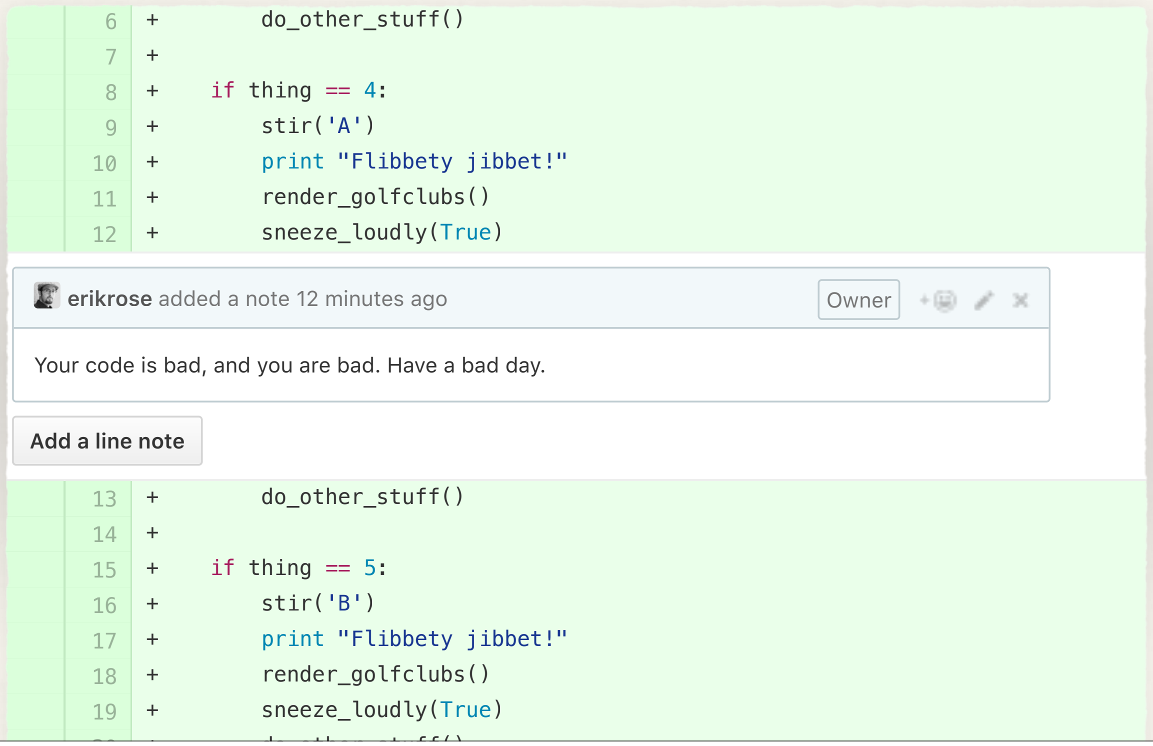
The Code Review Review

erik@mozilla.com · IRC: ErikRose · @ErikRose



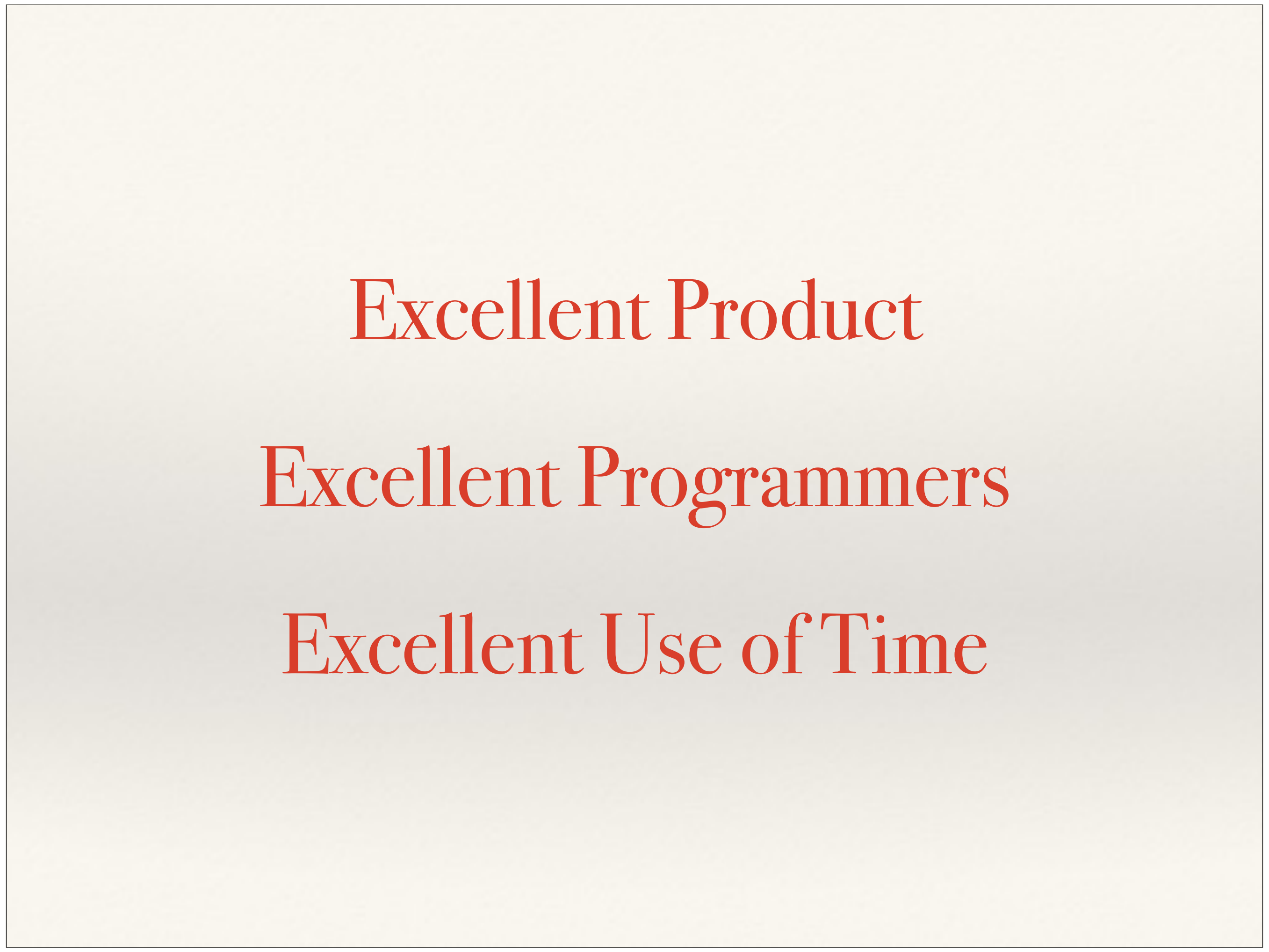
```
+       do_other_stuff()
+
+   if thing == 4:
+       stir('A')
+       print "Flibbety jibbet!"
+       render_golfclubs()
+       sneeze_loudly(True)
```

erikrose added a note 12 minutes ago

Your code is bad, and you are bad. Have a bad day.

```
+       do_other_stuff()
+
+   if thing == 5:
+       stir('B')
+       print "Flibbety jibbet!"
+       render_golfclubs()
+       sneeze_loudly(True)
```

- ☐ I'm Erik Rose
  - ☐ I have been through hundreds of code reviews.
  - ☐ Been with Mozilla for **6 years**
  - ☐ Maintain about a **dozen** open-source projects.
  - ☐ So today we're going to take a look at how to do **constructive** code reviews that avoid this sort of thing.



