Commit message structure

<type>[scope]: **<description>**
[body]
[footer]

A commit that has the text `BREAKING CHANGE:` at the beginning of its optional body or footer section introduces a breaking API change.

Types

`feat`	A new feature
`fix`	A bug fix
`docs`	Documentation only changes
`style`	Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)
`refactor`	A code change that neither fixes a bug nor adds a feature
`perf`	A code change that improves performance
`test`	Adding missing tests or correcting existing tests
`build`	Changes that affect the build system or external dependencies (e.g. Gradle, Maven)
`ci`	Changes to our CI configuration files and scripts (e.g. GitLab, Jenkins)
`chore`	Other changes that don't modify src or test files
`revert`	Reverts a previous commit

Specification

1. Commits **MUST** be prefixed with a type, which consists of a verb, `feat`, `fix`, etc., followed by a colon and a space.

2. The type `feat` **MUST** be used when a commit adds a new feature to your application or library.

3. The type `fix` **MUST** be used when a commit represents a bug fix for your application.

4. An optional scope **MAY** be provided after a type. A scope is a phrase describing a section of the codebase enclosed in parenthesis (e.g. `fix(pa rser)`:).

5. A description **MUST** immediately follow the type/scope prefix. The description is a short description of the pull request (e.g.,`fix: array pa rsing issue when multiple spaces were contained in string`).

6. A longer commit body **MAY** be provided after the short description. The body **MUST** begin one blank line after the description.

7. A footer **MAY** be provided one blank line after the body. The footer SHOULD contain additional meta-information about the pull-request (such as the issues it fixes, e.g., `fixes #13, #5`).

8. Breaking changes **MUST** be indicated at the very beginning of the footer or body section of a commit. A breaking change **MUST** consist of the uppercase text `BREAKING CHANGE`, followed by a colon and a space.

9. A description **MUST** be provided after the BREAKING CHANGE: , describing what has changed about the API (e.g.`BREAKING CHANGE: en viro nment variables now take precedence over config files`).

10. Types other than `feat` and `fix` **MAY** be used in your commit messages.

By **marco.lizza**
cheatography.com/marco-lizza/

Published 8th August, 2023.
Last updated 8th August, 2023.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com