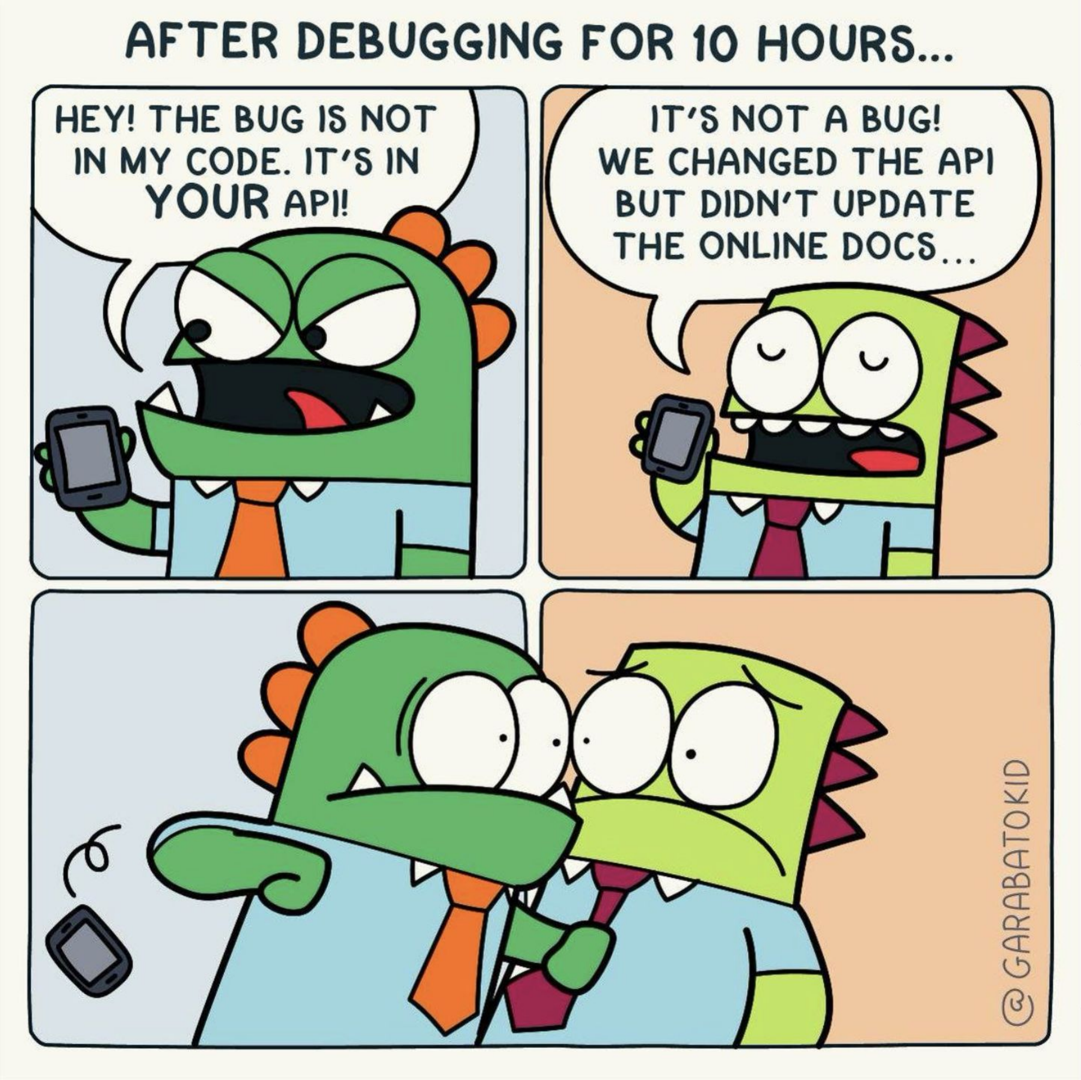
"To #comment or not?"

A data-driven look at conflicting attitudes towards commenting and documentation.

@veronica_hanus #PyDataNYC


AFTER DEBUGGING FOR 10 HOURS...
HEY! THE BUG IS NOT IN MY CODE. IT'S IN YOUR API!
IT'S NOT A BUG! WE CHANGED THE API BUT DIDN'T UPDATE THE ONLINE DOCS...
@GARABATOKID

@veronica_hanus

Comments as a learner’s first documentation

Purpose of comments:

➔ **Summary**: authorship, purpose

➔ **Describe** functions

➔ **Clearify** “tricky code”/unclear decisions

Allowing:

➔ **Easier stand-alone doc-writing!**

➔ **A “notepad”**

➔ **Tell why!**

Comments get a pretty bad rep!

Code should speak for itself
Comments rot
Keep it dry
Comments mean time to refactor

@veronica_hanus

When does refactoring happen?

- ➔ **New work > refactoring**
- ➔ **Time for feedback?**
- ➔ **Feel like a “nice to have”**

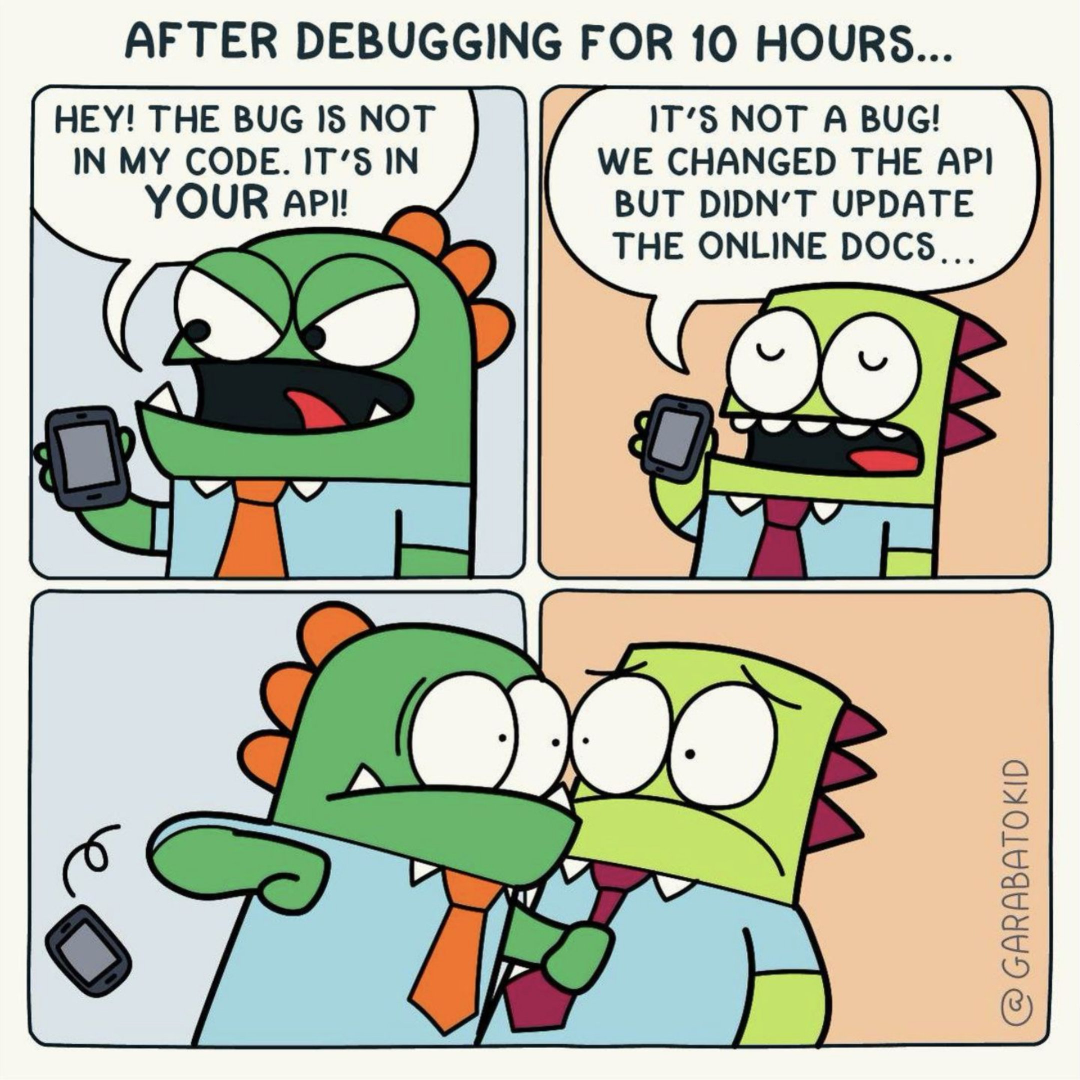
Refactor when….

- ➔ **Can’t stand it anymore**
- ➔ **Problem grows**

Refactoring == self-care??

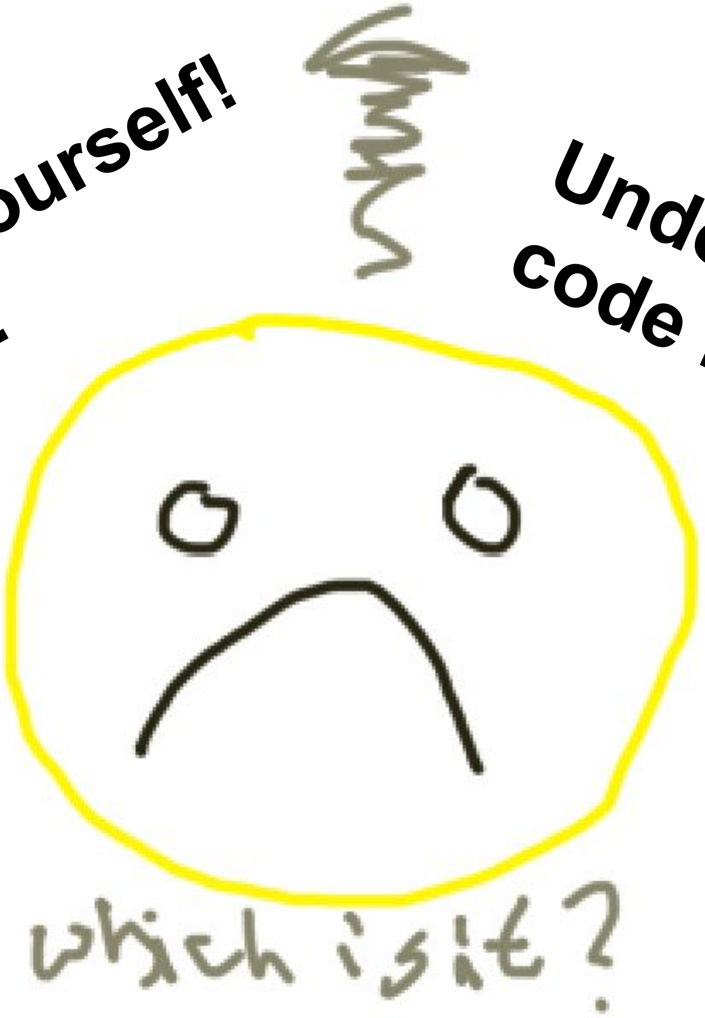
@veronica_hanus

Finally! Time to refactor!