Excellent Product

Excellent Programmers

Excellent Use of Time

Our goal:
__excellent__ product
- ensure **works** (beyond tests & tools)
- ensure **understandable** (2nd brain)

develop **__excellent__** programmers,
- knowledge of **codebase** & craft of **programming**

make **__excellent__** use of time
- neither nitpicking nor glossing over important errors

- Who's done code review before?
- All code at Mozilla gets reviewed before it gets merged.
  - Put it up in some system.
  - comment.
  - tweaks.

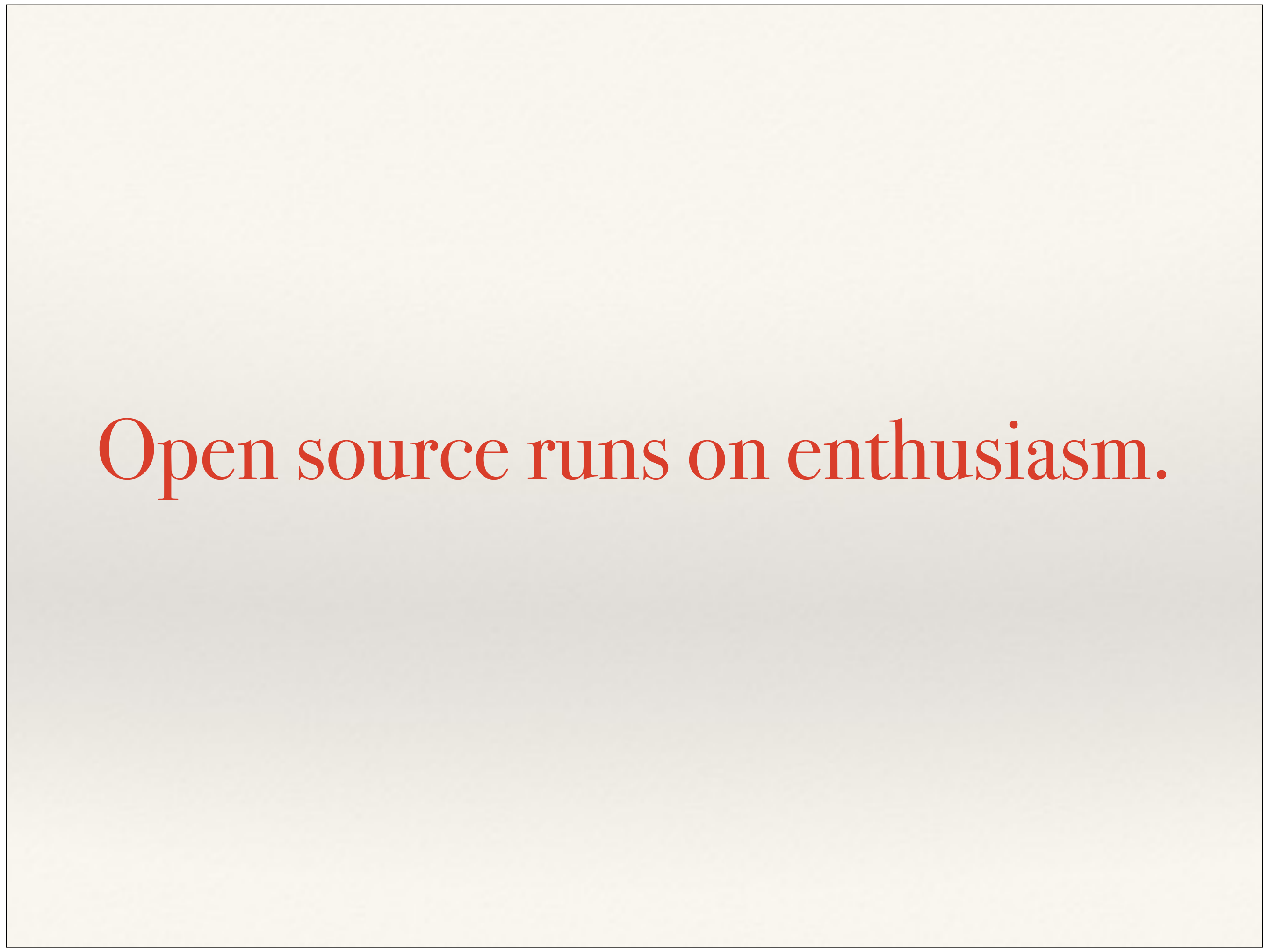
^**Good** things.

- **__Lurking horrors…__**

- Improve the code
  - I like to think I'm good at this, but people still find improvements to make in my stuff almost every time.
  - Also ensures it's understandable, not just executable
- Spread knowledge about the codebase
  - As a reviewer, you can inform other people what's going on in the codebase so you're not stuck with the sole responsibility of maintaining it.
- Spread proficiency with the craft
  - It's as much about teaching as fixing. It's a major way we bring junior people up the ladder.

- We can be short with each other, miscommunicate, and under- or overreview.
- This is especially dangerous, since…

