Cabinet Office

Be a good Git Citizen

How git can help you and your reviews

#proj-supporting-juniors

What we talking about?

- Git commits, why do we care?
- When they are good, how does that help?
- Tips for writing a good commit message
- How good commits can help you with a review

Git Commits

Good commits help you understand why

Recent example

Good commits are atomic

This means that each commit:

- is self-contained
- only includes changes that achieve a specific step
- should be in a logical order

Good commits tell a story

Consider being asked to review this.
Compare it to being asked to review this.

Good commits, serve user needs

Who are your users?

- New or returning devs, who want to get up to speed
- Folks in other departments
- Members of the public (Public money, public code)
- Future developers (including yourself!)

Good commits, are better than comments

If you are about to leave a comment, ask - should this be a commit message?

Good commits, tell me *why* more than *what*

I should be able to work out what you did by comparing to what was there before (the diff).

But why you did it… only you know for sure.
Tell us before you forget

Good commits - follow a format

```
Capitalized, short (50 chars or less) summary

More detailed explanatory text, if necessary. Wrap it to about 72 characters or so. In some
contexts, the first line is treated as the subject of an email and the rest of the text as the
body. The blank line separating the summary from the body is critical (unless you omit the body
entirely); tools like rebase can get confused if you run the two together.

Write your commit message in the present tense: “Fix bug” and not “Fixed bug.” This convention
matches up with commit messages generated by commands like git merge and git revert.

Further paragraphs come after blank lines.

    . Bullet points are okay, too
    . Typically a hyphen or asterisk is used for the bullet, preceded by a single space, with
      blank lines in between, but conventions vary here
    . Use a hanging indent
```

Good commits - have links

```
Redirect user to the requested page after login


https://trello.com/path/to/relevant/card


Users were being redirected to the home page after login, which is less
useful than redirecting to the page they had originally requested before
being redirected to the login form.
* Store requested path in a session variable
* Redirect to the stored location after successfully logging in the user
```

Good commits take time

Commits are a core part of the work, take time on them. Use them to help you think about what you are doing and why

Good commits… can happen later

Ideally you'd commit as you go.
But we are human.

It is OK to do a commit that's just "WIP", but come back and fix it later.

This requires confidence with either `reset` or `rebase`

Good commits make for good reviews

Let’s take a look at the review guidelines

Consider how good commits help you with this

Reviews?

Good reviews are a safety net

But… not a great one. Lots of bugs get through

They’re really about something else…

What reviews are great for!

Increased knowledge transfer
“Huh, you can do *that*?”

Increased team awareness
“So this is how that works…”

Finding alternative solutions
“What if we tried… this…”

So code reviews are not a permission gate

Code reviews are a discussion with your colleagues

Getting better at that, you are getting better at communication.

So code reviews are not a permission gate

- **Modern standards approach**. You can competently apply a modern standards approach and guide others to do so. (Skill level: practitioner)

  Programming and build (software engineering). You can collaborate with others when necessary to review specifications. You can use the agreed specifications to design, code, test and document programs or scripts of medium-to-high complexity, using the right standards and tools. (Skill level: practitioner)

- **Prototyping**. You can approach prototyping as a team activity, actively soliciting prototypes and testing with others. You can establish design

Last thoughts on reviews

There’s a negativity bias in written feedback

Humans react worse to written feedback, compared to the same feedback given verbally

Solution: Compensate for that by being more complimentary / friendly that you might otherwise be

Ask, Don’t tell

What do you think about..?

Did you consider..?

Can you clarify..?

Solution: *Good comments end in question marks*

Avoid the word “Just” in your questions

“Did you consider..?”

vs

“Why didn’t you just..?”

Take care of comments that put folks on the defensive

Let’s review a PR

https://github.com/alphagov/digital-identity-team-manual/pull/203

Some links

Working with Git - The GDS way

How to review code - The GDS way

RailsConf 2015 - Implementing a Strong Code-Review Culture

Thoughtbot - Code Reviews