GitLab-
Semantic Release

以 TypeScript Node.JS Module
Development 为例

什么是 Semantic Release

- https://semver.org/lang/zh-CN/

语义化版本 2.0.0

摘要

版本格式：主版本号.次版本号.修订号，版本号递增规则如下：

1. 主版本号：当你做了不兼容的 API 修改，
2. 次版本号：当你做了向下兼容的功能性新增，
3. 修订号：当你做了向下兼容的问题修正。

先行版本号及版本编译元数据可以加到“主版本号.次版本号.修订号”的后面，作为延伸。

Commit Message -> version

Commit message	Release type
fix(pencil): stop graphite breaking when too much pressure applied	~~Patch~~ Fix Release
feat(pencil): add 'graphiteWidth' option	~~Minor~~ Feature Release
perf(pencil): remove graphiteWidth option BREAKING CHANGE: The graphiteWidth option has been removed. The default graphite width of 10mm is always used for performance reasons.	~~Major~~ Breaking Release (Note that the BREAKING CHANGE: token must be in the footer of the commit)

版本号是根据 commit message 计算出来的

14 Jun, 2022 1 commit

fix: 升级scuttlebutt-pull到8.4.0 支持sync的缓解时间跳变功能
quwenzhe authored 13 hours ago — `bc700367`

13 Jun, 2022 1 commit

feat: update sb 支持校准时间戳; 更新 ci
manzhiyuan authored 1 day ago — `f2c4c84c`

05 May, 2022 1 commit

fix: 升级 scuttlebutt-pull 到8.1.2 减少 update 不必要地传播
shenrong authored 1 month ago — `386e1d12`

29 Apr, 2022 1 commit

fix: 升级 @li-mesh/scuttlebutt-pull 到8.1.1，以解决 Duplex 导致的资源泄露
shenrong authored 1 month ago — `575df235`

13 Apr, 2022 1 commit

docs: examples 增加为不同topicName 增加不同的 debounce 事件函数的例子
shenrong authored 2 months ago — `32d81826`

31 Mar, 2022 1 commit

feat: topic 增加 limitsOfTopic 现在，push/pushWithTs 返回值表示是否成功压入 topic
shenrong authored 2 months ago — `5cf3a8bf`

26 Mar, 2022 1 commit

feat: 增加 disableEvent 构造可选项, 用于禁用通知事件, 解决压力测试 hang 住的问题
jacobbubu authored 2 months ago — `31d252b9`

07 Mar, 2022 1 commit

fix: 升级 @li-mesh/scuttlebutt-pull 到 v8
jacobbubu authored 3 months ago — `173931ce`

24 Jan, 2022 2 commits

Merge branch 'master' of gitlab.chehejia.com:li-mesh/r-topic
jacobbubu authored 4 months ago — `fe1f0946`

li-mesh › r-topic › Tags

Filter by tag name | Last updated | New tag

Tags give the ability to mark specific points in history as being important

- `v1.8.1` — `bc700367` · **fix:** 升级**scuttlebutt-pull**到**8.4.0** 支持**sync**的缓解时间跳变功能 · **13 hours ago**
- `v1.8.0` — `f2c4c84c` · **feat: update sb** 支持校准时间戳; 更新 **ci** · **1 day ago**
- `v1.7.2` — `386e1d12` · **fix:** 升级 **scuttlebutt-pull** 到**8.1.2** 减少 **update** 不必要地传播 · **1 month ago**
- `v1.7.1` — `575df235` · **fix:** 升级 **@li-mesh/scuttlebutt-pull** 到**8.1.1**，以解决 **Duplex** 导致的资源泄露 · **1 month ago**
- `v1.7.0` — `5cf3a8bf` · **feat: topic** 增加 **limitsOfTopic** 现在，**push/pushWithTs** 返回值表示是否成功压入 **topic** · **2 m**
- `v1.6.0` — `31d252b9` · **feat:** 增加 **disableEvent** 构造可选项, 用于禁用通知事件, 解决压力测试 **hang** 住的问题 · **2 mo**
- `v1.5.1` — `173931ce` · **fix:** 升级 **@li-mesh/scuttlebutt-pull** 到 **v8** · **3 months ago**
- `v1.5.0` — `fe1f0946` · **Merge branch 'master' of gitlab.chehejia.com:li-mesh/r-topic** · **4 months ago**
- `v1.4.1` — `3bb594dd` · **Merge branch 'dev' into 'master'** · **5 months ago**
- `v1.4.0` — `d0490556` · **feat: getTopic** 增加了按照晚于等于,早于等检索条件获取事件的可选项 · **9 months ago**
- `v1.3.2` — `fb37a1b4` · **fix:** 修复较早 **push** 的 **event** 如果同步给 **peer** 也是放在 **topic** 最后的问题 · **1 year ago**
- `v1.3.1` — `8aac94b0` · **fix:** 上级 **sb** 到 **7.3.1,** 同时配合暴露 **InternalMemoryStore** 的类型声明出来 · **1 year ago**

