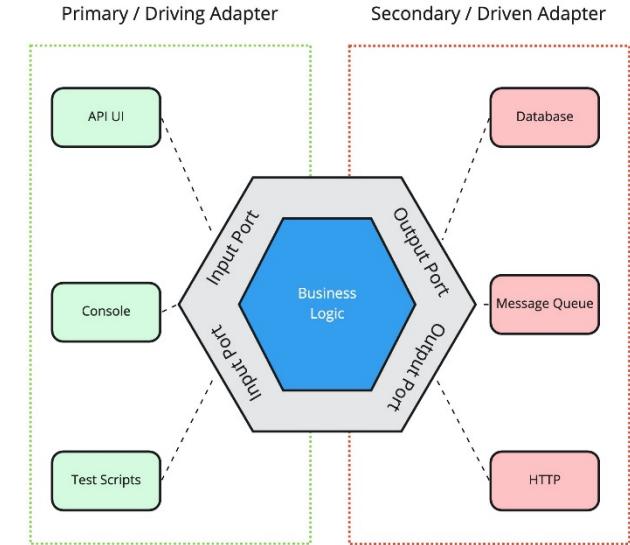
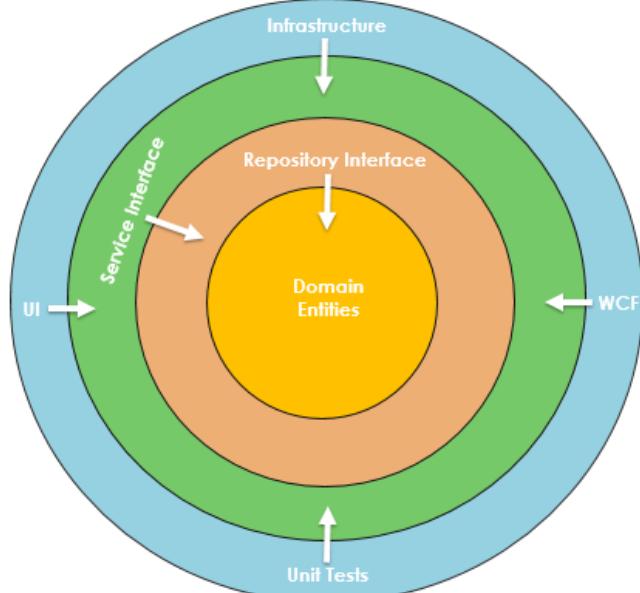
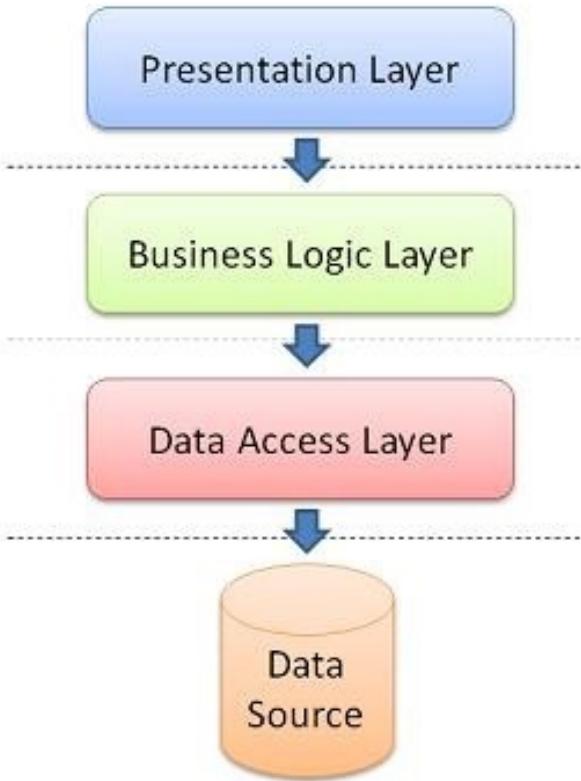
Communicate
to the
customer



