

Description

A specification for adding human and machine readable meaning to commit messages from Conventional Commits

Commit message structure

```
<type>[optional scope]: <description>
[optional body]
[optional 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

fix	Correct code issue (Patch)	-
feat	Introduce new feature (Minor)	-
BREAKING CHANGE	Breaking backward compatibility (Major)	-
build	Build system changes	Angular convention
chore	Maintenance or tooling	Angular convention
ci	Continuous integration adjustments	Angular convention
docs	Documentation changes made	Angular convention
style	Code style updates	Angular convention
refactor	Code restructuring done	Angular convention
perf	Performance improvement made	Angular convention
test	Testing-related changes	Angular convention

Several types use words based on Angular's conventional commit

Specification

- ✓ Commits **MUST** be prefixed with a type, which consists of a noun, feat, fix, etc., followed by the **OPTIONAL** scope, **OPTIONAL** !, and **REQUIRED** terminal colon and space.
- ✓ The type feat **MUST** be used when a commit adds a new feature to your application or library.
- ✓ The type fix **MUST** be used when a commit represents a bug fix for your application.
- ✓ A scope **MAY** be provided after a type. A scope **MUST** consist of a noun describing a section of the codebase surrounded by parenthesis, e.g., fix(parser):
- ✓ A description **MUST** immediately follow the colon and space after the type/scope prefix. The description is a short summary of the code changes, e.g., fix: array parsing issue when multiple spaces were contained in string.
- ✓ A longer commit body **MAY** be provided after the short description, providing additional contextual information about the code changes. The body **MUST** begin one blank line after the description.
- ✓ A commit body is free-form and **MAY** consist of any number of newline separated paragraphs.
- ✓ One or more footers **MAY** be provided one blank line after the body. Each footer **MUST** consist of a word token, followed by either a :<space> or <space># separator, followed by a string value (this is inspired by the git trailer convention).
- ✓ A footer's token **MUST** use - in place of whitespace characters, e.g., Acked-by (this helps differentiate the footer section from a multi-paragraph body). An exception is made for BREAKING CHANGE, which **MAY** also be used as a token.
- ✓ A footer's value **MAY** contain spaces and newlines, and parsing **MUST** terminate when the next valid footer token/separator pair is observed.
- ✓ Breaking changes **MUST** be indicated in the type/scope prefix of a commit, or as an entry in the footer.



Specification (cont)

- ✓ If included as a footer, a breaking change **MUST** consist of the uppercase text **BREAKING CHANGE**, followed by a colon, space, and description, e.g., **BREAKING CHANGE:** environment variables now take precedence over config files.
- ✓ If included in the type/scope prefix, breaking changes **MUST** be indicated by a **!** immediately before the **:**. If **!** is used, **BREAKING CHANGE:** **MAY** be omitted from the footer section, and the commit description **SHALL** be used to describe the breaking change.
- ✓ Types other than feat and fix **MAY** be used in your commit messages, e.g., docs: update ref docs.
- ✓ The units of information that make up Conventional Commits **MUST NOT** be treated as case sensitive by implementors, with the exception of **BREAKING CHANGE** which **MUST** be uppercase.
- ✓ **BREAKING-CHANGE** **MUST** be synonymous with **BREAKING CHANGE**, when used as a token in a footer.



By **finuex**
cheatography.com/finuex/

Published 13th July, 2024.
Last updated 13th July, 2024.
Page 2 of 2.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish
Yours!
<https://apollopad.com>