https://gitlab.chehejia.com/li-mesh/r-topic

脚手架

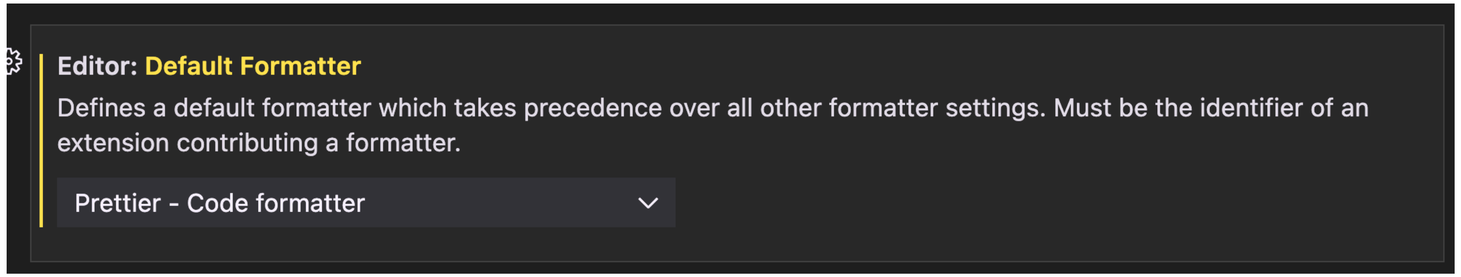
- https://gitlab.chehejia.com/li-mesh/typescript-library-starter-gl
  - 单 Repo 的脚手架，久经考验
- https://gitlab.chehejia.com/li-mesh/typescript-library-mono-gl
  - Monorepos 的脚手架，没有久经考验
- https://gitlab.chehejia.com/li-mesh/create-gl-repo

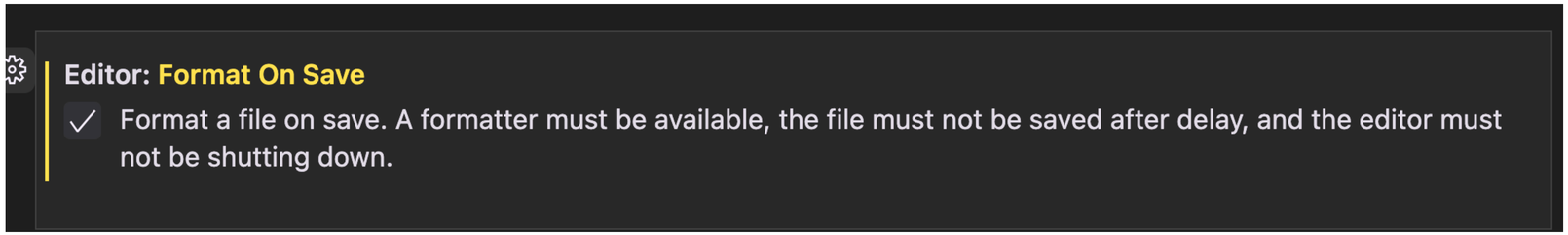
☕ Demo-new repo.