Team-work, conventions\standards

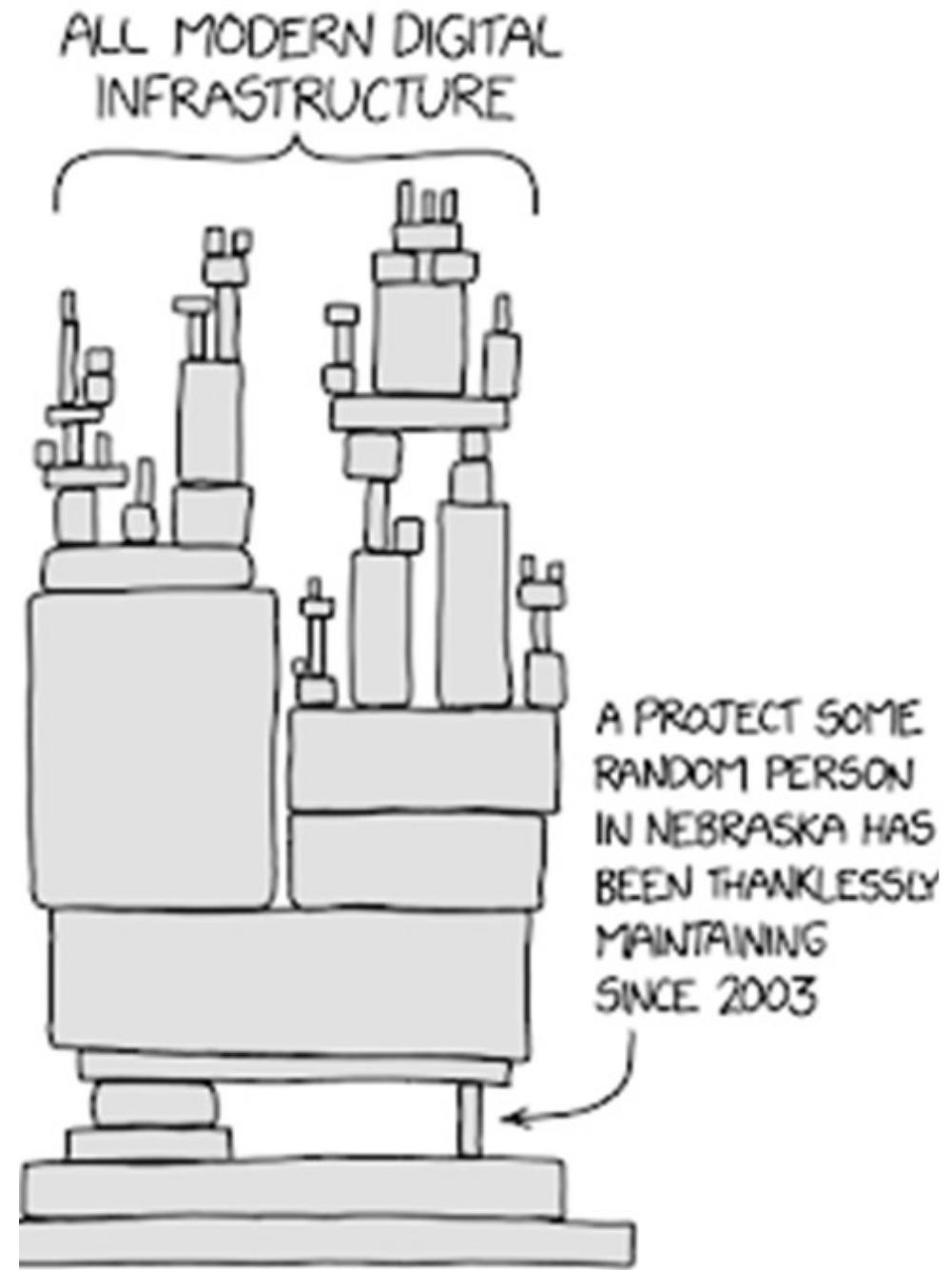
Diversify Your Knowledge: T-Shaped





Architecture: N-Tier, Onion, Hexagonal,...