Open source runs on enthusiasm…

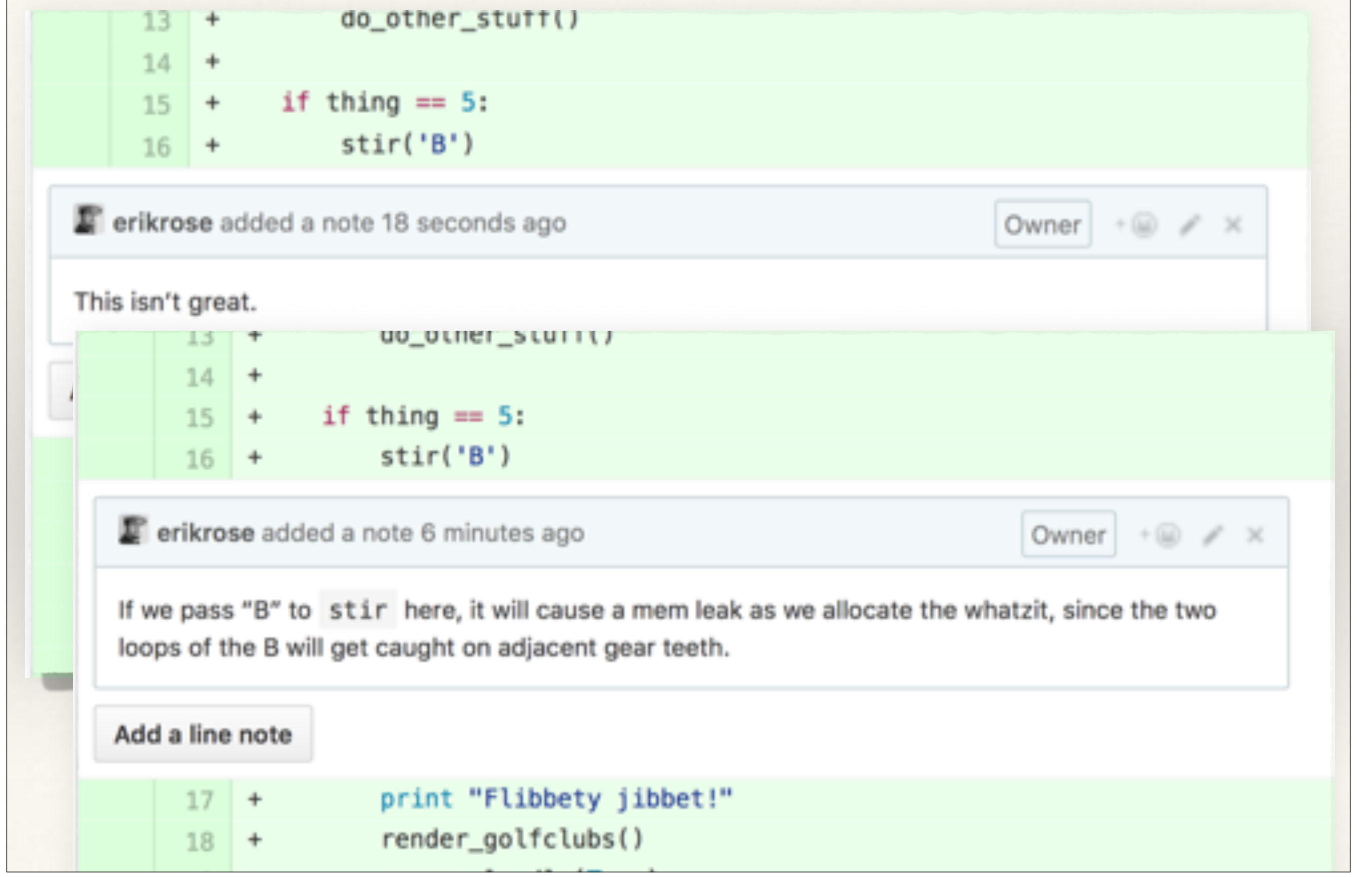


Open source runs on enthusiasm.

- Even when you're getting paid to do it
  - There's a lot of intrinsic motivation around here.
- If we beat people up in code reviews
  - we can drive the desire to contribute out of people
  - even drive out the people themselves
- But if we take care, we can instead magnify our power through teaching and encouragement.
  - For example…

Bad Review,
Good Review

Clarity of Explanation



```
+        do_other_stuff()
+
+    if thing == 5:
+        stir('B')
```

erikrose added a note 18 seconds ago Owner

This isn't great.

```
+        do_other_stuff()
+
+    if thing == 5:
+        stir('B')
```

erikrose added a note 6 minutes ago Owner

If we pass "B" to `stir` here, it will cause a mem leak as we allocate the whatzit, since the two loops of the B will get caught on adjacent gear teeth.

Add a line note

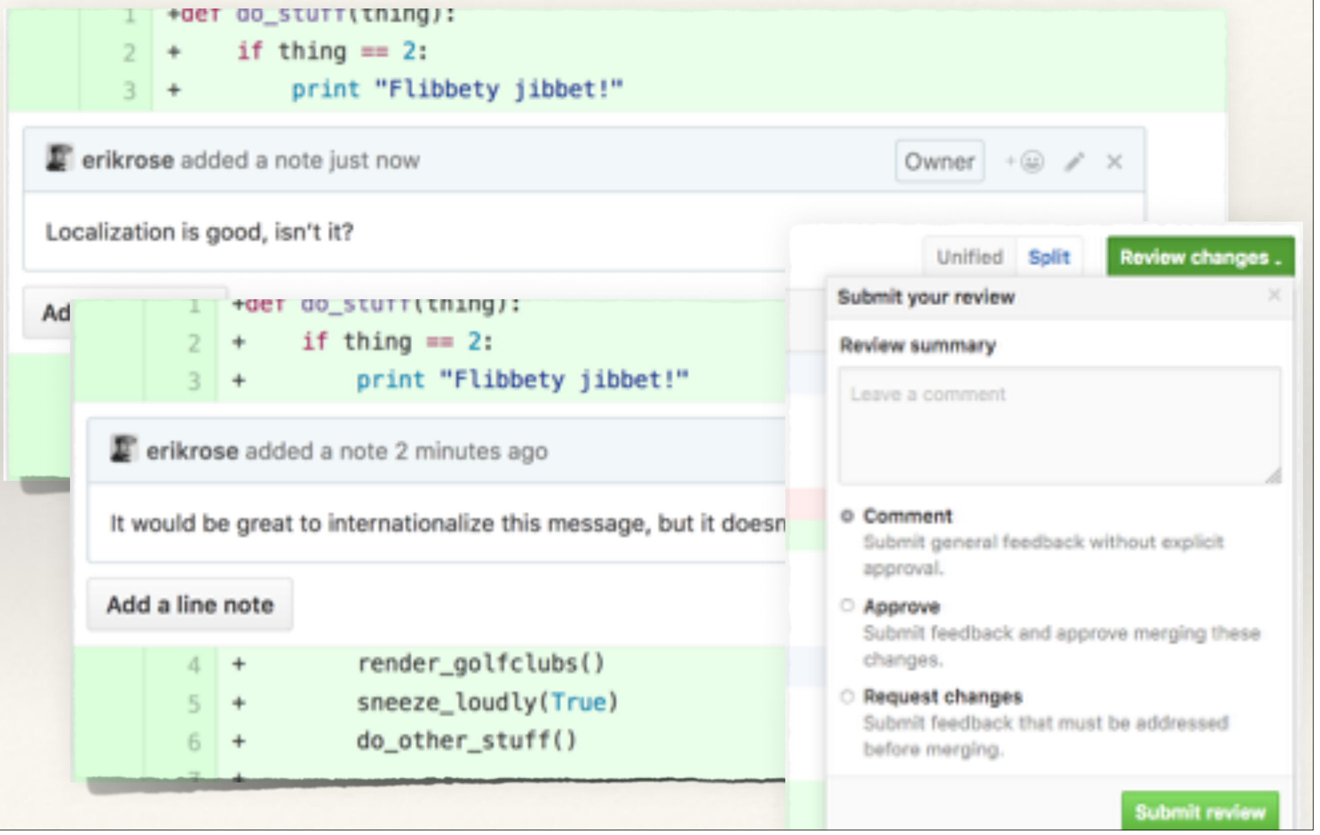
```
+        print "Flibbety jibbet!"
+        render_golfclubs()
```

- Clarity

Most important. After all, our first duty is to the truth, since nature cannot be fooled. "This is wrong." or "This makes frob() buggy." **vs. "…"**

- Example code snippets, REPL sessions, links to relevant documentation, etc.
  - Help bridge gaps when English isn't a first language
  - When people don't know each other well
  - When someone's new to the project
  - When someone hasn't had their morning prosthetic neurotransmitters
- If you're talking past each other, try Vidyo.
  - Be sure to write down the result.
  - Be considerate of the other person's time
    - Timezones
    - Offer to make an appointment; don't expect them to drop everything right now.