插播 - 代码格式化

- 编辑时保障：
  - .editorconfig, .prettierrc

VSCode





Commit Hooks

- npm i husky —save-dev
  - 先用 Husky 把所有 Git Hooks 挂住
- 然后在 .huskyrc.json 添加脚本

.huskyrc.json > ...

```
1 {
2   "hooks": {
3     "pre-commit": "npm run precommit",
4     "commit-msg": "commitlint -E HUSKY_GIT_PARAMS",
5     "pre-push": "npm run test:prod && npm run build"
6   }
7 }
```

第 3 行: 在 git commit 之前对 staged file 进行操作

第 4 行: 对 git commit -m 进行操作

第 5 行: 在 git push 之前进行运行

git add .
git commit -m "feat: initial commit"

git push

pre-commit → 针对 staged 代码

npm run precommit

用 prettier 格式化代码
用 eslint 来检查代码质量

静态质量

commit-msg → 针对 staged 代码

commitlint -E HUSKY_GIT_PARAMS
检查 commit message 的消息

语义化消息

pre-push → 单元测试，build 等

npm run test:prod &&
npm run build

运行质量

precommit

npm run precommit

- "precommit": "lint-staged"
  - lint-staged: 专门对 staged 文件进行操作的工具

package.json

```
36  "lint-staged": {
37    "{src,test}/**/*.ts": [
38      "prettier --write",
39      "eslint --fix"
40    ]
41  },
```

第 38 行: 提交前格式化代码，确保提交的代码格式一致

第 39 行: 对 ts/js 代码进行 linting 操作

https://github.com/okonet/lint-staged

撰写语义化的
commit message

"commit-msg": "commitlint -E HUSKY_GIT_PARAMS"

- commitlint: 是检查 commit message 规范的工具
- .commitlintrc.json 中包含具体采用哪种规范的配置

.commitlintrc.json > ...

```
1 {
2   "extends": [
3     "@commitlint/config-conventional"
4   ]
5 }
```

第 3 行: https://github.com/conventional-changelog/commitlint/blob/master/%40commitlint/config-conventional/index.js

https://github.com/conventional-changelog/commitlint

局部安装和全局安装

- npm i commitizen —global
  - 全局安装
- npm i commitizen —save-dev
  - 安转在当前 repo. 的 node_modules 下
  - 如果是可执行文件，会在 node_modules/.bin 下创建软链
  - 避免和其他 repo. 冲突，但是只有在 npm 自己的脚本内运行才会自动到 node_modules/.bin 下去检索

☕ Demo-bad message

记不住怎么写 Commit Message?

- npm i commitizen —save-dev
  - 安装 giz-cz/cz (别名) 命令行工具
- npm i cz-conventional-changelog —save-dev
  - 安装语义化提交模版
- npm run commit

https://github.com/commitizen/cz-cli
https://github.com/commitizen/cz-conventional-changelog

Demo-commitizen

git push

git push

.huskyrc.json > ...

```
1 {
2   "hooks": {
3     "pre-commit": "npm run precommit",
4     "commit-msg": "commitlint -E HUSKY_GIT_PARAMS",
5     "pre-push": "npm run test:prod && npm run build"
6   }
7 }
```

第 5 行: push 之间运行 unit test + build

Unit Test

```
30    "badge-text": "lcov-badge coverage/lcov.info",
31    "test:prod": "npm run lint && npm run test -- --coverage --no-cache && npm run badge-text",
```

第 31 行:

- **手工 lint 一下**
- **然后测试生成报告**
- **在 console 输出覆盖率文本，用于 GitLab Badge 展示**

Demo-Unit Test

jest.config.js

jest.config.js > <unknown> > coverageThreshold > global

```
1  module.exports = {
2    transform: {
3      '.(ts|tsx)': 'ts-jest',
4    },
5    testRegex: '(/__tests__/.*|\\.(test|spec))\\.(ts|tsx|js)$',
6    moduleFileExtensions: ['ts', 'tsx', 'js'],
7    coveragePathIgnorePatterns: ['/node_modules/', '/test/', '/tools/'],
8    // jest-junit is used to generate unit test reports in junit format,
9    // which GitLab can store in Pipeline history.
10   reporters: ['default', 'jest-junit'],
11   coverageThreshold: {
12     global: {
13       branches: 90,
14       functions: 95,
15       lines: 95,
16       statements: 95,
17     },
18   },
19   collectCoverage: true,
20 }
```