*Technology is only
how you get to the
solution, it is not
THE solution*



TASK DESCRIPTION vs. EFFORT

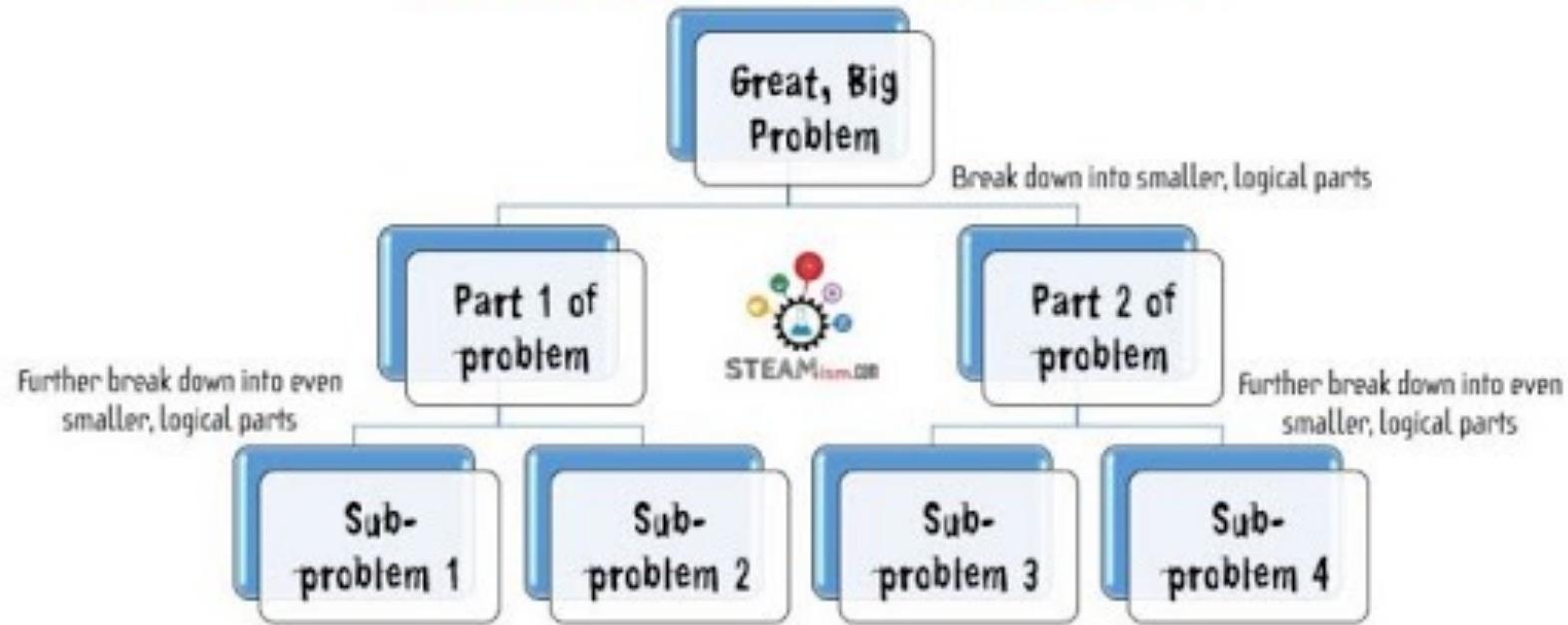


Estimate
goals

Set Deadlines

"The man who moves a mountain begins by carrying away small stones."
—Confucius

DECOMPOSITION



Setting goals



Do: Set real numbers with real deadlines.

Don't: Say, "I want more visitors."

Do: Make sure your goal is trackable.

Don't: Hide behind buzzwords like, "brand engagement," or, "social influence."

Do: Work towards a goal that is challenging, but possible.

Don't: Try to take over the world in one night.

Do: Be honest with yourself- you know what you and your team are capable of.

Don't: Forget any hurdles you may have to overcome.

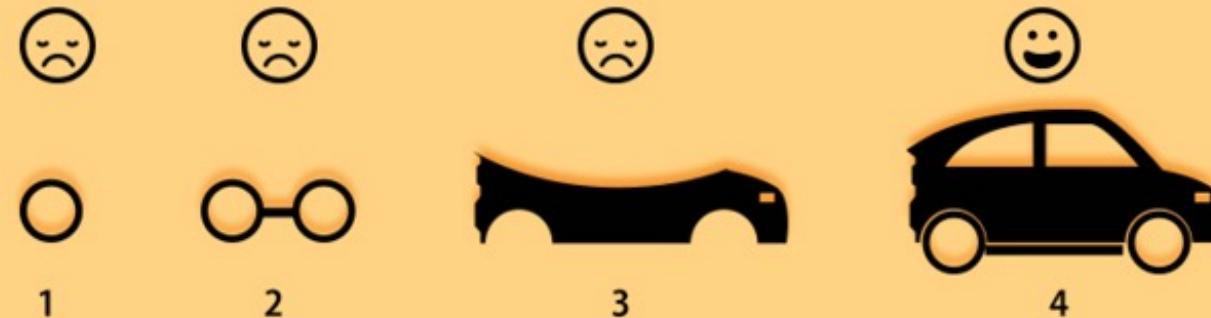
Do: Give yourself a deadline.

Don't: Keep pushing towards a goal you might hit, "some day."