Clarity of Expectation



```
+def do_stuff(thing):
+    if thing == 2:
+        print "Flibbety jibbet!"
```

erikrose added a note just now

Localization is good, isn't it?

```
+def do_stuff(thing):
+    if thing == 2:
+        print "Flibbety jibbet!"
```

erikrose added a note 2 minutes ago

It would be great to internationalize this message, but it doesn

```
+        render_golfclubs()
+        sneeze_loudly(True)
+        do_other_stuff()
```

Submit your review

Review summary

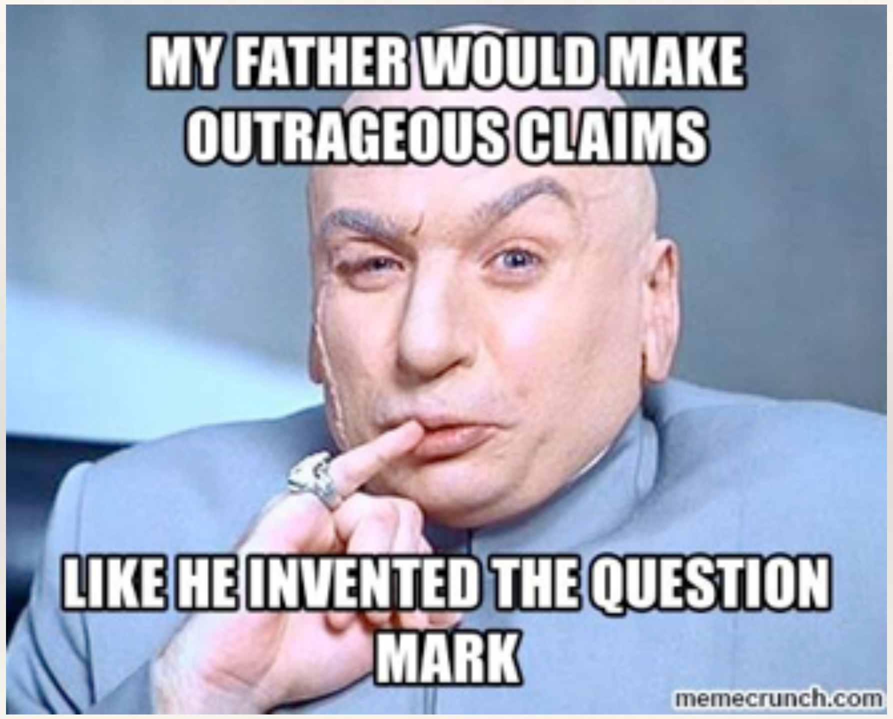
Leave a comment

- Comment — Submit general feedback without explicit approval.
- Approve — Submit feedback and approve merging these changes.
- Request changes — Submit feedback that must be addressed before merging.

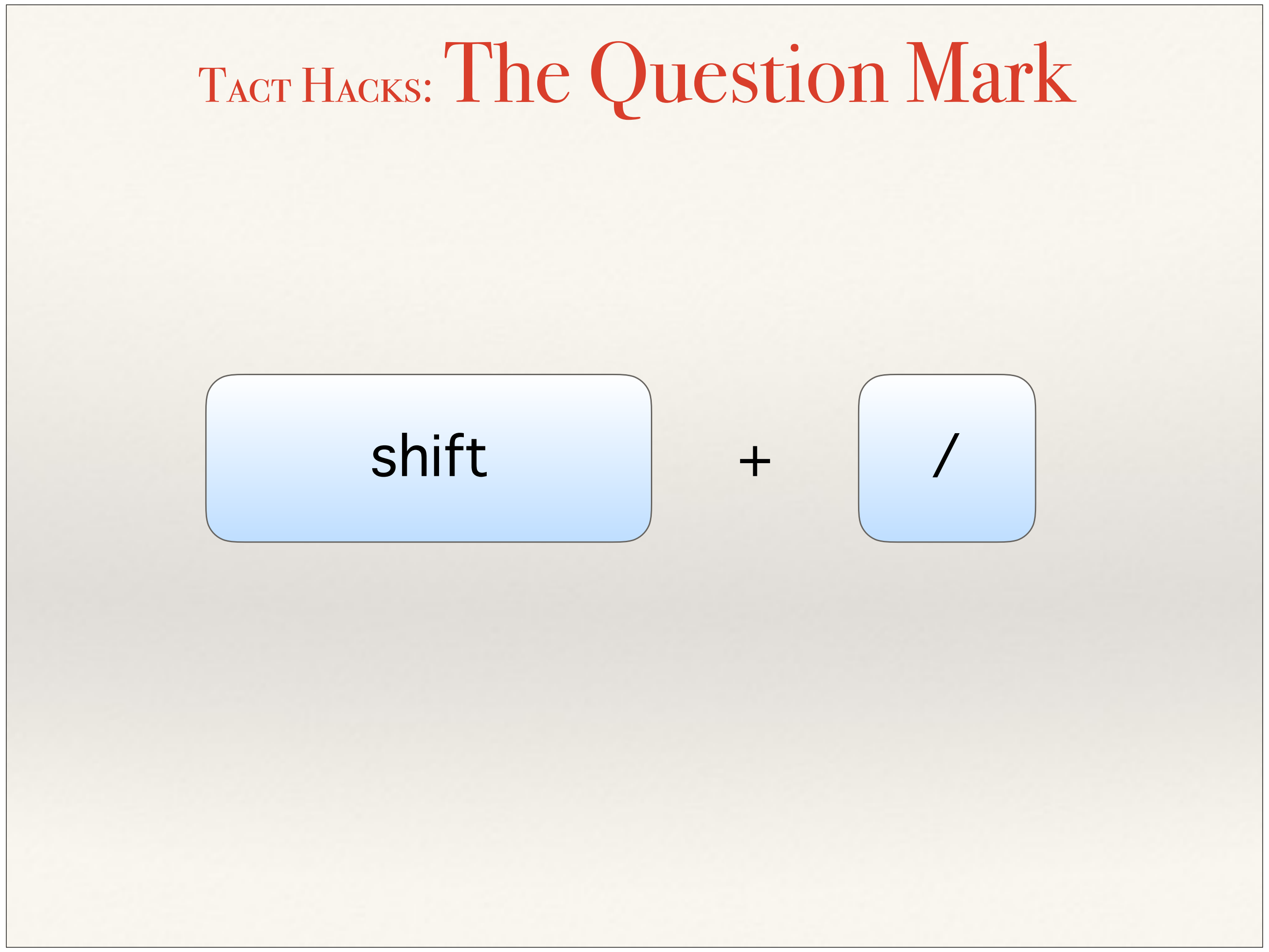
- Clarity of expectations
  - Clarity of explanation has a little brother, clarity of expectation.
    - Be clear about whether you're asking the person to do something and how much you care.
    - “Localization is good, isn't it?” vs. “**It would be great to internationalize** this message, but it doesn't need to block the merge.”
    - Unless it's trivial, lean toward doing it later so the PR doesn't go on forever, with nobody enjoying its benefits.
    - A big "That's all I see. With those changes, I think it's ready to merge." lets them (and you!) know you're done and ends with a note of encouragement.

Or **GH reviews.**

Tact Hacks


MY FATHER WOULD MAKE OUTRAGEOUS CLAIMS
LIKE HE INVENTED THE QUESTION MARK
memecrunch.com

- When you have to point out something bad, you words can be construed as a personal attack, even if you didn't mean them that way.
- A spoonful of sugar
- The easist of these is the humble question mark.