AFTER DEBUGGING FOR 10 HOURS...
HEY! THE BUG IS NOT IN MY CODE. IT'S IN YOUR API!
IT'S NOT A BUG! WE CHANGED THE API BUT DIDN'T UPDATE THE ONLINE DOCS...
@GARABATOKID

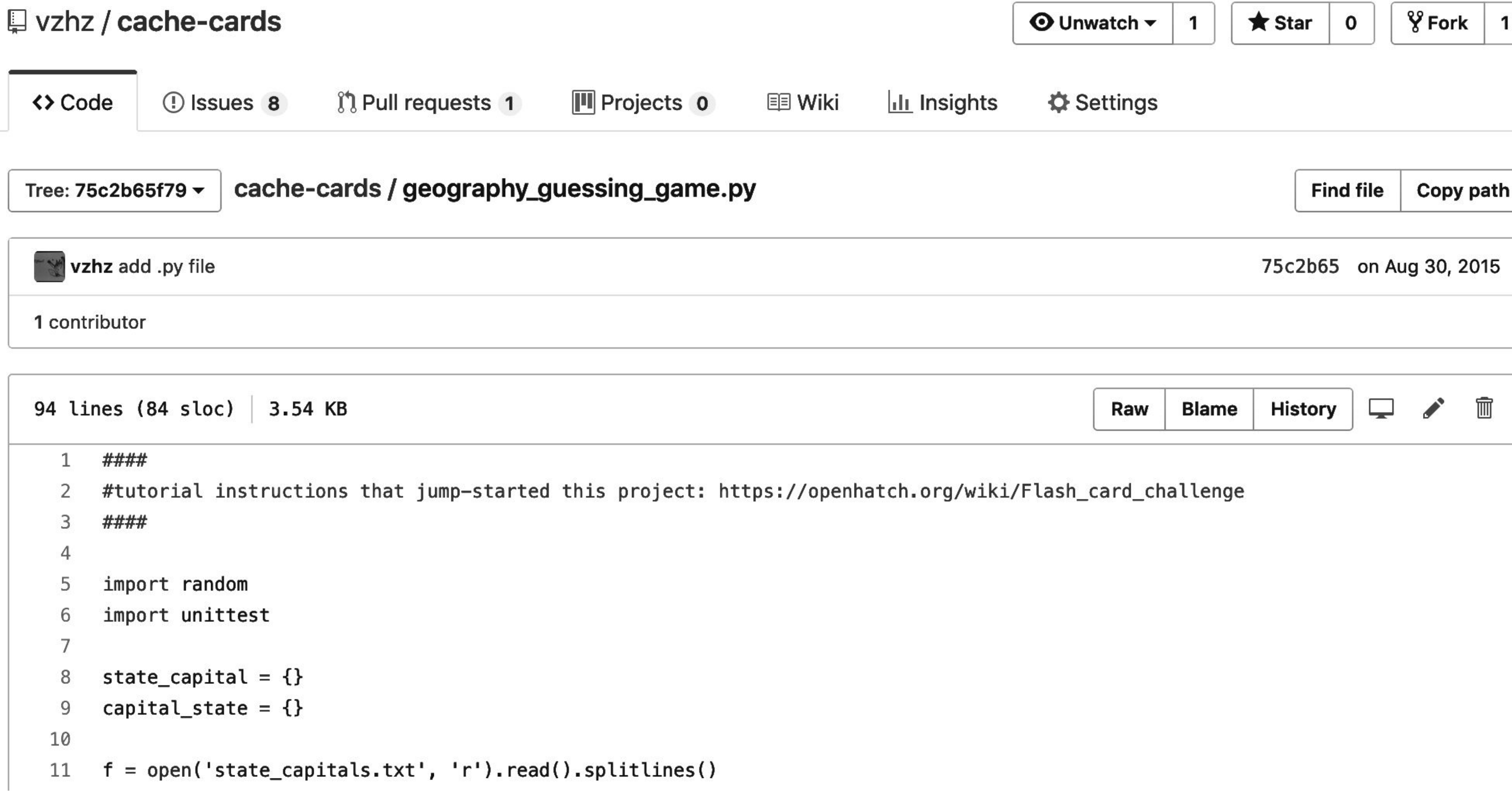
@veronica_hanus

Don’t Repeat Yourself!
Keep it D.R.Y.!



Undocumented code is unusable!

@veronica_hanus



vzhz / cache-cards

cache-cards / geography_guessing_game.py

vzhz add .py file 75c2b65 on Aug 30, 2015

1 contributor

94 lines (84 sloc) | 3.54 KB

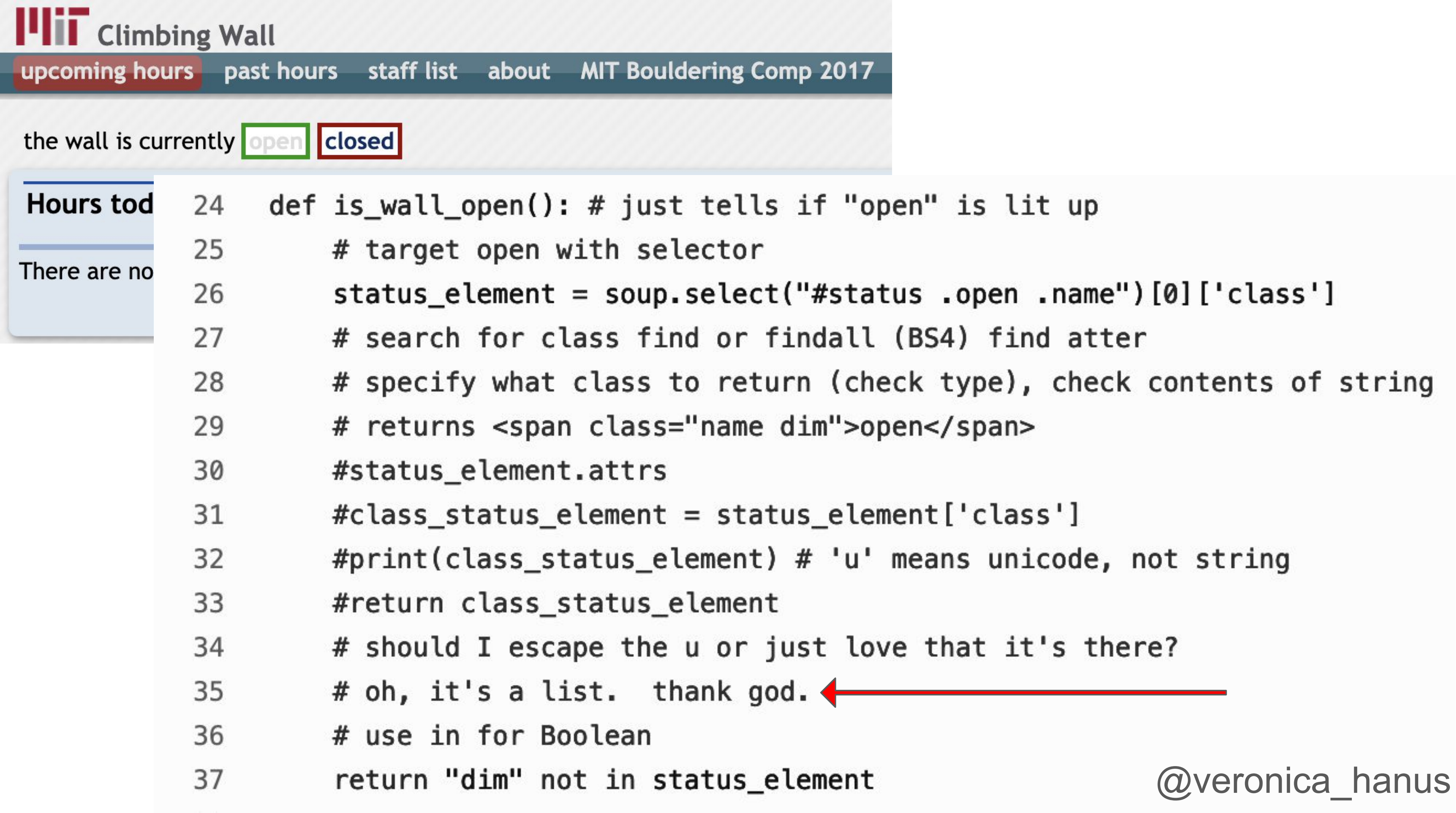
```
####
#tutorial instructions that jump-started this project: https://openhatch.org/wiki/Flash_card_challenge
####

import random
import unittest

state_capital = {}
capital_state = {}

f = open('state_capitals.txt', 'r').read().splitlines()
```

@veronica_hanus



the wall is currently open closed

Hours tod

There are no

```
def is_wall_open(): # just tells if "open" is lit up
    # target open with selector
    status_element = soup.select("#status .open .name")[0]['class']
    # search for class find or findall (BS4) find atter
    # specify what class to return (check type), check contents of string
    # returns <span class="name dim">open</span>
    #status_element.attrs
    #class_status_element = status_element['class']
    #print(class_status_element) # 'u' means unicode, not string
    #return class_status_element
    # should I escape the u or just love that it's there?
    # oh, it's a list.  thank god.
    # use in for Boolean
    return "dim" not in status_element
```

@veronica_hanus

@veronica_hanus

Best practices we can (usually) get behind

Always!

