**Semantic Versioning**

***semantic-release*** automates the whole package release workflow including determining the next version number, generating the release notes, and publishing the package.

Given a version number MAJOR.MINOR.PATCH, increment the:

1. MAJOR version when you make incompatible API changes,
2. MINOR version when you add functionality in a backward-compatible manner, and
3. PATCH version when you make backward-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

Refer: - https://semver.org

**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 describe 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 that stem 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**

1. **Production**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Description** | **Version Change** | **Examples** |
| **BREAKING CHANGE:** | Major/breaking API change | **Major**  (Note that the BREAKING CHANGE: token must be in the footer of the commit) | *“BREAKING CHANGE: The graphiteWidth option has been removed. The default graphite width of 10mm is always used for performance reasons. ”* |
| **feat(scope):** | A new feature | **Minor**  (Note that the feat(scope): token must be in the subject of the commit) | *“feat(pencil): add 'graphiteWidth' option”* |
| **fix(scope):** | A bug fix | **Patch**  (Note that the fix(scope): token must be in the subject of the commit) | *“fix(pencil): stop graphite breaking when too much pressure applied”* |
| **perf(scope):** | A code change that improves performance |  | *“perf: don’t store public input names in two places”* |

1. **Development**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Description** | **Version Change** | **Examples** |
| **docs:** | Documentation Only changes |  | *“docs: improve docs for linting****”*** |
| **style:** | Changes that do not affect the meaning of the code (white-spacing, missing semicolons, formatting). |  | *“style: use single quote consistently”* |
| **refactor:** | A code change that neither fixes a bug nor adds a feature |  | *“refactor: rename file name”* |
| **test:** | Changes related to tests - such as refactoring existing tests or adding new tests. |  | *“test: fix tests”* |
| **chore:** | Changes related to the build system (involving scripts, configurations, or tools) and package dependencies. |  | *“build: update jest to 1.0.14”* |