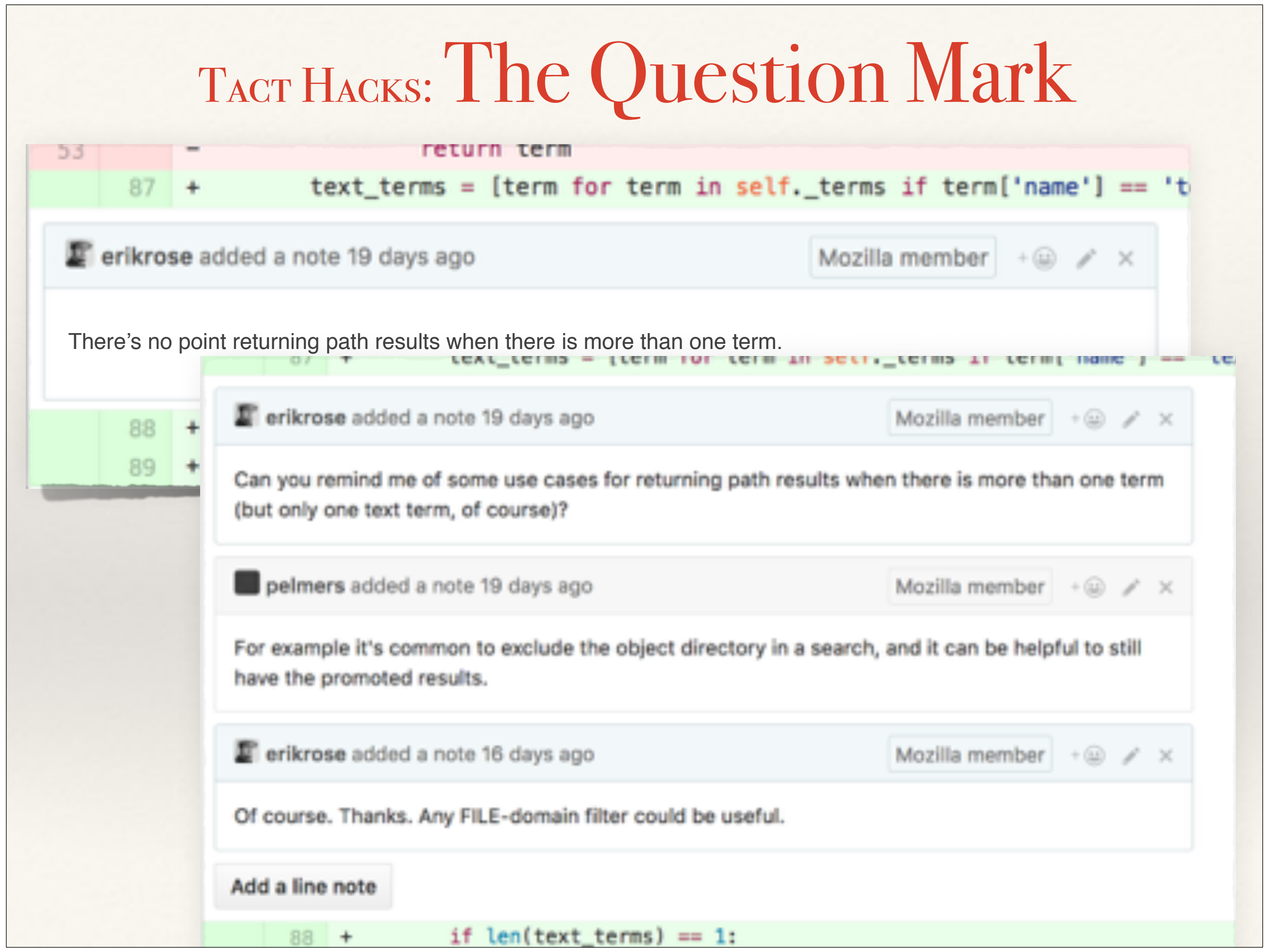


Tact Hacks: The Question Mark

shift + /

On many of the latest computing machines, you can make one by pressing shift-slash.

Tact Hacks: The Question Mark



```
53      -               return term
    87  +           text_terms = [term for term in self._terms if term['name'] == 't
```

erikrose added a note 19 days ago — Mozilla member

There's no point returning path results when there is more than one term.

erikrose added a note 19 days ago — Mozilla member

Can you remind me of some use cases for returning path results when there is more than one term (but only one text term, of course)?

pelmers added a note 19 days ago — Mozilla member

For example it's common to exclude the object directory in a search, and it can be helpful to still have the promoted results.

erikrose added a note 16 days ago — Mozilla member

Of course. Thanks. Any FILE-domain filter could be useful.

Add a line note

```
    88  +           if len(text_terms) == 1:
```

Just feel the difference between…

- “There's no point returning path results when there is more than one term.” vs. **<u>“Can you remind me of some use cases for returning path results when there is more than one term?”</u>**

This is an actual review.

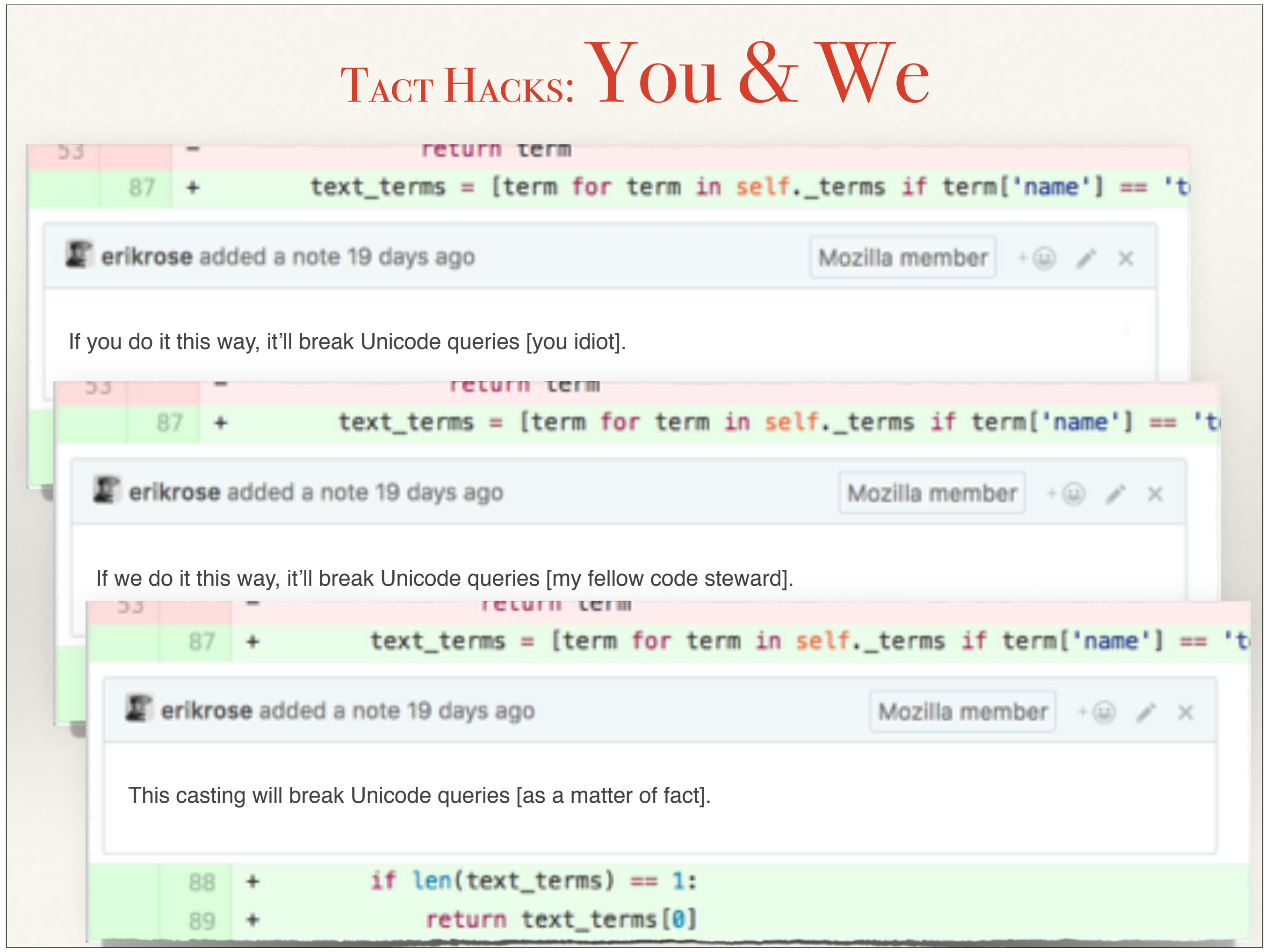
- Maybe you missed something, after all (saves face!)