Docstrings should have inputs, outputs, transformation

Outdated comments == lies

Too much is too much

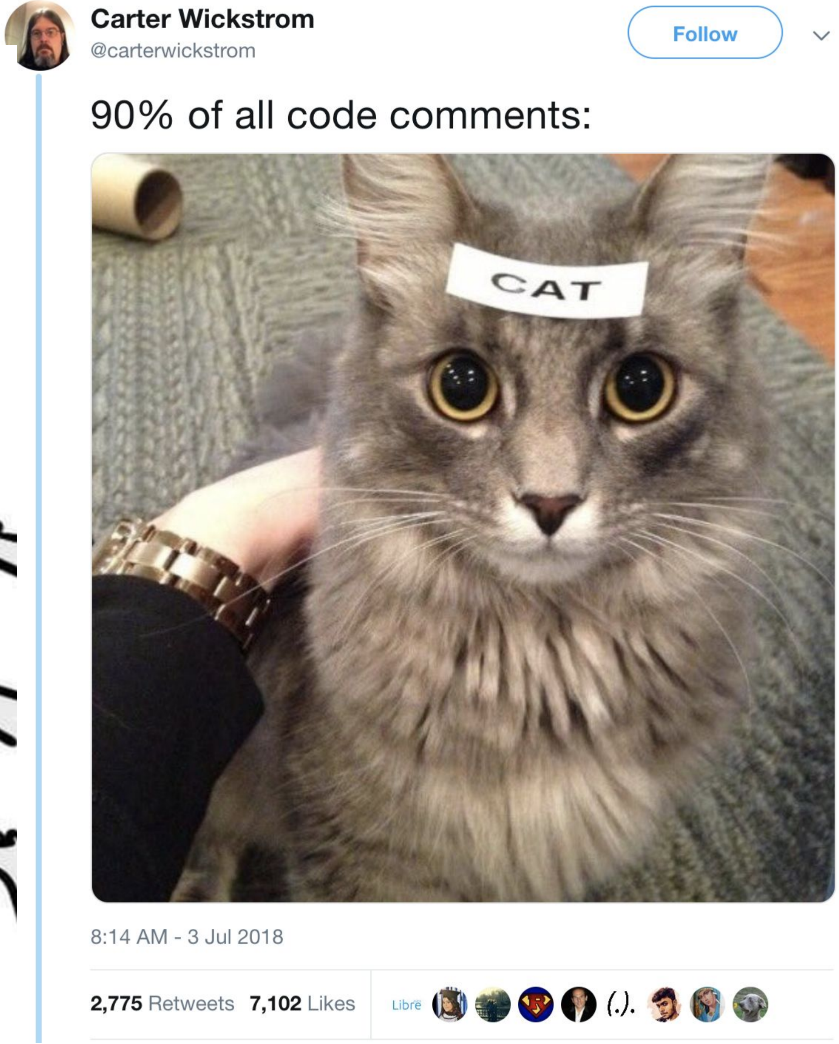
Maybe?

Code is the “how”, comments are the “why”

Don’t Waste Everyone’s Time (WET)

Line-by-line show lack of understanding

fb.com/programmingjokes/
NERD 4 LIFE.studio
what makes code bad?
No comment.


Carter Wickstrom
@carterwickstrom

90% of all code comments:

8:14 AM - 3 Jul 2018

2,775 Retweets 7,102 Likes

...Refactor...

...justCode better...

...don't use comments...

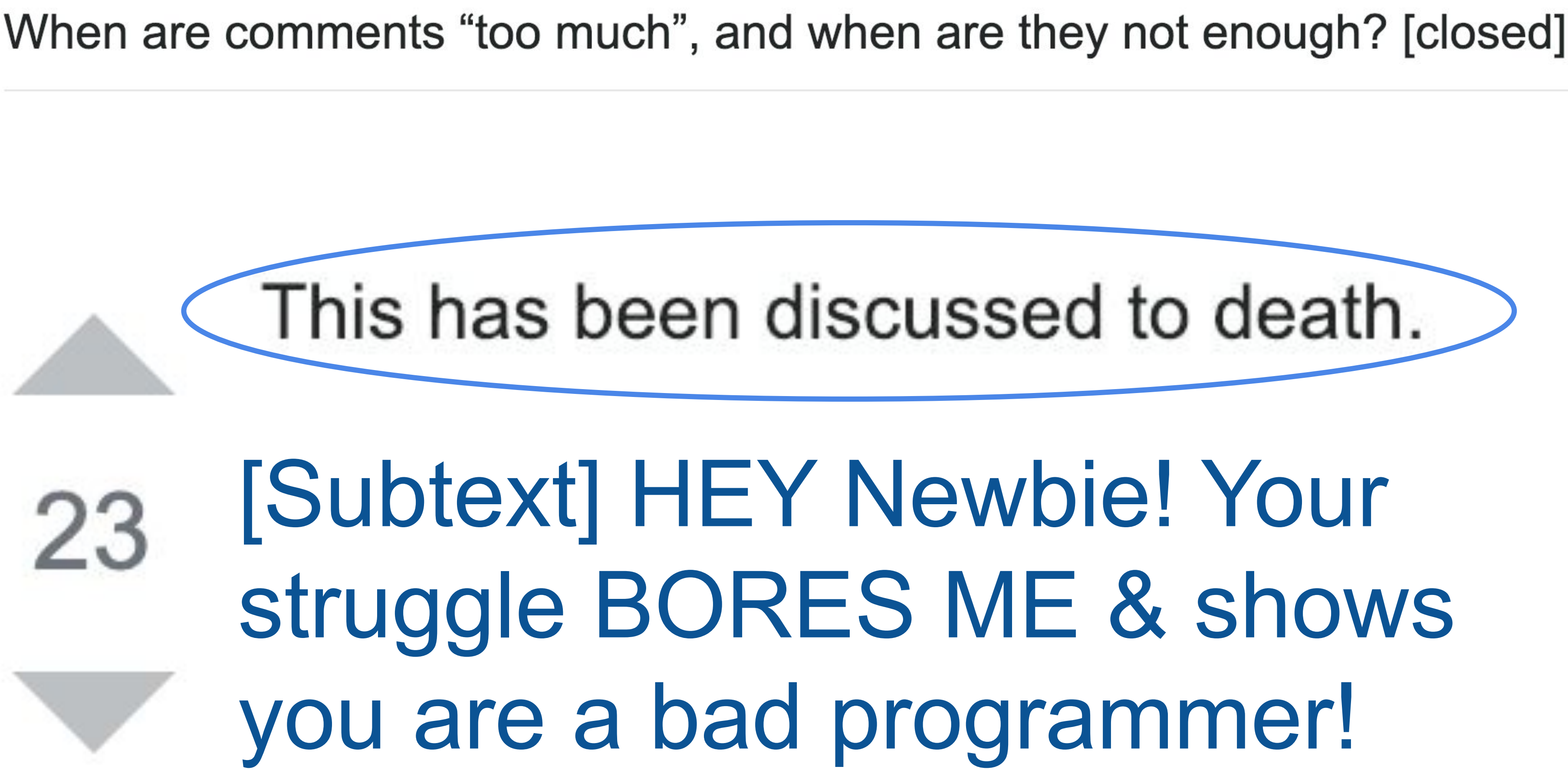
ONLY CODE MATTERS!

@veronica_hanus

“JUST CODE BETTER”

Thanks, people of the Internet!

© Mary McGowan



When are comments “too much”, and when are they not enough? [closed]

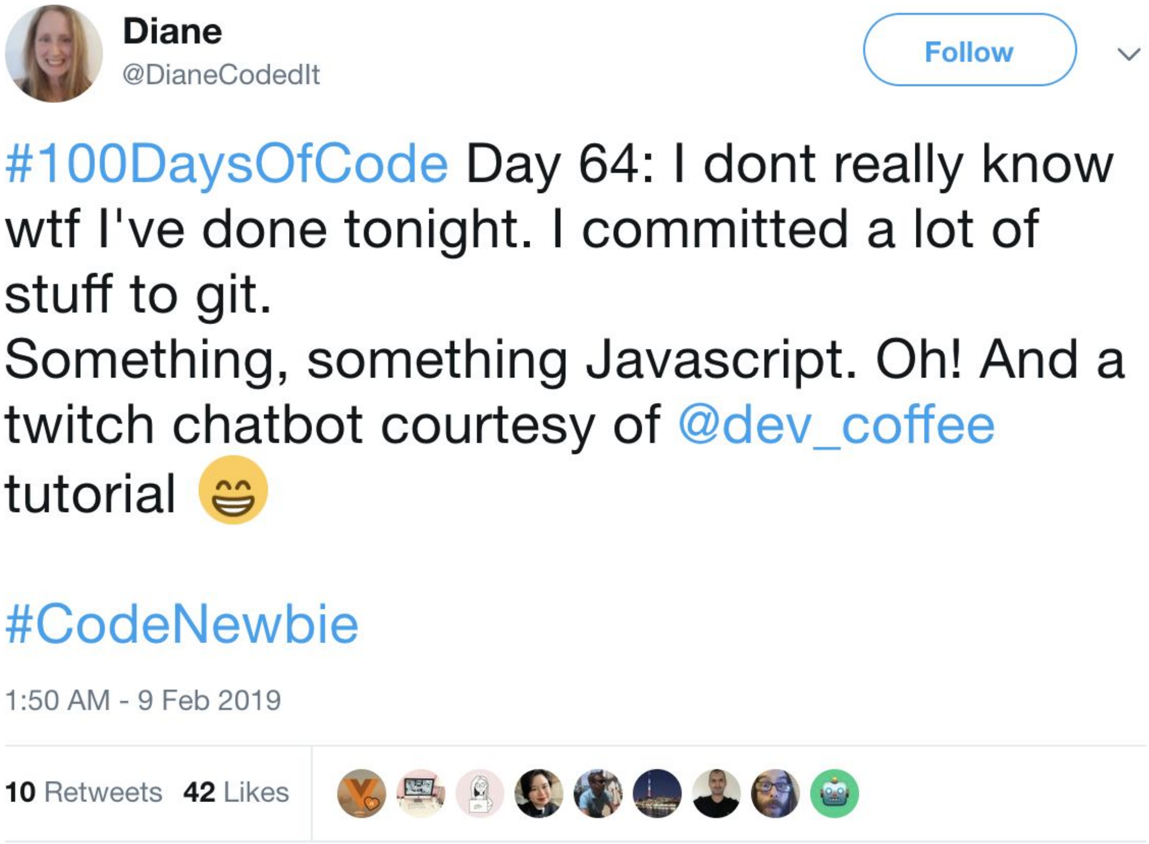
This has been discussed to death.

