"Oh laugh, Curtin, old boy. It's a great joke played on us by the Lord, or fate, or nature, whatever you prefer. But whoever or whatever played it certainly had a sense of humor! Ha!"

— *Howard*, The Treasure of the Sierra Madre

The Clean Coder
Chapter 1: Professionalism

Thursday, January 26th

Advice from the Book

- What is a software professional?
- How does a professional behave?
- How does a professional deal with conflict, tight schedules, and unreasonable managers?
- When, and how, should a professional say “no”?
- How does a professional deal with pressure?

Be Careful What You Ask For

- Responsibility and Accountability
- Nonprofessionals don’t have to take responsibility for the job they do—they leave that to their employers.
- If a nonprofessional makes an error, the employer cleans up the mess, But when a professional makes a mistake, he cleans up the mess.

Volkswagon Engineer Sentenced to 40 Months in Prison + $200,000 fine(2017)

- James Liang designed and developed software related to emissions;
  - Designed to underreport emissions levels
    - Hardware + software temporarily reduced emissions if emission detection device was plugged in; once removed, emissions exceeded allowance
  - Huge recall
- Yotam Lurie: It's shocking that the software engineers of Volkswagen overlooked and neglected their fiduciary responsibility as professionals. Professionals who have a semi-regulatory responsibility within the organization to ensure safety, in this case environmental safety, even when this is less efficient or economical.

First, Do No Harm

- Do No Harm to Function
  - Harm the function of our software when we create bugs
  - Apologize and do not make the errors again
  - Errors should rapidly decrease
  - Take responsibility for bugs and fixing them
- QA Should Find Nothing
  - Unprofessional to purposely send code that you know is faulty or you have not tested
  - When QA finds bugs figure out why those bugs managed to escape your notice and do something to prevent it from happening again
  - Every time anyone finds a bug you should strive to prevent it from happening again

First, Do No Harm Cont.

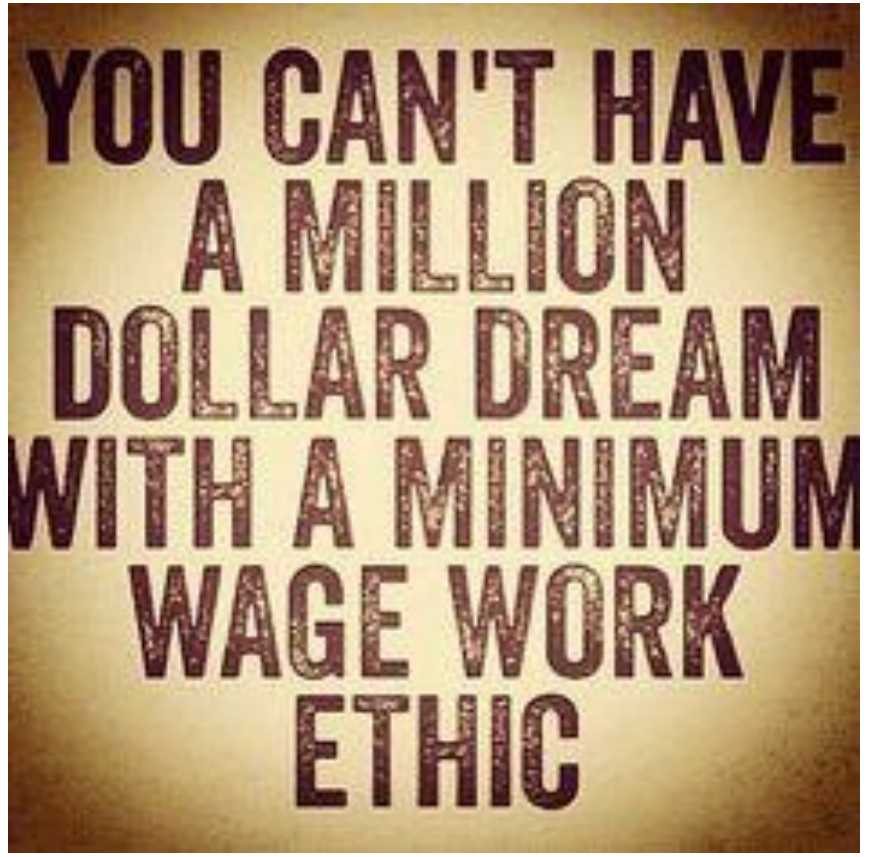
- You must know it works
  - Test your code!
  - Automate your tests, write unit tests, and run the tests often
  - Every line of code should be tested
  - Write your tests first, before you write the code that passes them (Test Driven Development)
- Automated QA
  - Entire QA procedure is the execution of the unit and acceptance tests
    - Mission-critical systems automated tests are insufficient

First, Do No Harm Con't.