- or “If I do this, it crashes. Do you get that as well?”

Pro:

- Activates the Socratic method: great for teaching because it makes people think along with you, not just get battered with assertions

Tact Hacks: You & We



```
53      -            return term
   87   +        text_terms = [term for term in self._terms if term['name'] == 't
```

erikrose added a note 19 days ago — Mozilla member

If you do it this way, it'll break Unicode queries [you idiot].

```
53      -            return term
   87   +        text_terms = [term for term in self._terms if term['name'] == 't
```

erikrose added a note 19 days ago — Mozilla member

If we do it this way, it'll break Unicode queries [my fellow code steward].

```
53      -            return term
   87   +        text_terms = [term for term in self._terms if term['name'] == 't
```

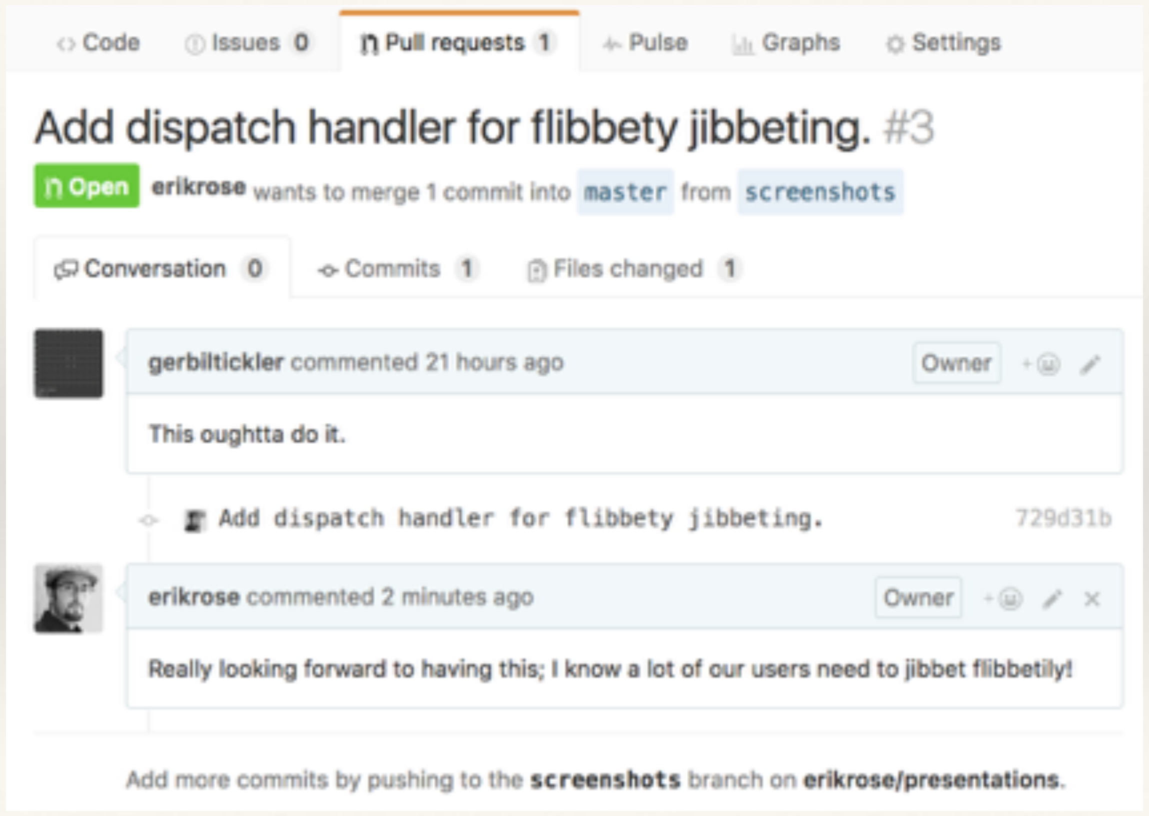
erikrose added a note 19 days ago — Mozilla member

This casting will break Unicode queries [as a matter of fact].

```
   88   +        if len(text_terms) == 1:
   89   +            return text_terms[0]
```

- An implied "you idiot"
- **A recognition of building something together**
- "**This.**" It's always safe to keep it about the code.

Tact Hacks: Compliments



Add dispatch handler for flibbety jibbeting. #3

Open erikrose wants to merge 1 commit into `master` from `screenshots`

gerbiltickler commented 21 hours ago Owner

This oughtta do it.

`Add dispatch handler for flibbety jibbeting.` 729d31b

erikrose commented 2 minutes ago Owner

Really looking forward to having this; I know a lot of our users need to jibbet flibbetily!

Add more commits by pushing to the **screenshots** branch on **erikrose/presentations**.

I like to write something encouraging as a top-level comment about the patch in general, while I'm still excited about the actual feature it delivers, before I dig in and discover the code is terrible:

"Aw, man, I can't wait to land this. I'm going to love being able to close Flash tabs without reaching for the mouse. Now let's get these couple things polished up so we can merge it."

- "Good catch!"
  These comments would make a good visual, as actual line comments on a PR.
  - Also makes a good reply back to the reviewer
- A favorite of mine: "Thank you so much for refactoring this scary mess."

Tact Hacks: 💩┬─┬ ¯_(🎀ツ)

- **Emoticons**
  - Emoji
  - GIFs
    - Keeps it lighthearted
- That wraps it up for tact hacks. Now let's look at some review antipatterns.

TL; DR; LGTM

- TLDR;LGTM

  A 1000-line review gets 1 comment. A 10-line one gets 5.
  - Feel free to ask for a more reviewable commit structure, better commit messages, more comments, a couple of paragraphs of prose explaining the approach. If you can't understand it now (with somebody around to answer questions), just imagine the trouble in a year.
    - Likewise, when you submit a patch, structure it so you'd be happy to review it.
      - The smaller the commits, the better.
      - Use a util like GitX that makes it easy to chunk commits.
    - And definitely read your diffs first before submitting.
      - Catch print statements and debugger breakpoints
      - Hugely worth it to get a diff viewer you like
      - Diff-viewing is also a good time to make sure you've addressed all the review comments.
  - Don't burn yourself out on a big review; some studies have found a decrease in review effectiveness after about an hour. Attack it in chunks.