[Subtext] HEY Newbie! Your struggle BORES ME & shows you are a bad programmer! KTHXBYE

@veronica_hanus

Let’s face it… comments are *magic*

@veronica_hanus



Diane
@DianeCodedIt

#100DaysOfCode Day 64: I dont really know wtf I've done tonight. I committed a lot of stuff to git.
Something, something Javascript. Oh! And a twitch chatbot courtesy of @dev_coffee tutorial 😄

#CodeNewbie

1:50 AM - 9 Feb 2019

10 Retweets **42** Likes

Comments can:

- Label
- Questions
- Notes
- Outline
- Storage
- References
- Support overwhelmed learners

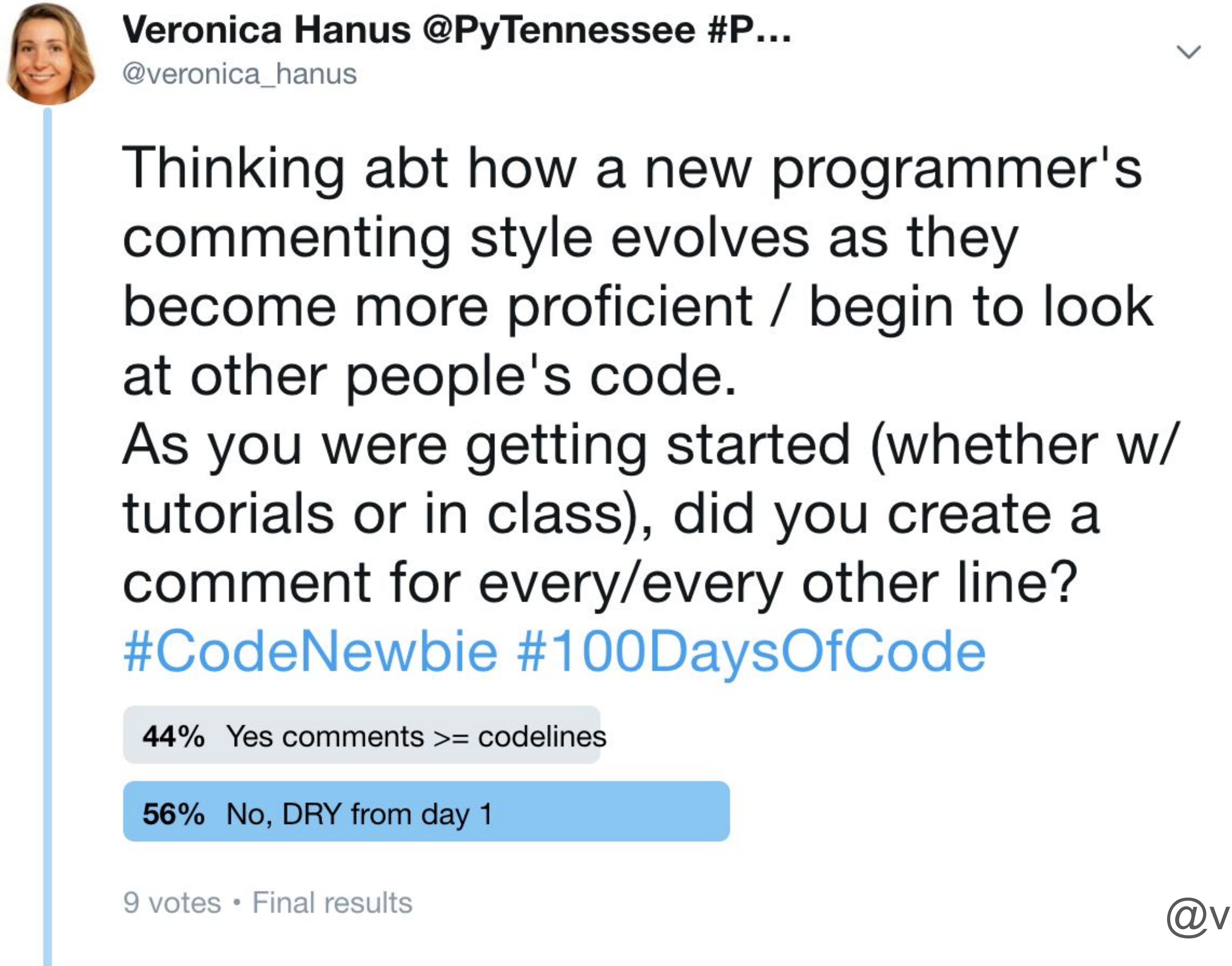
```
# V!: TODO
[........]
```

```
# Veronica
was heree!
```

Move fast & ~~break things~~
write bad comments

@veronica_hanus

@veronica_hanus



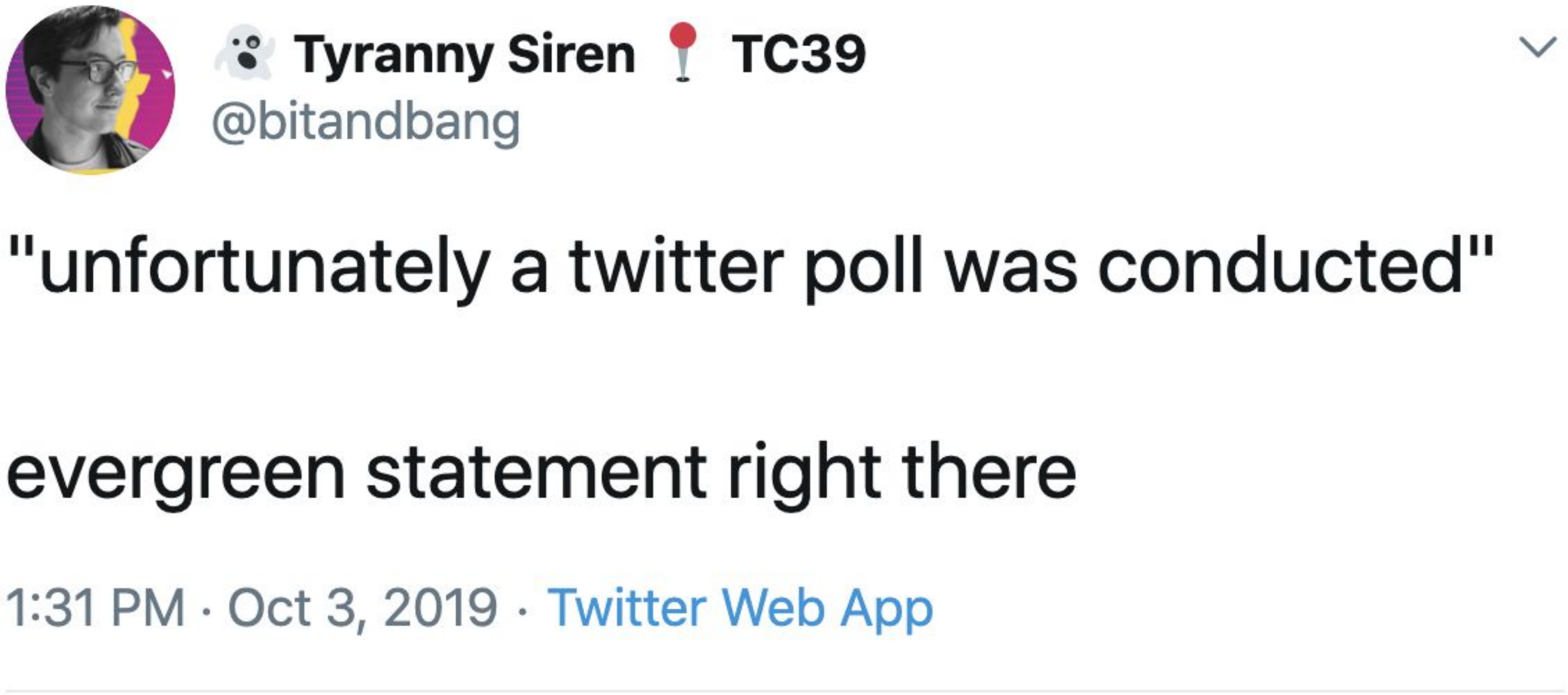
Veronica Hanus @PyTennessee #P...
@veronica_hanus

Thinking abt how a new programmer's commenting style evolves as they become more proficient / begin to look at other people's code.
As you were getting started (whether w/ tutorials or in class), did you create a comment for every/every other line? #CodeNewbie #100DaysOfCode