There are
no ideal
solutions

HOW TO BUILD A MINIMUM VIABLE PRODUCT

NOT LIKE THIS



LIKE THIS

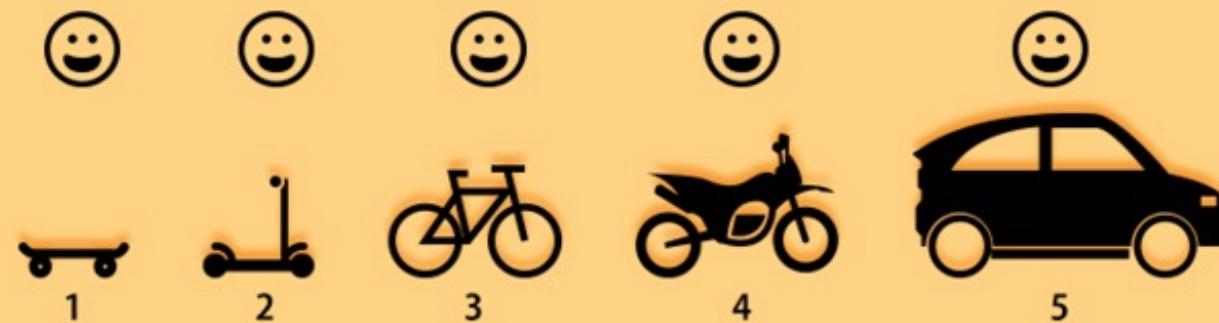
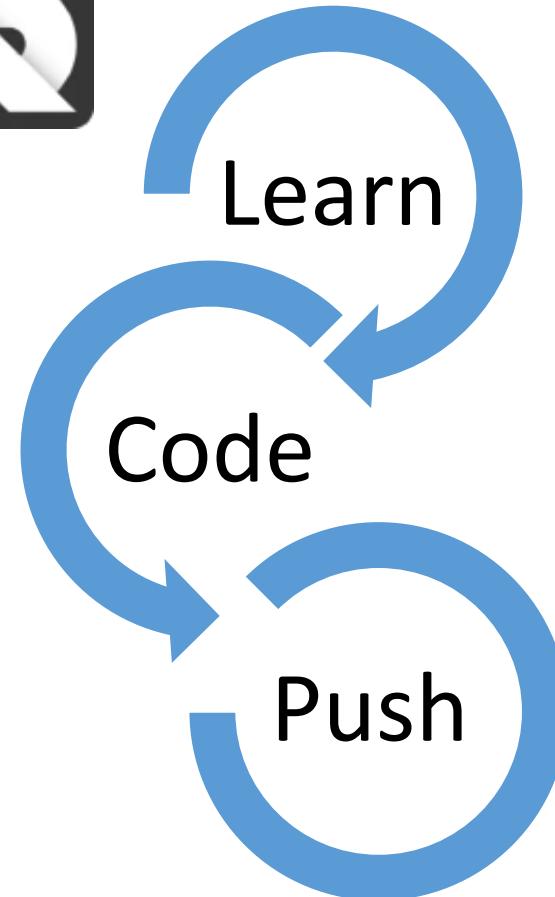


