Emails to Myself

Composing Effective Git Commit Messages

Speaker: Jason Franklin
Date: 2019-10-10
Slides: https://bit.ly/325MOhT
Files: https://bit.ly/2VvzD7k

The Roadmap

- General ideas
- Rules
- Example implementation
- Resources

Part I: General Ideas

Part I: General Ideas

- You should care about your commit messages.
- Commit messages make the repository history useful (blame, log, etc.).
- Each commit message is an opportunity to help and to establish yourself as a good collaborator!
- Commit messages over commentary.

Part I: General Ideas

Elements of good style...

- The **content** of a commit message should explain the *how* and the *why*. The *what* is already given in the diff.
- The **format** of a commit message should be defined so that it can be displayed and shared easily (email, log, etc.).
- Commit **metadata** should have a place in the message along with the explanation (ticket numbers, URLs, etc.).

Part II: Rules

Part II: Rules

1. Separate summary line from body with a blank line.
2. Limit the subject line to 50 characters.
3. Capitalize the subject line.
4. Do not end the subject line with a period.

Part II: Rules

5. Use the imperative mood in the subject line.

Bad: I Updated Flask to version 2.2.2

Bad: Flask was updated to v2.2.2

Good: Update Flask to version 2.2.2

Pro tip: Finish the sentence!

If applied this commit will ...

Part II: Rules

6. Wrap the body at 68 characters.

7. Use the body to explain *how* and *why*.

Part II: Rules

You don’t need to worry about…

- Leading and trailing empty lines
- Trailing whitespace
- Consecutive empty lines
- Removing # lines

Note: See “man git-commit” for how to customize this.

Part III: Example Implementation

Part III: Example

Some commands to run first…

```
$ git config --global core.editor vim
$ git config --global commit.verbose true
```

Part III: Example (vimscript)

```
if has('syntax')

    " Show a line at the text width limit (optional).
    setlocal colorcolumn=70

   " Enable spell checking if available.
   setlocal spell
endif
```

Part III: Example (vimscript)

```vim
" Indentation settings.
setlocal expandtab
setlocal shiftwidth=2 softtabstop=2
setlocal tabstop=8

" Wrap the text at 68 characters.
setlocal textwidth=68
```

Part III: Example

In the talk notes, you’ll find two files:

1. gitcommit.vim (stand-alone plugin, preferred)
2. gitcommit_vimrc.vim (to paste into .vimrc)

Use these files to make editing commits in Vim easier!

Part IV: Resources

Part IV: Resources

- https://chris.beams.io/posts/git-commit/
- From the talk notes: “template.txt”
- https://github.com/Subsurface-divelog/subsurface

Part V: Questions?