44% Yes comments >= codelines

56% No, DRY from day 1

9 votes • Final results



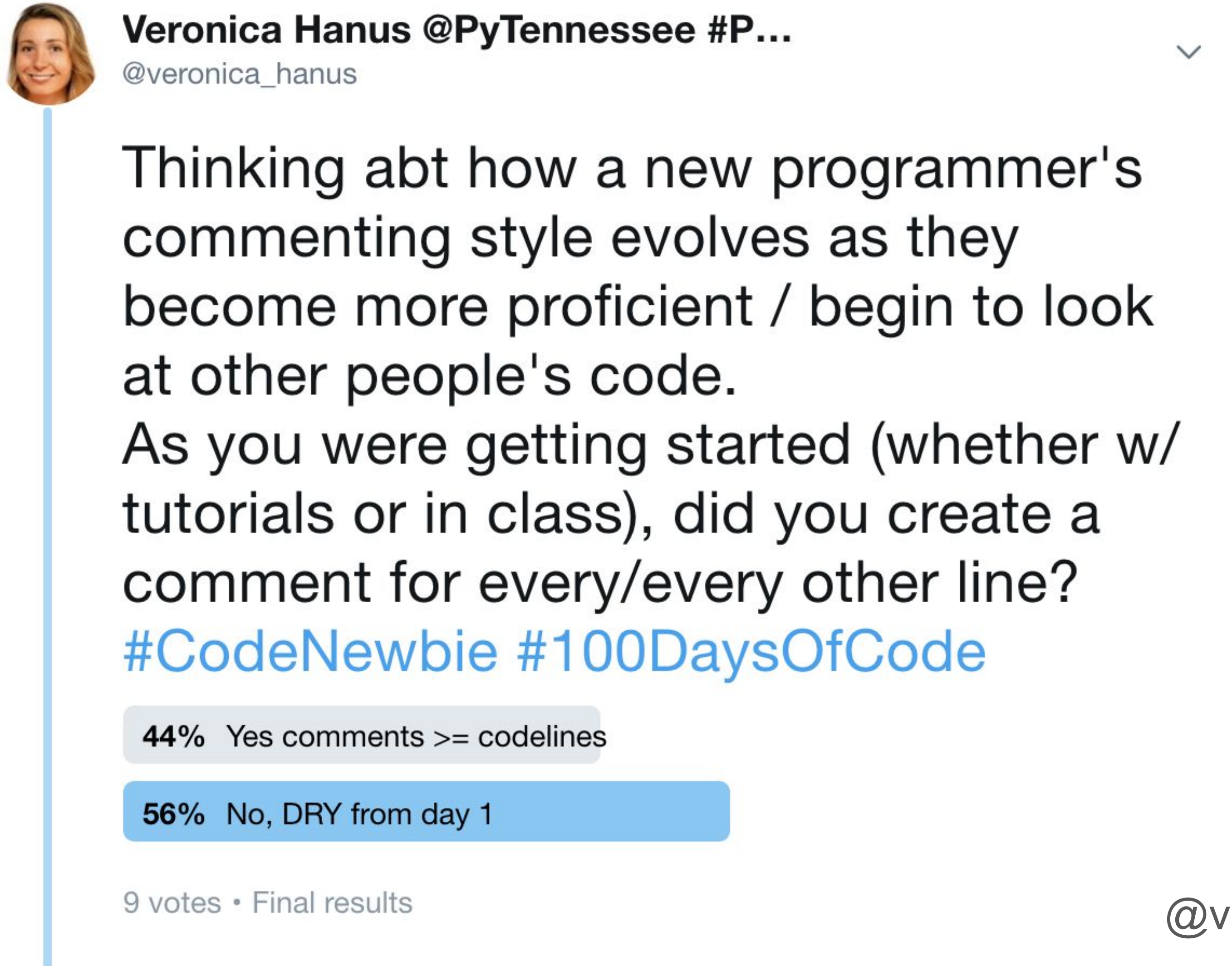
👻 Tyranny Siren 📍 TC39
@bitandbang

"unfortunately a twitter poll was conducted"

evergreen statement right there

1:31 PM · Oct 3, 2019 · Twitter Web App

@veronica_hanus



Veronica Hanus @PyTennessee #P...
@veronica_hanus

Thinking abt how a new programmer's commenting style evolves as they become more proficient / begin to look at other people's code.
As you were getting started (whether w/ tutorials or in class), did you create a comment for every/every other line? #CodeNewbie #100DaysOfCode

44% Yes comments >= codelines

56% No, DRY from day 1

9 votes • Final results

@veronica_hanus

Radial:

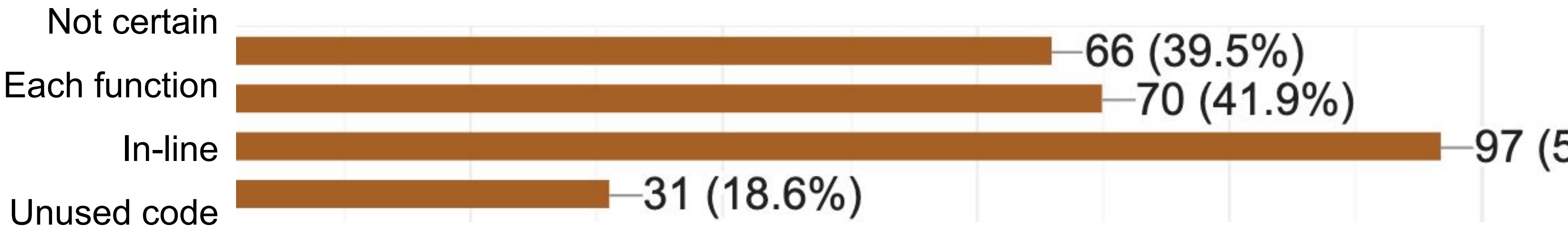
- Current/recent use: Comment uncertainty, Function-level comments, Clarification, Unused code, Other
- When comments added: Scoping & planning, As functions written, Pairing, As I learn people don’t understand, Clean-up
- How long programming?
- How long professionally?
- Path to programming?

Agree/Disagree (1-10):

- Comments:
    - Help me remember what my code does
    - Clarify my thinking
    - Help me learn
    - Save time
    - Delete before projects is shared
- Yes to function-level, no in-line
- Clear code is self-documenting
- Uncomfortable writing

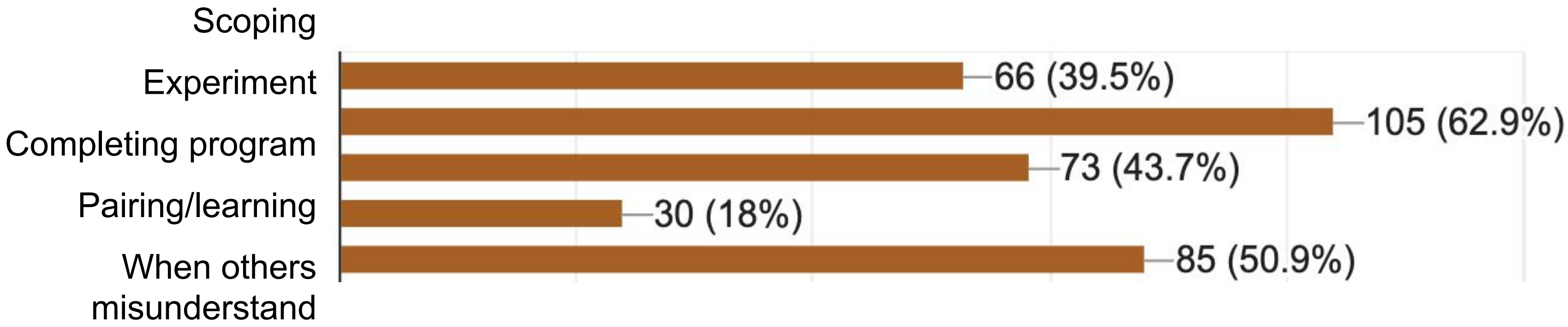
What best describes your current/recent use of comments?

167 responses



When do you add comments?

