DevOps for Defense

January 2021

Code Archeology: How to know if Refactoring is right for me

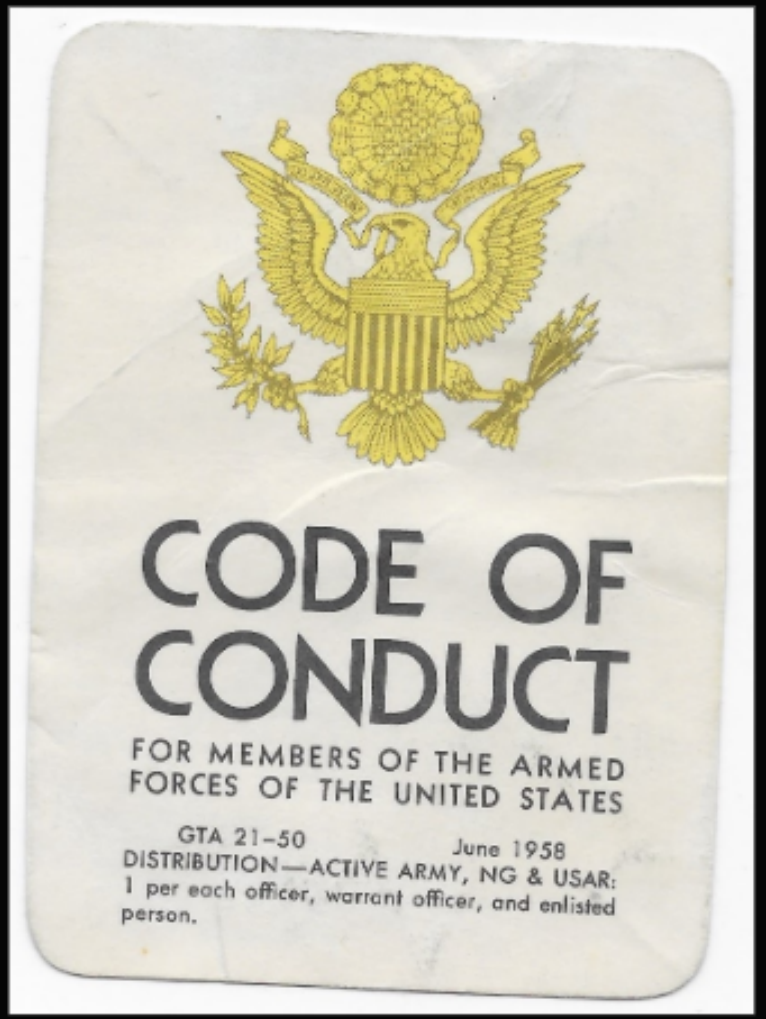
Karen Hirsch, PhD

https://devopsfordefense.org
https://www.meetup.com/DevOps-for-Defense/
https://github.com/jondavid-black/DevOpsForDefense
devopsfordefense@gmail.com
https://twitter.com/devops4defense

Sponsored by: TEKsystems Own change

DevOps for Defense Meetup: Code of Conduct

- UNCLASSIFIED ONLY!!!!
- Treat each other with respect and professionalism.
- Do not talk about private, sensitive, or proprietary work.
- Do talk about your experiences, needs, desires to improve work in our domain.
- Do share your thoughts.
- Do learn from others.
- Do mute yourself while others are speaking!



CODE ARCHEOLOGY: HOW TO KNOW IF REFACTORING IS RIGHT FOR ME

Karen Hirsch, PhD

FIRST A SHORT PERSONAL HISTORY

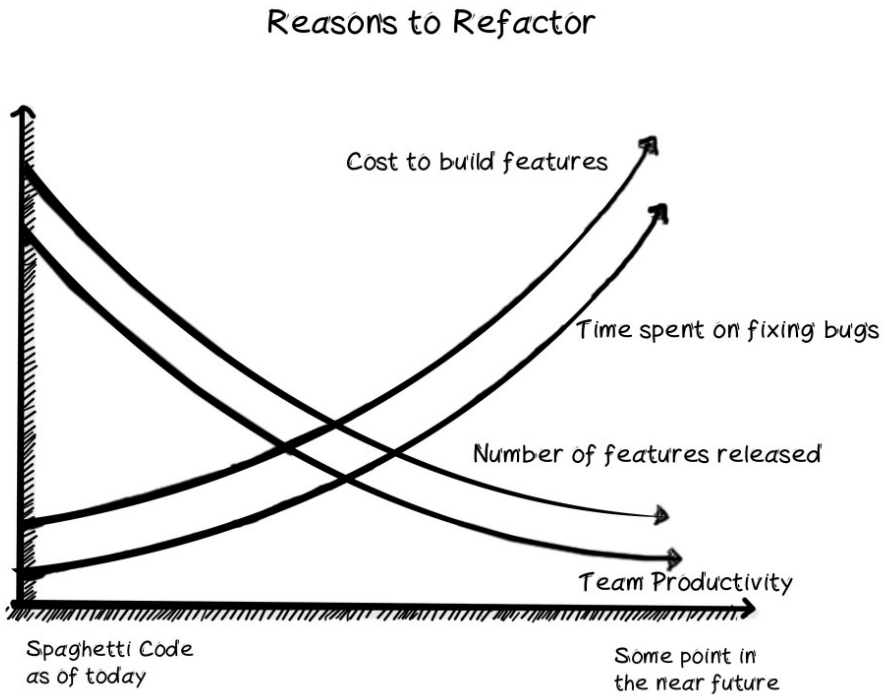
SO, WHAT HAS THAT GOT TO DO WITH REFACTORING?