第 11 行: 各种门槛值设置

第 19 行: 是否收集报告

create-gl-repo

create-gl-repo

本项目是 **typescript-library-starter-gl** 的配套项目。 当你使用 **typescript-library-starter-gl** 在自己的电脑上创建了新项目之后，可以用 `create-gl-repo` 在 **GitLab** 上迅速创建好新的 **project**，同时在本地项目中完成和 **project url** 相关的各种配置。

用法

```
npm i @li-mesh/create-gl-repo -g
cd YOUR_TYPESCRIPT-LIBRARY
create-gl-repo
```

`create-gl-repo` 会进行如下操作:

- 将当前 **project url** 写入 `package.json` 的 **repository** 字段；在 `package.json` 的 `publishConfig.registry` 字段中填入当前 **project** 对应的 **registry** 地址。
- 添加 **git origin** 地址。
- 为 `semantic-release` 工具创建好了 `GL_TOKEN`。目前的版本不再创建 `GL_URL` 和 `NPM_TOKEN`，这两个变量将由项目中的 `.gitlab-ci.yml` 创建。
- 为当前 **project** 配置了 `build_coverage_regex` 正则表达式，方便 **GitLab CI/CD** 从 **console** 输出中识别出 **Code Coverage Badge Text**。

`create-gl-repo` 从 `package.json` 、环境变量 `GL_TOKEN` 和 `GL_URL` 中获取需要的信息。

https://gitlab.chehejia.com/li-mesh/create-gl-repo

Demo-git push

git add package.json
git push --set-upstream origin master

.gitlab-ci.yml

- 完成 单测，构建，[release] 三个步骤
  - [release] 仅仅针对 master 分支
- [release] 意味着：
  - 根据当前的 tag 和 commit log 计算版本号
  - 为本次提交打上计算出的版本标签
  - 发布到对应的 npm registry
    - 我们使用 GitLab 内置的

