**Conventional Commits**

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

Conventional Commits 1.0.0

The commit message should be structured as follows:

Header

|  |
| --- |
| **type(scope): subject** |
| **Body(Optional)** |
| **Footer(Optional** |

**The Header**

The header is a **mandatory** line that simply describes the purpose of the change

Better yet, it consists of three parts in itself:

1. Type - a short prefix that represents the **kind** of the change
2. Scope - optional information (attached to the prefix) that represents the **context** of the change
3. Subject/description - represents a concise **description** of the actual change

Example:

git commit -m "fix(core): remove deprecated and defunct wtf\* apis"

**The Body**

The body is **optional** lines that introduce the motivation behind the change or just describing slightly more detailed information.

Example:

git commit -m "fix(core): remove deprecated and defunct wtf\* apis" -m "These apis have been deprecated in v8, so they should stick around till v10, but since they are defunct we are removing them early so that they don't take up payload size."

## **The Footer**

The footer is **optional** lines that mention consequences which stems from the change - such as announcing a breaking change, linking closed issues, mentioning contributors and so on.

Example:

git commit -m "fix(core): remove deprecated and defunct wtf\* apis" -m "These apis have been deprecated in v8, so they should stick around till v10, but since they are defunct we are removing them early so that they don't take up payload size." -m "PR Close #33949"

Semantic Versioning is all about versioning API.

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Description** | **Release** | **Examples** |
| **Feat:** | A new feature | 1.0.0 → 1.1.0 | “feat(parser): improve error handling” |
| **Fix:** | A bug fix | 1.0.0 → 1.0.1 | “fix(router): resolve shorthand property declaration” |
| **Breaking change:** | Major/breaking API change | 1.0.0 → 2.0.0 | “BREAKING CHANGE(core): introduced new API blog ” |
| **docs:** | Documentation Only changes | 1.0.0 → 1.0.0 | “docs: improve docs for linting**”** |
| **style:** | Changes that do not affect the meaning of the code (white-spacing, missing semicolon, formatting). | 1.0.0 → 1.0.0 | “style: use single quote consistently” |
| **refactor:** | A code change that neither fixes a bug or adds a feature | 1.0.0 → 1.0.0 | “refactor: rename file name” |
| **perf:** | A code change that improves performance | 1.0.0 → 1.0.0 | “perf: don’t store public input names in two places” |
| **test:** | Changes related to tests - such as refactoring existing tests or adding new tests. | 1.0.0 → 1.0.0 | “test: fix tests” |
| **chore:** | Changes related to the build system (involving scripts, configurations or tools) and package dependencies. | 1.0.0 → 1.0.0 | “build: update jest to 1.0.14” |