**Angular 团队的 commit 规范**

它的 message 格式如下:

<type>(<scope>): <subject>

// 空一行

<body>

// 空一行

<footer>

分别对应 Commit message 的三个部分：Header，Body 和 Footer。

**一、Header**

Header 部分只有一行，包括三个字段：type（必需）、scope（可选）和subject（必需）。

* **type:** 用于说明 commit 的类型。
  + feat: 新增feature
  + fix: 修复bug
  + docs: 仅仅修改了文档，如readme.md
  + style: 仅仅是对格式进行修改，如逗号、缩进、空格等。不改变代码逻辑。
  + refactor: 代码重构，没有新增功能或修复bug
  + perf: 优化相关，如提升性能、用户体验等。
  + test: 测试用例，包括单元测试、集成测试。
  + chore: 改变构建流程、或者增加依赖库、工具等。
  + revert: 版本回滚
* **scope:** 用于说明 commit 影响的范围，比如: views, component, utils, test...
* **subject:** commit 目的的简短描述

**二、Body**

对本次 commit 修改内容的具体描述, 可以分为多行。如下所示:

# body: 72-character wrapped. This should answer:

# \* Why was this change necessary?

# \* How does it address the problem?

# \* Are there any side effects?

# initial commit

**三、Footer**

一些备注, 通常是 BREAKING CHANGE(当前代码与上一个版本不兼容) 或修复的 bug(关闭 Issue) 的链接。

简单介绍完上面的规范，我们下面来说一下commit.template，也就是 git 提交信息模板。

| **type** | **detail** | **说明** |
| --- | --- | --- |
| feat | A new feature | 新功能 |
| fix | A bug fix | 修复 bug |
| docs | Documentation only changes | 文档修改 |
| style | Changes that do not affect the meaning of the code (white-space, formatting, missing semicolons, etc) | 格式（不影响代码运行的变动） |
| refactor | A code change that neither fixes a bug nor adds a feature | 重构 |
| perf | A code change that improves performance | 提高性能 |
| test | Adding missing tests or correcting existing tests | 添加缺失测试或更正现有测试 |
| build | Changes that affect the build system or external dependencies (example scopes: gulp, broccoli, npm) | 依赖的外部资源变化 |
| chore | Other changes that don’t modify src or test files | 构建过程或辅助工具的变动 |
| revert | Reverts a previous commit | 恢复先前的提交 |