- Do No Harm to Structure
  - Delivering function at the expense of structure
  - Software should be easy to change with out exorbitant costs
  - Adding additional resources will not solve the problem
  - Refine the design so the next change is easier
    - Every time you read through the code you adjust the structure (refactoring)
    - Always check in the module cleaner when you checked it out
  - Afraid they are going to break something
  - Continuously shape and mold it (bad smells)

Work Ethic

- Your career is your responsibility
  - Employers may be willing to buy you books, training classes, or conferences
  - The time you need to learn
- You owe your employer a certain amount of time and effort
- Plan to working 60 hours a week (40 for employer 20 for you)
  - Professionals spend time caring for their profession



Know your field

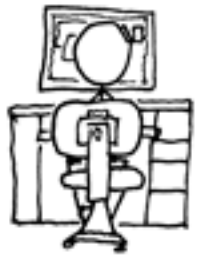
- Last 50 years have provided a wealth of ideas, disciplines, techniques, tools, and terminologies
- Hard-won ideas of the last 50 years are as valuable today as they were then
- List of items that every software professional should be conversant with:
  - Design patterns. You ought to be able to describe all 24 patterns in the GOF book and have a working knowledge of many of the patterns in the POSA books.
  - Design principles. You should know the SOLID principles and have a good understanding of the component principles.
  - Methods. You should understand XP, Scrum, Lean, Kanban, Waterfall, Structured Analysis, and Structured Design.
  - Disciplines. You should practice TDD, Object-Oriented design, Structured Programming, Continuous Integration, and Pair Programming.
  - Artifacts: You should know how to use: UML, DFDs, Structure Charts, Petri Nets, State Transition Diagrams and Tables, flow charts, and decision tables.

Continuous Learning

- Must continue to learn to keep up with a growing field
- Read books, articles, blogs, tweets, go to conferences, user groups
- Work to continually expand knowledge

Practice

- True professionals work hard to keep their skills sharp and ready
- Practices when you specifically exercise your skills outside of the performance of your job
  - Simple programming problems to solve



Collaboration and Mentoring

- Lean by collaborating with other people
  - Program together, practice together, design, and plan together
- The best way to learn is to teach
- Professionals take personal responsibility for mentoring juniors

Know Your Domain

- Responsibility of every software professional to understand the domain of the solutions they are programming
  - Know your industry
  - When starting a project in a new domain, read a book or two on the topic
  - Interview customers
  - Spend time with experts and try to understand their principles and values
- Unprofessional to code from a spec without understanding why that spec makes sense to the business

Identify with your employer/customer

- Your employer's problems are your problems
- Put yourself in your employer's shoes
- Professionals avoid the them vs. us mentality

Humility

- Professionals know they are arrogant and are not falsely humble
- A professional knows his job and takes pride in his work
- A professional is confident in his abilities, and takes bold and calculated risks based on that confidence
- A professional is not timid
- A professional makes mistakes
  - Never ridicules others, but will accept ridicule when it is deserved and laugh it off when it’s not
  - Not demean another for making a mistake, because he knows he may be next to fail

What we learned from this chapter

- A professional takes responsibility
- Test your code, QA should find no issues
- Your career is your responsibility and you should do things to help further it (classes, books, practice, etc.) and contiguously learn
- Take the time to learn about your field
- Collaborating, teaching, or mentoring is a great way test your skills and learn from others
- Know the domain of the solutions you are programing
- Show humility and respect to your other fellow employees

Word of the Day:

1001 - Space Sticker
1002 – Planet Sticker

Professional Tip of the Day: The Myth of Unlimited Time Off

- A way for employers to not pay out your earned time off when you leave a position
- Check your time off policy
- Subject to manager deaccession and approval

Professional Tip of the Day #2

- Does your company have a 401k or retirement plan?
  - Does your company do matching funds?
- If I have crazy amounts of debt and or am super poor should I invest in my company's 401K plan?
  - Money comes out pre-tax
  - Invest $5,000 at age 22, with an annual rate of return of 4%, your employer matching 100% of the 6% of your salary, and an annual plan contribution of 6%