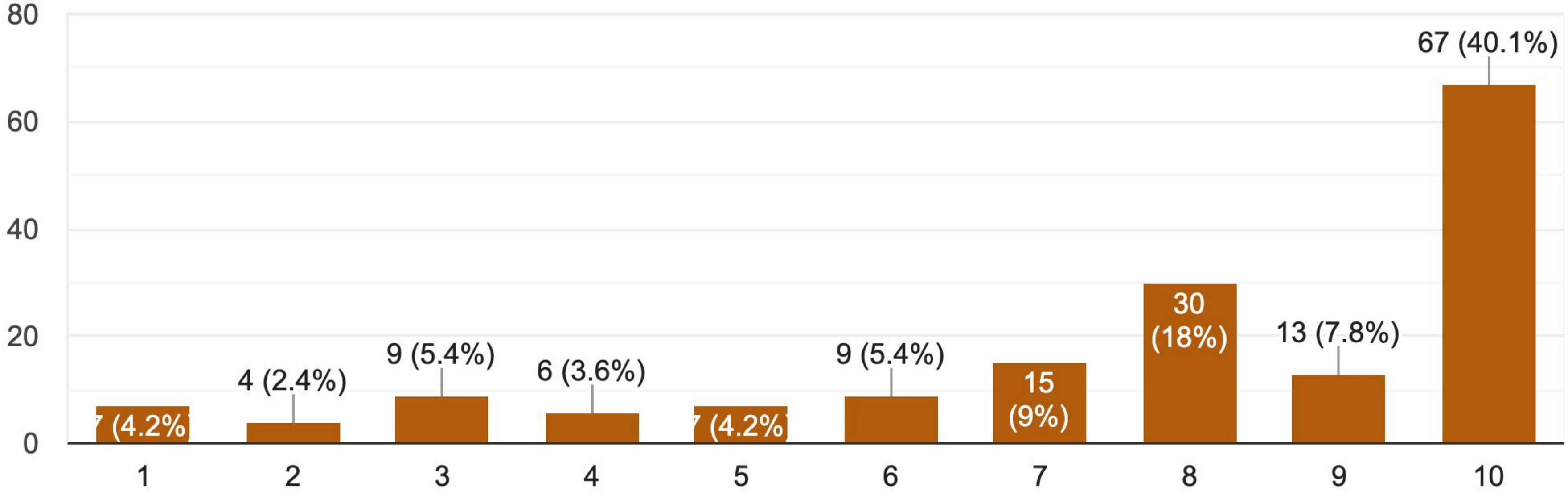
167 responses



@veronica_hanus

Comments can help me remember what my code does

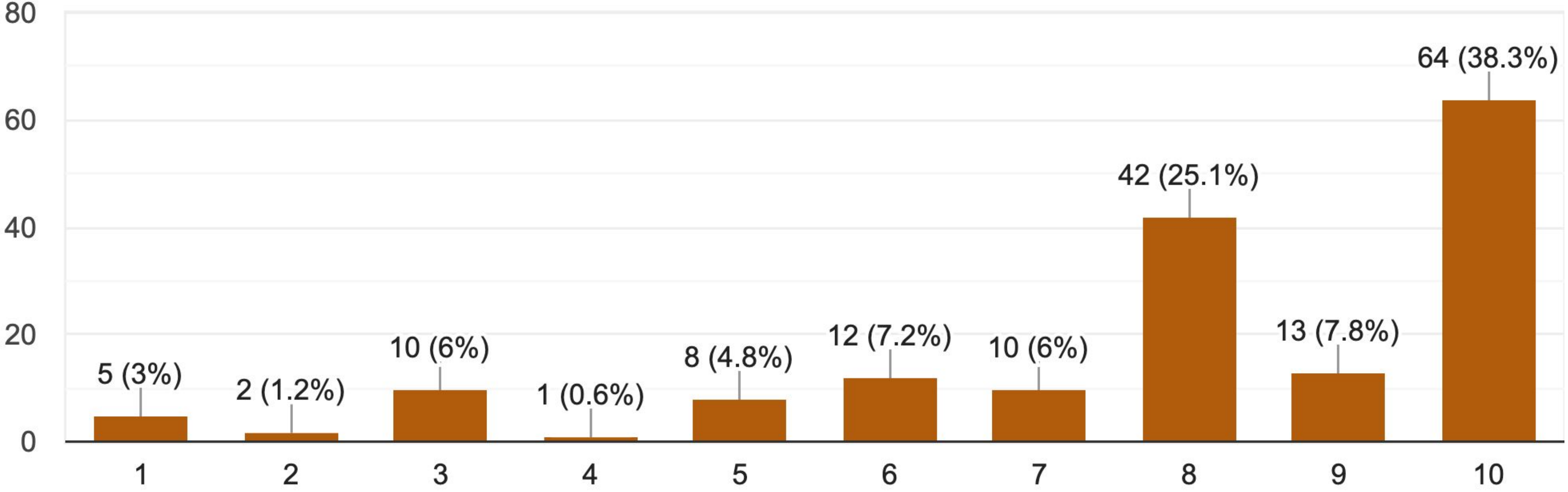
167 responses



@veronica_hanus

Comments help me clarify my thinking

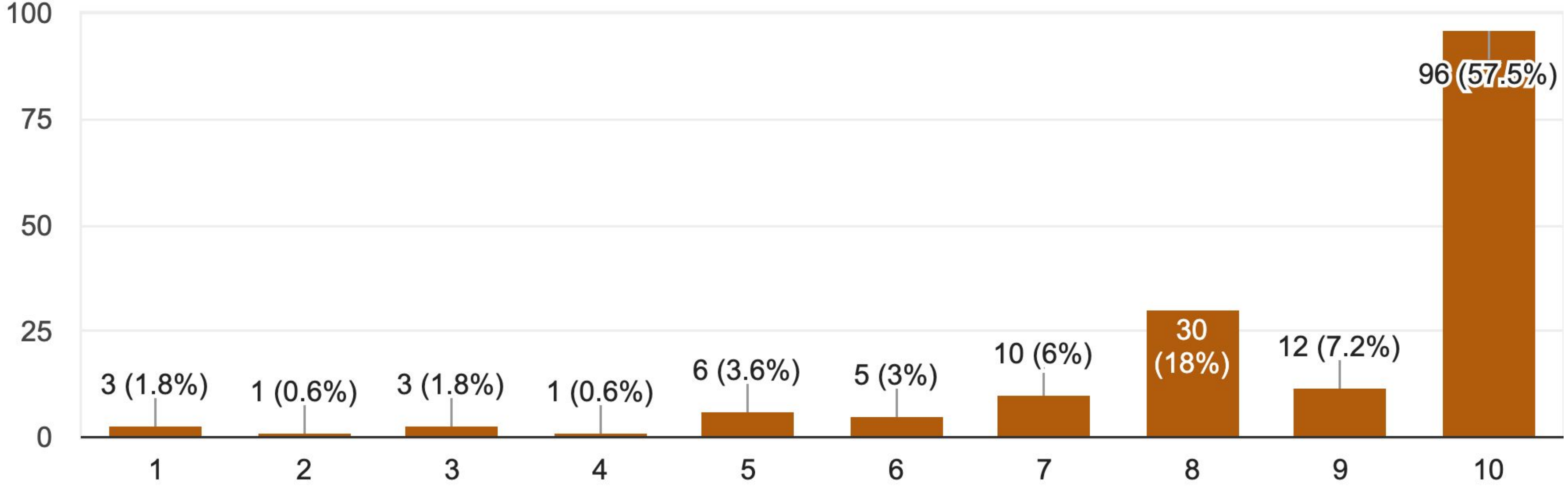
167 responses



@veronica_hanus

Comments can save current and future project developers time

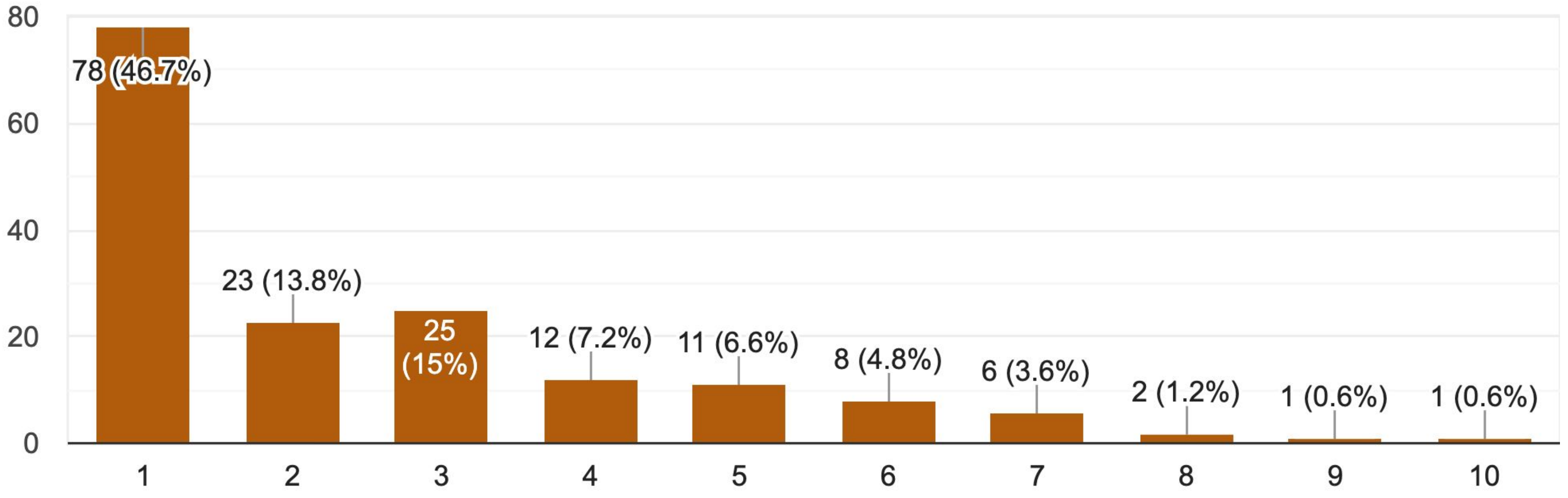
167 responses



@veronica_hanus

Comments are helpful to an individual contributor building a project, but should be deleted before the project is shared