While you're at it…

- (the inverse of TLDR;LGTM) Fix the rest of this file while you're at it.
    - Don't expect them to fix the whole module if they touch part of it.
        - The idea is for the codebase to be GettingBetter, not BeingPerfect. You can always come back and fix other parts later, and it's okay to point that out, but make clear that it doesn't block the merge (and shouldn't distract them).

Nitpicks

```
# Group lines into files:
for path, lines in groupby(results, lambda r: r['path'][0]):  # noqa
    lines = list(lines)
    highlit_path = highlight(  # noqa
        path,
        chain.from_iterable((h(lines[0]) for h in  # noqa
                             path_highlighters)))
    here_is_some_new(code, that.is_ridiculously_long_than(the_surrounding_code)).and_thus(really).distracting("isn't it?")
    icon_for_path = icon(path)
    yield (icon_for_path,
           highlit_path,
           [(line['number'][0],
             highlight(line['content'][0].rstrip('\n\r'),
                       chain.from_iterable(h(line) for h in
                                           content_highlighters)))
            for line in lines])
```

- Having a style guide or a linter smooths things along, saves back-and-forth.
    - But overzealous linters the hurt the understandability of code just make me angry. Code is for humans.
- We have a bunch of decent style guides at Mozilla; you should just adopt them when possible and save everybody work and make it easy to move code and people from project to project.
- Try to be adaptable. You can always do your favorite style on your personal stuff.
- Style: consistency within a function, then within a file, then within a module, then within a project.
  Avoid inadvertently emphasizing your changes
- Remember what it costs per hour to have you sit there, and throttle accordingly.

Slow Turnarounds

- Even if you aren't comprehensive
    - You can say “I'm not going to have the time to look at the rest of this for a week. Someone else can feel free to step in.” Don't feel like you married the patch.
- Stuff falls out of working memory pretty fast, and it's expensive to load back in.
- It's energizing to get your patches turned around quickly.
- If you have to say "no" completely or ask someone to rethink their approach, better to do it quickly so they don't waste their time building more castles on the sand.

- That wraps up the quick-tip portion of the program. Now let's see what some of the core emotional issues are that drive bad reviews.

Those Pesky Human Emotions

- …behind a lot of review failures. For instance…

Insecurity

- **Insecurity**
  a kind of fear
  - Who in the room has ever secretly felt (or feared) they're not as competent as the engineers around you?
    (If they don't put up their hands: "Of course you don't. It's a secret.")
  - Almost everybody does. It doesn't really go away. Realize nobody's judging you; they're too busy worrying about people judging them.
  - Who's afraid their imposter syndrome isn't as good as everybody else's?
  - Realize you've already made it.
    - If you're hired, congratulations. We don't have an easy hiring process. You're competent to do this job.
    - Sometimes somebody will find a problem in your PR because they know something you don't. (in which case you'll know it afterward)
    - Sometimes it's just having a fresh set of eyes.
  - In general, it's been a help to me to notice when I'm feeling defensive or angry.
    - Why am I feeling that way? Sometimes that reveals things I believe that aren't true or things I need to think about.
    - I generally make bad decisions and say things I regret when feeling that way.
    - It's a good time to talk a break and come back with a clear head.

Feeling Short on Time

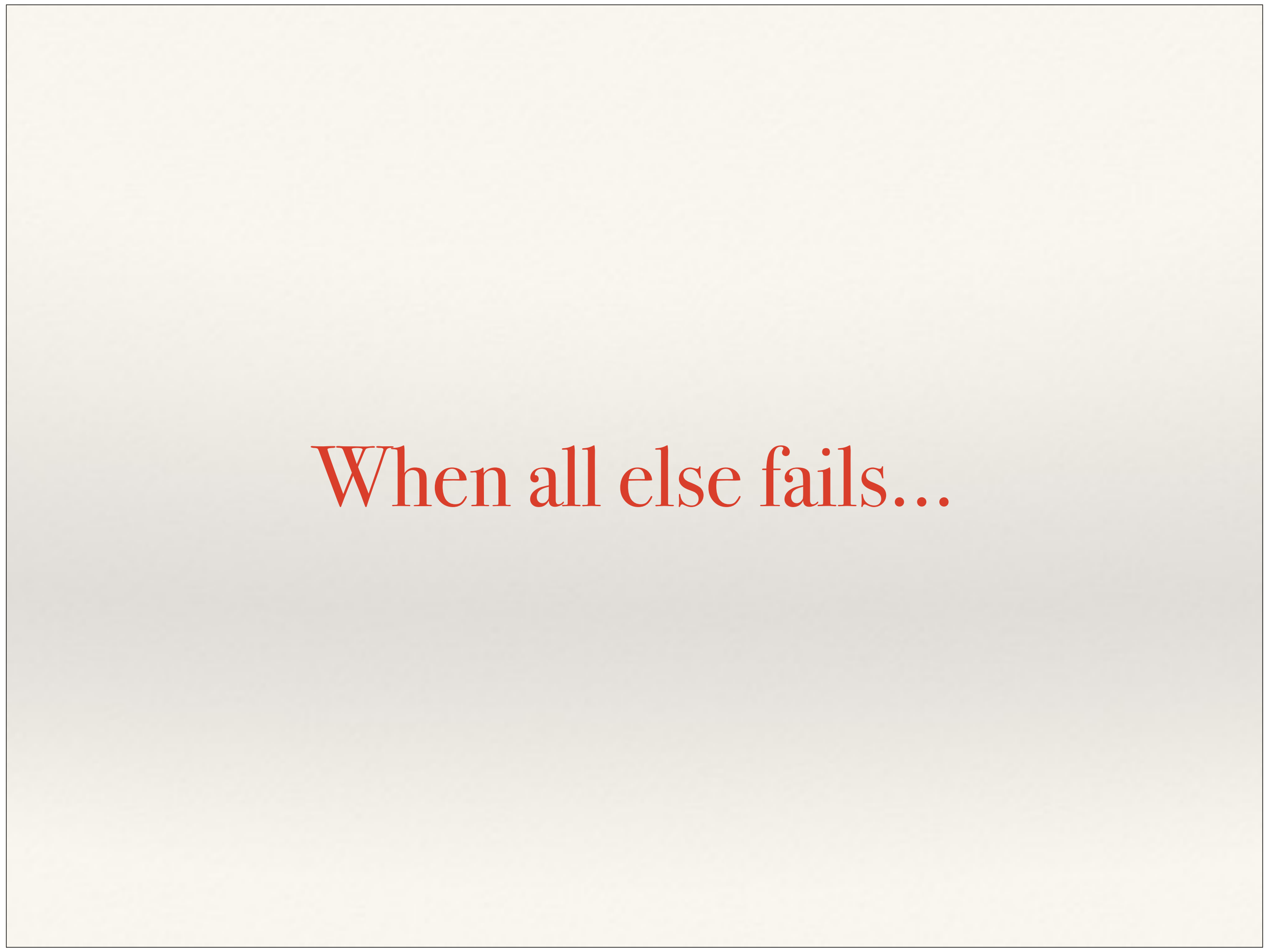
There will always be more to review than can be reviewed. So we have to learn to adapt to that reality: lower standards, never sleep, or just go at a sustainable pace. It can do a lot for a contrib's peace of mind just to know you've seen that the patch is there and that you haven't forgotten about it. Sometimes I even let people know "It's next in my queue."

