Better commits

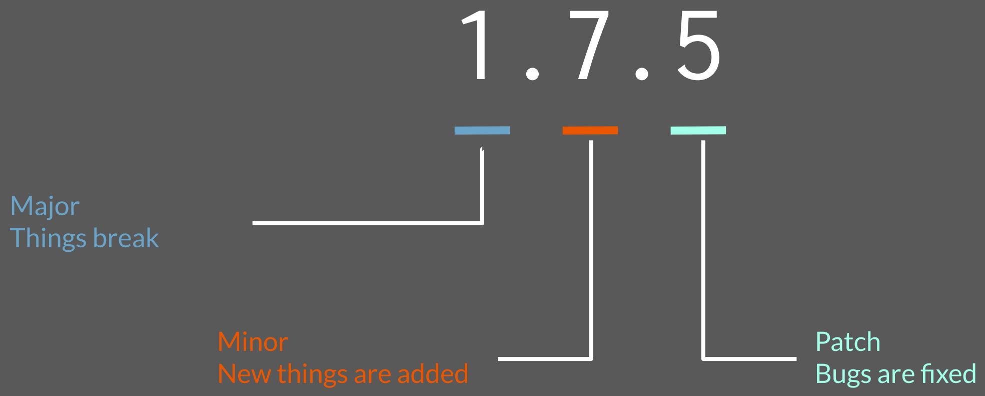
How can commits improve the life on the project?

Agenda

1. Conventional commits
2. A case study: Citerate
3. What about HubDigital?

Conventional commits?

Semver



1.7.5

Major
Things break

Minor
New things are added

Patch
Bugs are fixed

Semver is nice but...

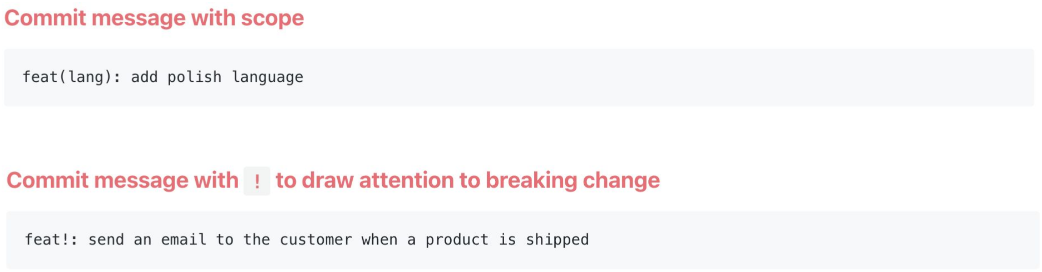
It is hard to apply:

- What happened?
- Who says if what happened is breaking?

→ Git history is here to help

Conventional commits

- readable git history for humans AND machines
- Lightweight convention (full spec here):
  - `<type>[optional scope]: <description>`
  - 3 types: fix / feat / ! (BREAKING)
- Examples:



Commit message with scope

```
feat(lang): add polish language
```

Commit message with `!` to draw attention to breaking change

```
feat!: send an email to the customer when a product is shipped
```

Benefits

- Changelog automated
- Version bump automated
- Better commits because of the light constraints

Case study: Citerate

Context

- Game project
- I’m the sole contributor to the codebase
- 3 platforms to handle (web, iOS, Android)
- Full devops :’) but not much time
- High expectations from the client

The pieces

- https://gitlab.com/karen/terrence/-/commits/master/ Commit history
- https://gitlab.com/karen/terrence/-/blob/master/CHANGELOG.md Generated changelog
- https://gitlab.com/karen/terrence/-/blob/master/package.json#L21 The comand to update the versions / create git tags / push and create packages

The tools

Standard version

What about HubDigital?

A proposal

- Stick to a simple implementation
- First start to commit with the new convention
- Then implement the standard version tool for changelogs