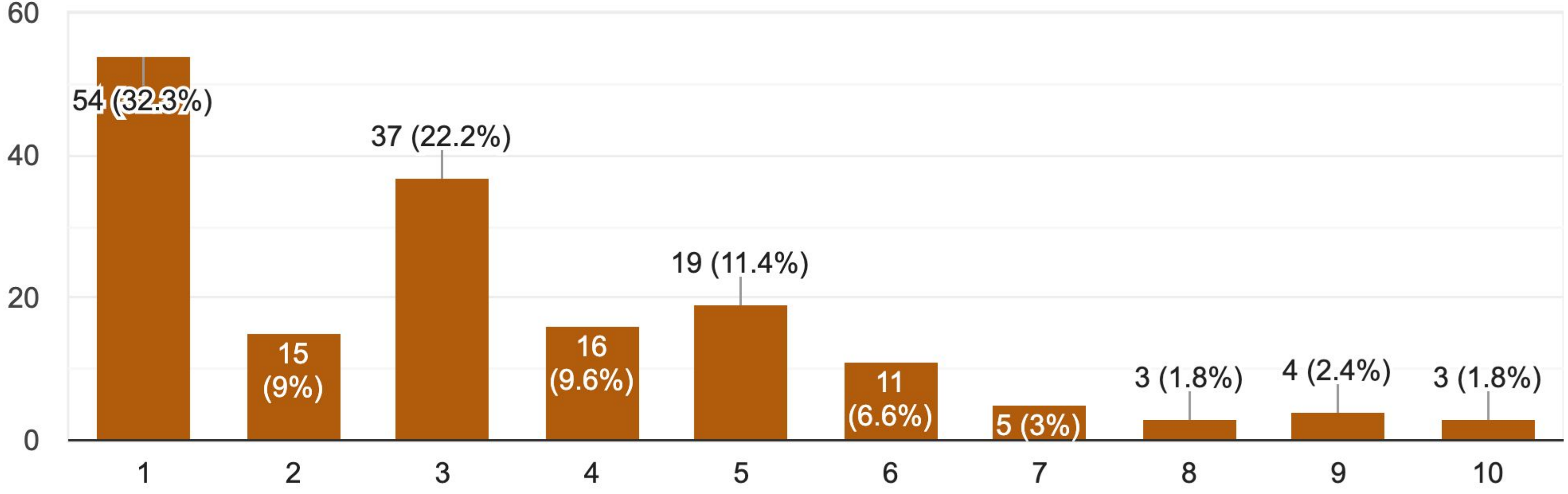
167 responses



@veronica_hanus

Function-level comments are acceptable, but in-line comments are clutter

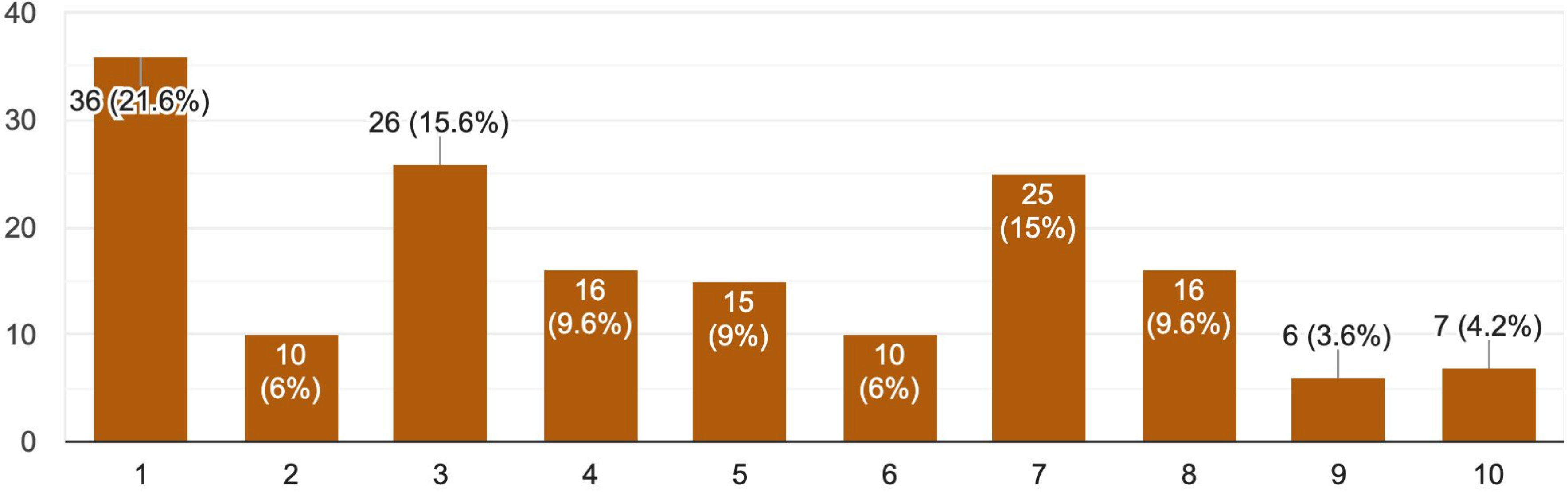
167 responses



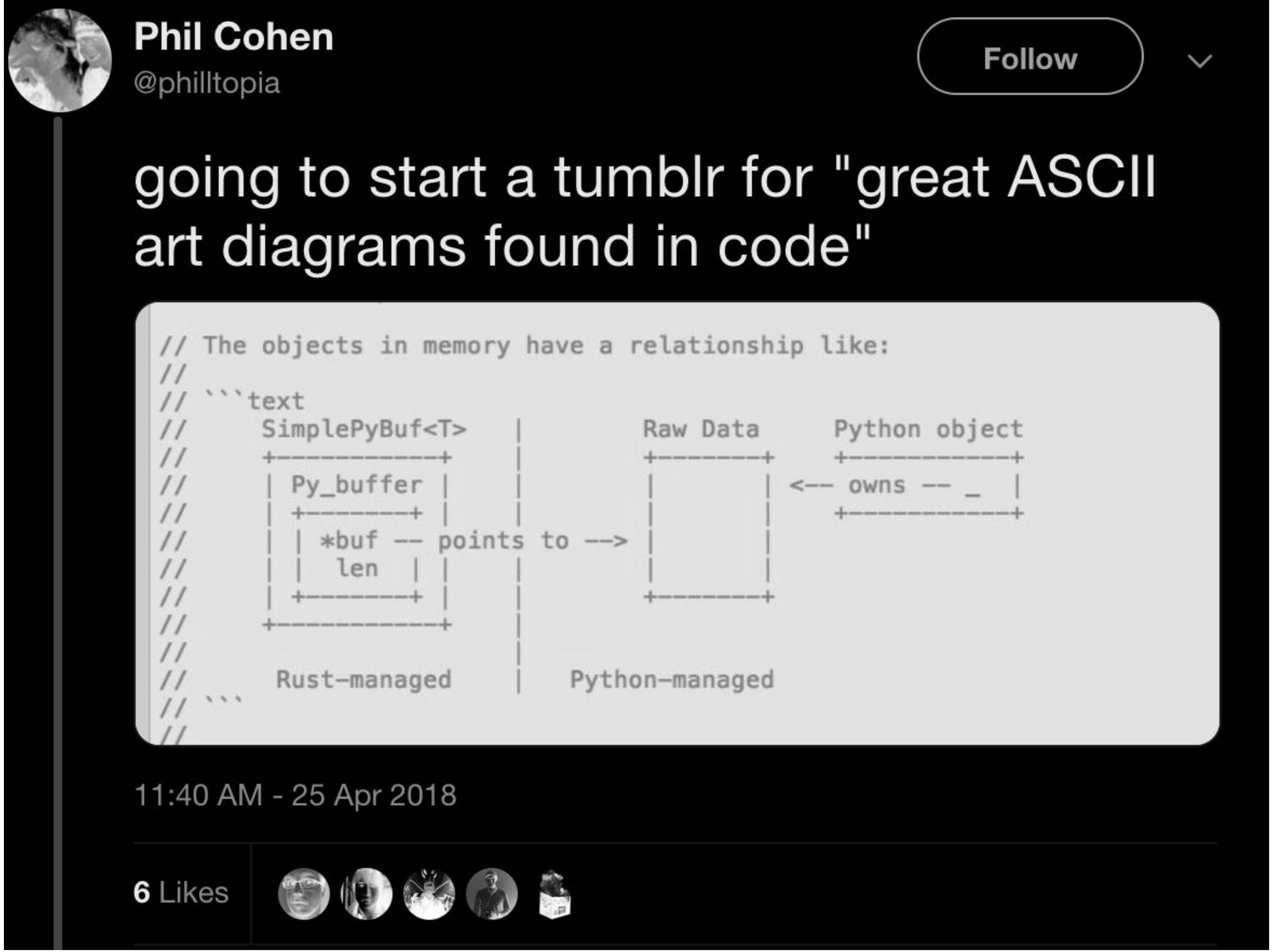
@veronica_hanus

Clear code is self-documenting &amp; doesn’t need comments

167 responses



@veronica_hanus



Phil Cohen
@philltopia

going to start a tumblr for "great ASCII art diagrams found in code"

````
// The objects in memory have a relationship like:
//
// ```text
//     SimplePyBuf<T>   |        Raw Data     Python object
//     +-----------+    |        +-------+    +-----------+
//     | Py_buffer |    |        |       | <-- owns -- _  |
//     | +-------+ |    |        |       |    +-----------+
//     | | *buf -- points to --> |       |
//     | |  len  | |    |        |       |
//     | +-------+ |    |        +-------+
//     +-----------+    |
//                      |
//      Rust-managed    |   Python-managed
// ```
//
````

11:40 AM - 25 Apr 2018

6 Likes

Make your own! https://www.ascii-art-generator.org/

@veronica_hanus

“A global patchwork of Github & Gitlab repositories don’t just contain software--they contain **our shared understanding** & collaboration around common interests & problem solving.”

Jono Bacon in his forward to “The Business Value of Developer Relations”

Comments teach us about ourselves

@veronica_hanus

What can we do?

Goal: **Support learners where they are at**, praising their accomplishments, while pointing them **gently toward the future**

- ➔ **Empathy** can be hard!
- ➔ Remember **our overwhelmed learner**
- ➔ Advise **current them**, not future them!
- ➔ Write w/ comments? **Share** them!
- ➔ Suggest **a deep dive** & **reading others' code**?

What can we say? What do they need?

- ➔ Someone is learning their attitudes toward documentation ***from you***

Rethink comments

- ➔ Comments == docs?
- ➔ Comments teach us about ourselves

@veronica_hanus

I tweet at @veronica_hanus

Non-tweeters 👋me@veronicahanus.com

Survey: http://bit.ly/comment-use

Video & Slides 🔜
http://veronicahanus.com/talks

🙌**Write the comments you wish you had!**

🙏 **Each of you** for coming, the **PyData organizing team** for the opportunity, & **the ~170 internet-folk** who have shared their "comments on comments" **<3**

👇 *Your company recruiting a DevRel or Dev Advocate?*👇
🙌**@veronica_hanus**🙌

Learning resources

- **Docs that drive code:** https://blog.izs.me/2017/06/documentation-driven-development
- **Readmes (start here!):** http://tom.preston-werner.com/2010/08/23/readme-driven-development.html
- **My post on Readme mechanics :** http://veronicahanus.com/blog/2017/03/06/writing-readmes.html
- **Docs are part of code:** https://www.writethedocs.org/guide/docs-as-code/
- **Motivation for docs:** https://stoplight.io/blog/writing-documentation-when-you-arent-a-technical-writer-part-two-59997587cc2a/
- **Guideline for docs:** https://opensource.com/business/15/5/write-better-docs
- **Read code? Read docs too! (& resources for doc generation):** https://github.com/PharkMillups/beautiful-docs
- **Goal of documentation:** https://kadavy.net/blog/posts/productivity-cycles-podcast/
- **Talks from “Compassionate Coding”:** https://compassionatecoding.com/media
- **Notes from a deep dive into “Clean Code”:** https://medium.com/mindorks/how-to-write-clean-code-lessons-learnt-from-the-clean-code-robert-c-martin-9ffc7aef870c
- **Guidelines for comments:** https://www.cs.utah.edu/~germain/PPS/Topics/commenting.html