- Consider: what's the risk if there's something wrong with it?
    - Will it cause a lot of damage?
    - Will it be easy to fix, or is there, say, a data-exchange format being defined?
- Dealing with newcomers
    - One time I often feel like I'm getting a bad ROI is with newcomers.
    - We've all got to start where we're at. Don't make fun of scrawny people weightlifting.
        - And we *must* embrace newcomers unless we plan to live forever.
        - It's flattering when someone new wants to contribute to your project.
    - But I don't spend much time on new contribs the moment they show up.
        - But my investment parallels theirs.
    - Leveling up
        - They don't know where to start
            - Links to dev docs, a few sentences of explanation and ideas where to start
            - Links to resources to learn skills if they don't have them
        - First I will fix their patch (geared for one-timers so patches don't rot)
            - "Polish-up" commits
                - So their name still gets into the commit log and they feel good
        - Next, I just comment and request fixes
        - Then I require tests

The Trust Bank

Very important for a remote company.

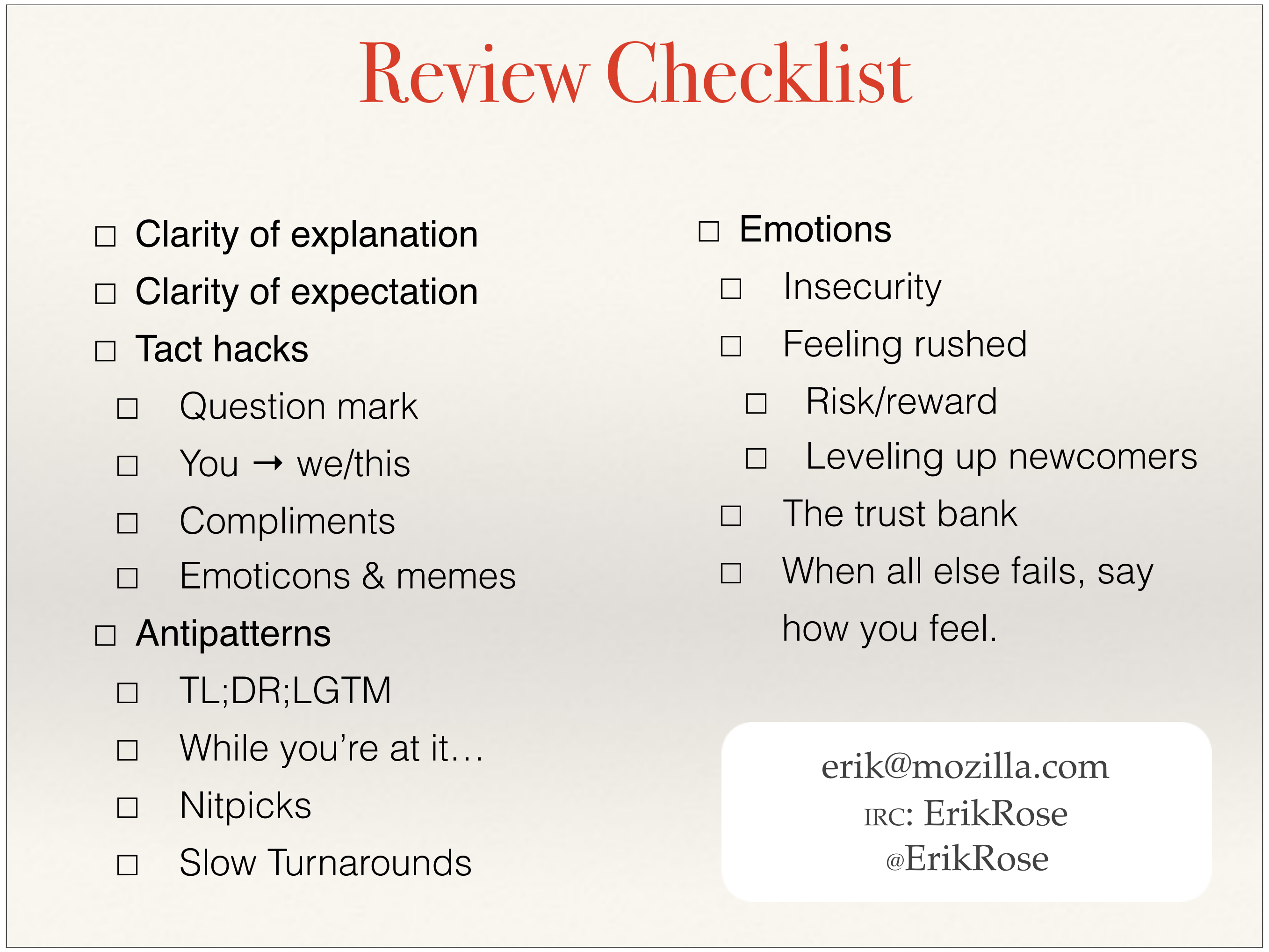
- When you're in person with your team, make a point of establishing good relations with them.
    - Then when you have to critique each other or talk on a low-bandwidth medium like IRC, you'll know how to take each other. You'll have some built-up offense-buffering.



When all else fails…

Other feelings

- When you can't decide what to say (something feels too bad to merge), you can always fall back to articulating your emotions: "I'm worried that if we make the destination param optional, people will write to the global datastore by default, and a mess will ensue."
- You don't have to be definitive, decisive, or an authority all the time. Just treat everybody like adults and articulate your feelings. I've said "I'm happy you want to work on that, but I'm afraid that you'll end up in merge hell if I land the elasticsearch stuff first." Inviting them into the tradeoff is a heckuva lot better than "no!". (If they don't mine merge hell, they can go for it!)



Review Checklist

- ☐ Clarity of explanation
- ☐ Clarity of expectation
- ☐ Tact hacks
  - ☐ Question mark
  - ☐ You ➝ we/this
  - ☐ Compliments
  - ☐ Emoticons & memes
- ☐ Antipatterns
  - ☐ TL;DR;LGTM
  - ☐ While you're at it…
  - ☐ Nitpicks
  - ☐ Slow Turnarounds
- ☐ Emotions
  - ☐ Insecurity
  - ☐ Feeling rushed
    - ☐ Risk/reward
    - ☐ Leveling up newcomers
  - ☐ The trust bank
  - ☐ When all else fails, say how you feel.

erik@mozilla.com
IRC: ErikRose
@ErikRose

- Here's a checklist to remind you of what we covered today.
  - Stick it on your desk, and draw from it to make excellent **code**, excellent **coders**, and excellent use of **time**.
  - And **welcome** to Mozilla!
  - Slides are on slideshare

Thank You

grinch@grinchcentral.com
Freenode IRC: ErikRose
Twitter: @ErikRose
https://www.slideshare.net/ErikRose

erikrose added a note 19 days ago

Mozilla member

Have an excellent day. 🍰

Thanks for being such an excellent audience!