Definition:

- (*noun*) A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior
- (*verb*) to restructure software by applying a series of refactorings without changing its observable behavior

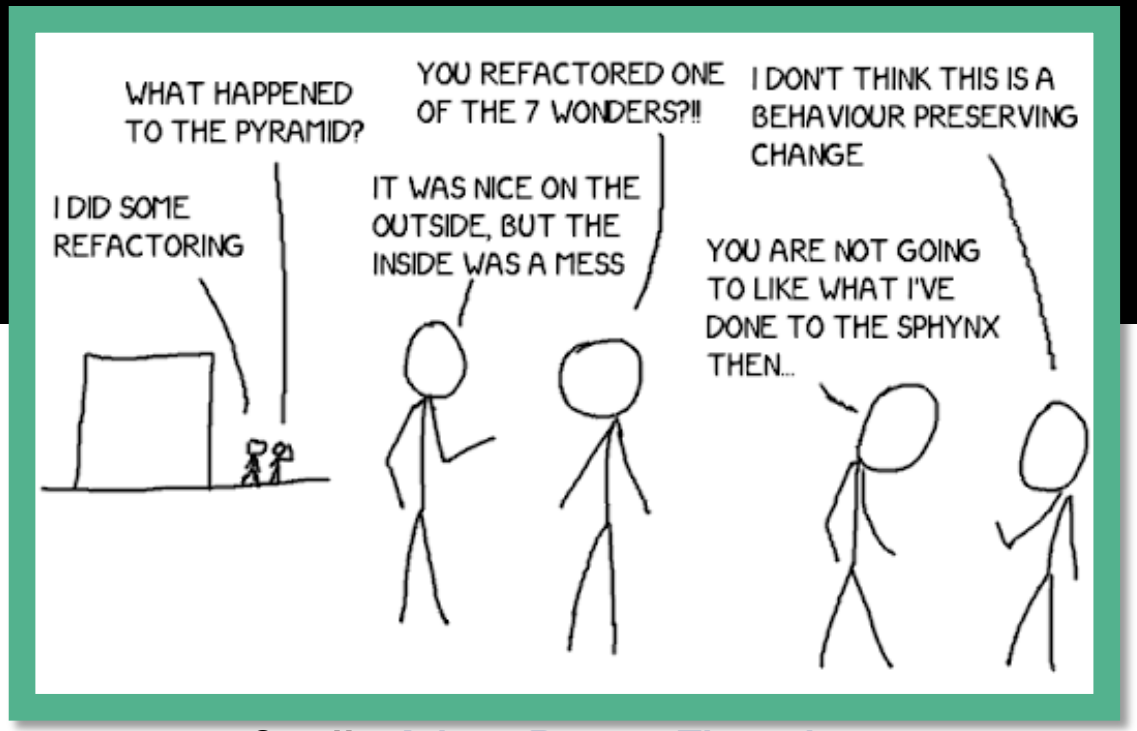
Fowler, Refactoring: Improving the Design of Existing Code,

Pearson Education, Inc., 2019. Pg 45



Credit: Tarun Kohli blog post

WHY CHANGE LEGACY CODE?



Credit: [Adam Bowen Thoughts on Software Blog]

- Add feature/capability
- Bug fix
- Improve design
- Optimize resource usage

NOTE: all those reasons above are really the reasons we touch old code at all - none of them suggest ‘perfecting product’

WHEN TO REFACTOR

It is time to refactor when the cost of code archeology to figure out when/how to change the code is higher than cleaning it up

Additional signs might include current logic structure doesn't support new features needed

Credit: XKCD comic

BUT WAIT – IN DEV*OPS WE NEED AUTOMATED TESTS

In order to refactor (or really work on the code at all) we need to know what its' current behavior is and ensure we build automated tests to allow us to confirm that that behavior isn't changing

Don't use "Edit and Pray" use "Cover and Modify"

(names courtesy of Feathers, Working Effectively with Legacy Code,

Pearson Education, Inc., 2005, pg 9.)