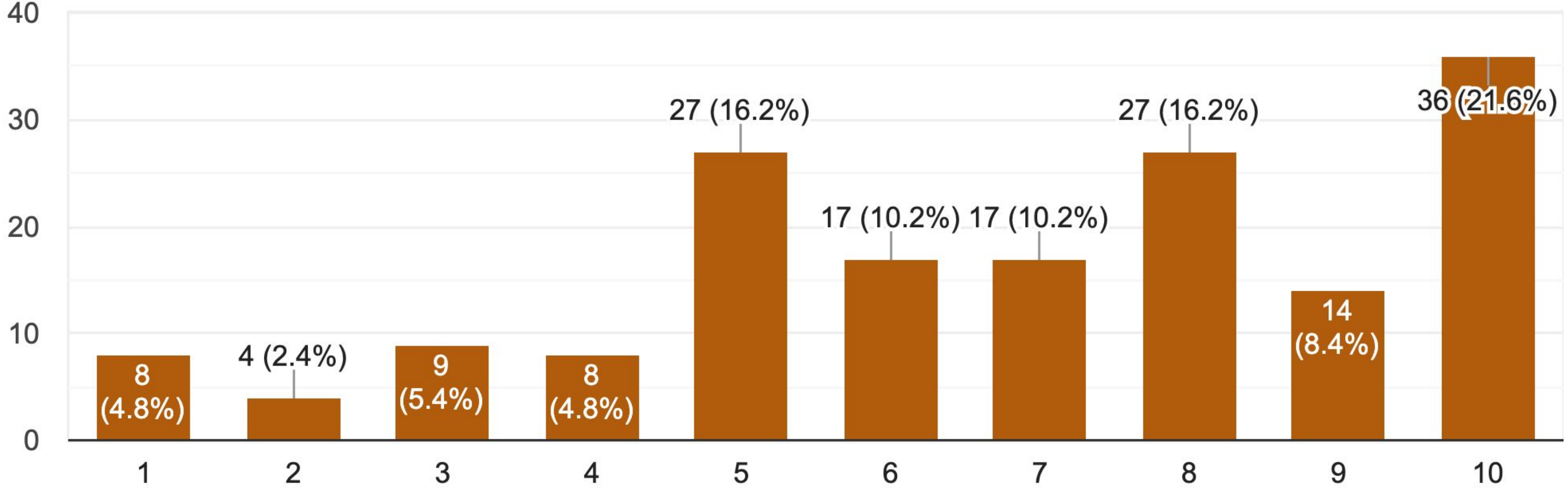
@veronica_hanus

Credits

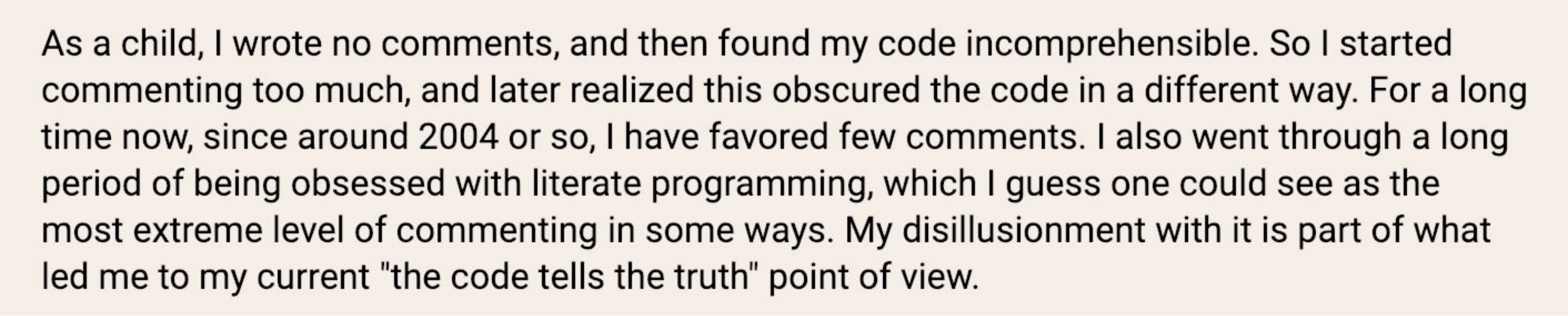
- **ASCII comments:** https://twitter.com/johnregehr/status/1095018518737637376
- **“Suffering on StackOverflow”:** https://medium.com/@Aprilw/suffering-on-stack-overflow-c46414a34a52
- **We all go through this:** https://twitter.com/anupbattasha/status/1094959013194649600
- **Comment your cats:** https://twitter.com/carterwickstrom/status/1014165500056596481
- **Commenting for learning:** https://twitter.com/jessfraz/status/1093713454781784065
- **Man pages for documentation:** https://twitter.com/aemeredith/status/1033445823181287424
- **Goal of documentation:** https://twitter.com/kadavy/status/1093820499271000064
- **Programmer in bath:** https://footage.framepool.com/en/shot/798293950-bath-tub-bathroom-information-technology-student-university
- **A good programmer:** https://code.likeagirl.io/herstory-software-engineer-maker-estefannie-d4fdec7b069a
- **Squirrels wants you to stop:** https://patch.com/florida/brandon/stop-right-there-brandon-squirrel-wins-world-photo-contest
- **Sad programmer:** https://drawception.com/game/H73GxcwzmW/a-sad-programmer-playing-drawception/#panel-1537906
- **Inconceivable:** https://images.app.goo.gl/2nnv4DN7E8yFkeKj8

Comments help me learn

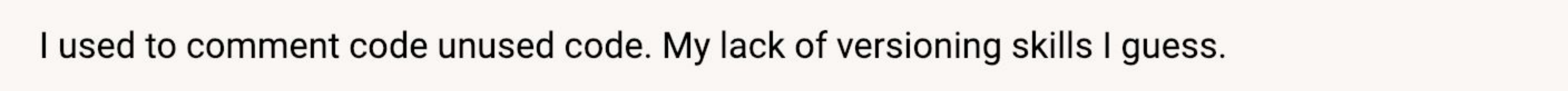
167 responses



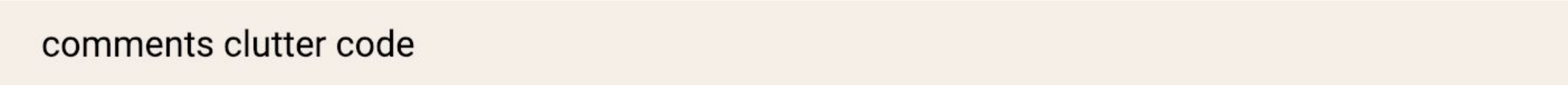
@veronica_hanus



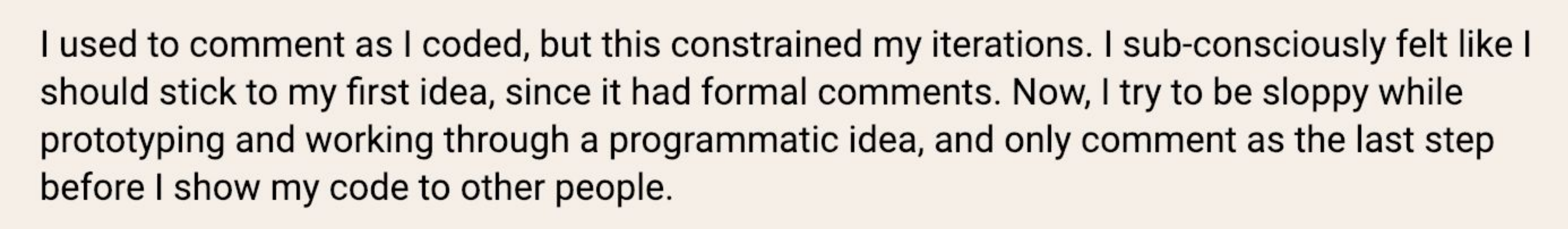
As a child, I wrote no comments, and then found my code incomprehensible. So I started commenting too much, and later realized this obscured the code in a different way. For a long time now, since around 2004 or so, I have favored few comments. I also went through a long period of being obsessed with literate programming, which I guess one could see as the most extreme level of commenting in some ways. My disillusionment with it is part of what led me to my current "the code tells the truth" point of view.



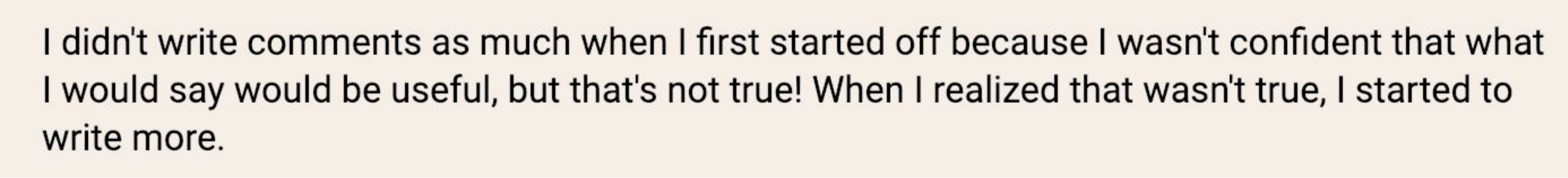
I used to comment code unused code. My lack of versioning skills I guess.



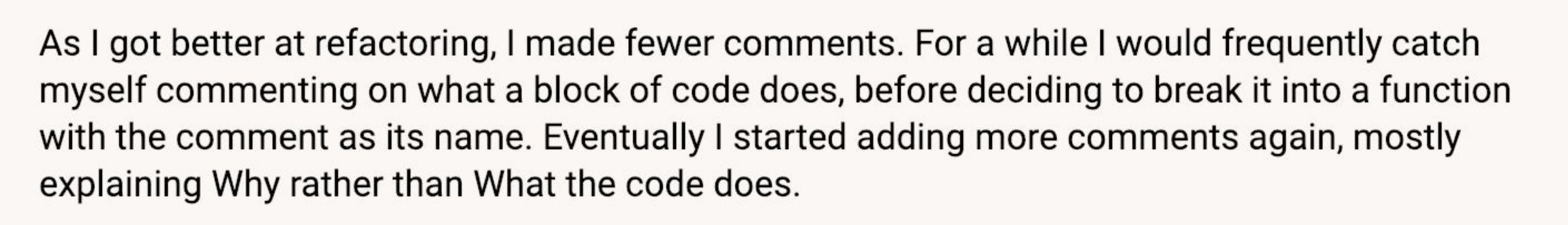
comments clutter code



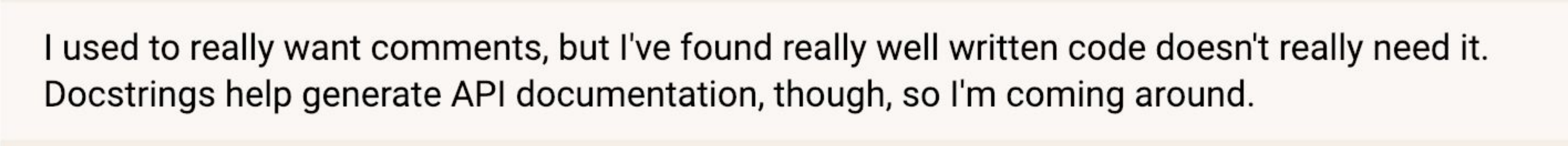
I used to comment as I coded, but this constrained my iterations. I sub-consciously felt like I should stick to my first idea, since it had formal comments. Now, I try to be sloppy while prototyping and working through a programmatic idea, and only comment as the last step before I show my code to other people.



I didn't write comments as much when I first started off because I wasn't confident that what I would say would be useful, but that's not true! When I realized that wasn't true, I started to write more.



As I got better at refactoring, I made fewer comments. For a while I would frequently catch myself commenting on what a block of code does, before deciding to break it into a function with the comment as its name. Eventually I started adding more comments again, mostly explaining Why rather than What the code does.



I used to really want comments, but I've found really well written code doesn't really need it. Docstrings help generate API documentation, though, so I'm coming around.