https://semantic-release.gitbook.io/semantic-release/

```
> semantic-release
[7:37:21 AM] [semantic-release] › ℹ  Running semantic-release version 19.0.3
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "verifyConditions" from "@semantic-release/npm"
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "analyzeCommits" from "@semantic-release/commit-analyzer"
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "generateNotes" from "@semantic-release/release-notes-generator"
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "prepare" from "@semantic-release/npm"
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "publish" from "@semantic-release/npm"
[7:37:21 AM] [semantic-release] › ✔  Loaded plugin "addChannel" from "@semantic-release/npm"
[7:37:22 AM] [semantic-release] › ✔  Run automated release from branch master on repository git@gitlab.chehejia.com:li-mesh/ci-test10.git
[7:37:23 AM] [semantic-release] › ✔  Allowed to push to the Git repository
[7:37:23 AM] [semantic-release] › ℹ  Start step "verifyConditions" of plugin "@semantic-release/npm"
[7:37:23 AM] [semantic-release] [@semantic-release/npm] › ℹ  Verify authentication for registry https://gitlab.chehejia.com/api/v4/projects/7368/packages/npm/
[7:37:23 AM] [semantic-release] [@semantic-release/npm] › ℹ  Reading npm config from /builds/li-mesh/ci-test10/.npmrc
[7:37:23 AM] [semantic-release] [@semantic-release/npm] › ℹ  Wrote NPM_TOKEN to /tmp/5dd72efdb3b1336cd12ec49f85669887/.npmrc
[7:37:23 AM] [semantic-release] › ✔  Completed step "verifyConditions" of plugin "@semantic-release/npm"
[7:37:23 AM] [semantic-release] › ℹ  No git tag version found on branch master
[7:37:23 AM] [semantic-release] › ℹ  No previous release found, retrieving all commits
[7:37:23 AM] [semantic-release] › ℹ  Found 3 commits since last release
[7:37:23 AM] [semantic-release] › ℹ  Start step "analyzeCommits" of plugin "@semantic-release/commit-analyzer"
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  Analyzing commit: docs: 在 package.json 中增加 repo. url
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  The commit should not trigger a release
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  Analyzing commit: docs: demo for seminar 1
I can talk as long as you want
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  The commit should not trigger a release
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  Analyzing commit: feat: initial commit
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  The release type for the commit is minor
[7:37:23 AM] [semantic-release] [@semantic-release/commit-analyzer] › ℹ  Analysis of 3 commits complete: minor release
[7:37:23 AM] [semantic-release] › ✔  Completed step "analyzeCommits" of plugin "@semantic-release/commit-analyzer"
[7:37:23 AM] [semantic-release] › ℹ  There is no previous release, the next release version is 1.0.0
[7:37:23 AM] [semantic-release] › ℹ  Start step "generateNotes" of plugin "@semantic-release/release-notes-generator"
[7:37:23 AM] [semantic-release] › ✔  Completed step "generateNotes" of plugin "@semantic-release/release-notes-generator"
[7:37:23 AM] [semantic-release] › ℹ  Start step "prepare" of plugin "@semantic-release/npm"
[7:37:23 AM] [semantic-release] [@semantic-release/npm] › ℹ  Write version 1.0.0 to package.json in /builds/li-mesh/ci-test10
v1.0.0
[7:37:23 AM] [semantic-release] › ✔  Completed step "prepare" of plugin "@semantic-release/npm"
[7:37:25 AM] [semantic-release] › ✔  Created tag v1.0.0
[7:37:25 AM] [semantic-release] › ℹ  Start step "publish" of plugin "@semantic-release/npm"
[7:37:25 AM] [semantic-release] [@semantic-release/npm] › ℹ  Publishing version 1.0.0 to npm registry on dist-tag latest
```

缺省的 Plugins

```
"@semantic-release/commit-analyzer"
"@semantic-release/github"
"@semantic-release/npm"
"@semantic-release/release-notes-generator"
```

每个 Plugin 会关心一个或多个“处理阶段”

- execute the `verifyConditions` implementation of `@semantic-release/npm` then `@semantic-release/git`
- execute the `analyzeCommits` implementation of `@semantic-release/commit-analyzer`
- execute the `prepare` implementation of `@semantic-release/npm` then `@semantic-release/git`
- execute the `generateNotes` implementation of `@semantic-release/release-notes-generator`
- execute the `publish` implementation of `@semantic-release/npm`

每个 Plugin 可以有自己的附加配置信息

```json
{
  "plugins": [
    ["@semantic-release/commit-analyzer", {
      "preset": "angular",
      "releaseRules": [
        {"type": "docs", "scope":"README", "release": "patch"},
        {"type": "refactor", "release": "patch"},
        {"type": "style", "release": "patch"}
      ],
```

对于既有Repo如何补 tag?

确保之前有有效的 git tag version (e.g. v2.1.1)

https://semantic-release.gitbook.io/semantic-release/usage/configuration#existing-version-tags

```
# Make sure the commit 1234567 is in the release branch history
$ git branch --contains 1234567

# If the commit is not in the branch history it means that either:
# - you use a different branch than the one your release from before
# - or the commit sha has been rewritten (with git rebase)
# In both cases you need to configure your repository to have the last r

# List the tags for the commit 1234567
$ git tag --contains 1234567

# If v1.1.0 is not in the list you add it with
$ git tag v1.1.0 1234567
$ git push origin v1.1.0
```

总结

- 准确的 change log 来自每一次提交
  - 一个提交解决一个问题 (除了第一次)
- 遵守规范，工具很重要
- commit hook 关注静态质量
- push hook 关注动态质量
- .gitlab-ci.yaml 关注发布质量