image by blog.fastmonkeys.com original idea: spotify product team



CLEAN CODERS

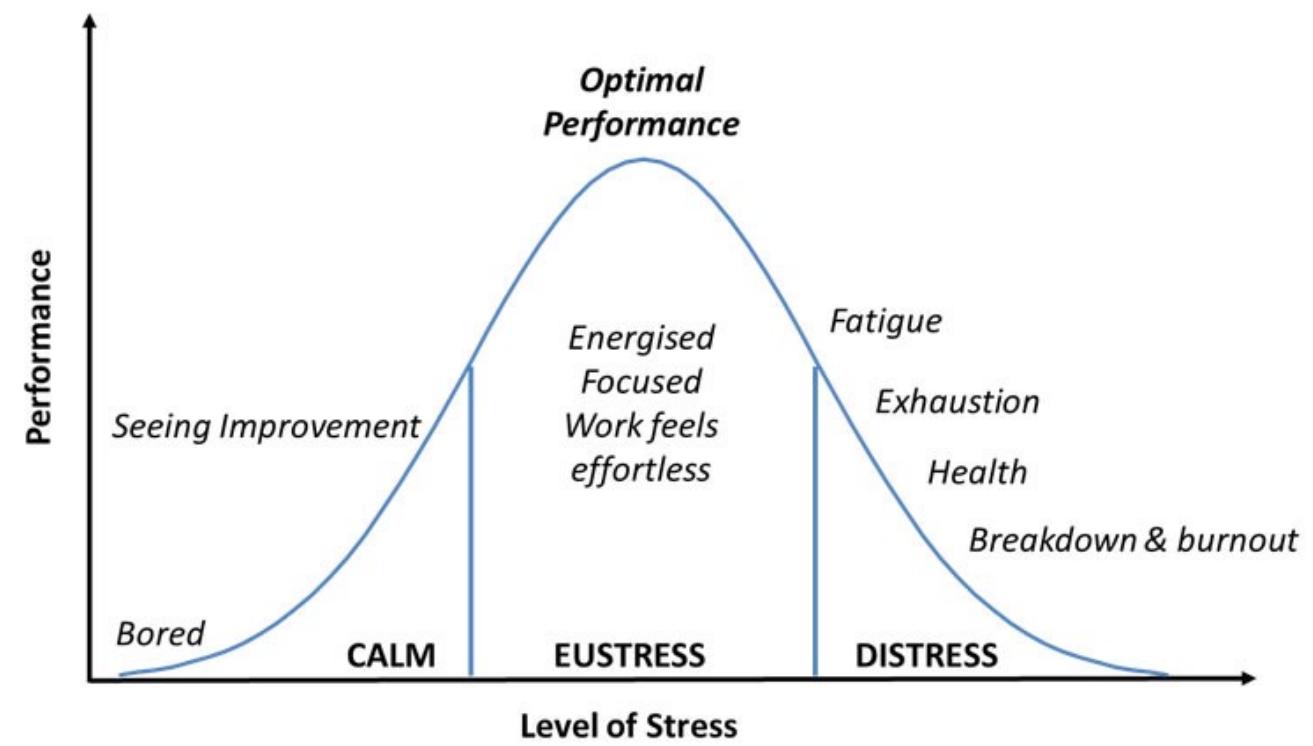
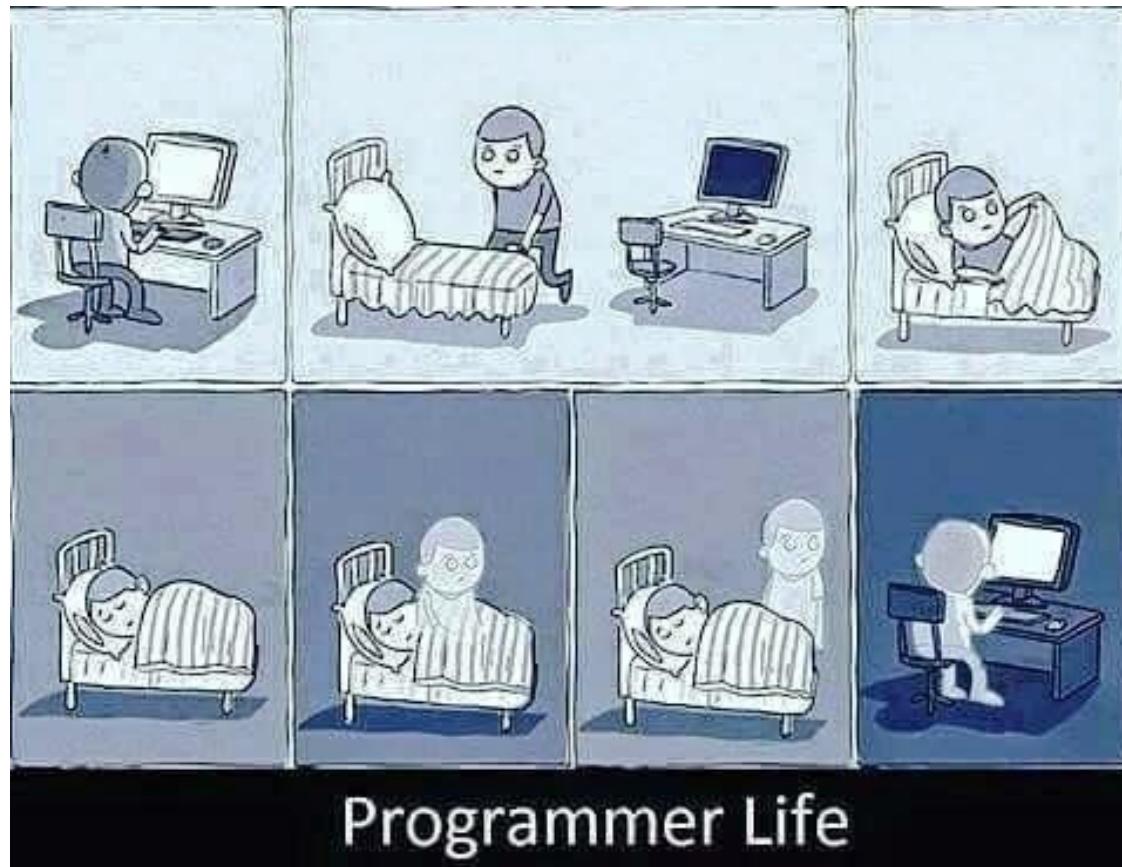


HackerRank



LeetCode

Work-life balance





Best
practices
are
context
dependent

Read with
pleasure





Thank you