WHICH TESTS NEED TO BE BUILT?

Unit Tests

Behavioral tests

To build them follow these guidelines

1. Identify change points
2. Find test points
3. Break dependencies
4. Write tests

Credit: Reddit

Feathers, Working Effectively with Legacy Code, Pearson Education, Inc., 2005, pg 18

WAIT – DON'T ADD *ALL* THE TESTS

It is tempting to do it right

It is tempting to do it all

It is tempting to do it all right now –

...but that is not what we are trying to achieve

WAIT – DON'T ADD *ALL* THE TESTS

It is tempting to do it right

It is tempting to do it all

It is tempting to do it all right now –

…but that is not what we are trying to achieve

OKAY - TEST ADDED, NOW WHAT?

Code Archeology –This might have been done to construct the test

Choose correct reason/rhyme for refactor (taken from Fowler)

- Preparatory
- Comprehension
- Opportunistic

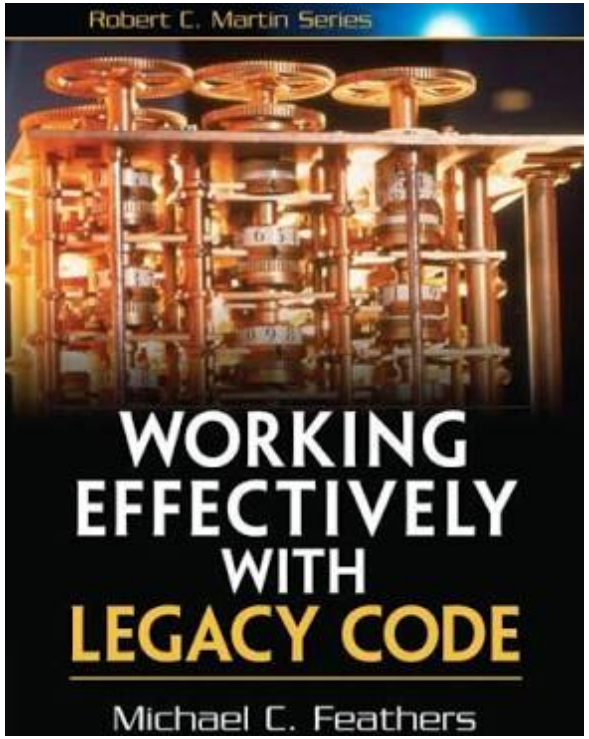
NOW WHAT? (CONT'D)

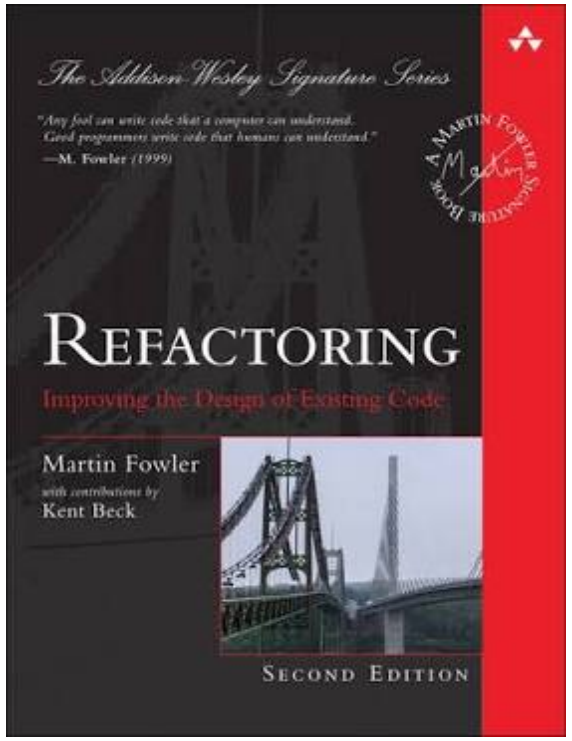
Things that might make life better include

- Interfaces that are simpler
- Data cleaning/simplification
- Removing duplicated codes
- Simpler functions to aid future code archeologists, etc.

Indiana Jones, *Lucas Films*

ARE THERE MORE ANSWERS?





Plus:

- Blog posts
- Reddit boards
- Interesting YouTube talks
- Etc.

DevOps Resources

https://devopsfordefense.org/resources/

https://www.meetup.com/DevOps-for-Defense/
https://github.com/jondavid-black/DevOpsForDefense
devopsfordefense@gmail.com

Books / Publications:

Conference Presentations (YouTube):

- DevOps Enterprise Summit (DOES)
- IT Revolution
- Velocity
- GoTo