npmで成果物をsemantic-release 🚀

目的: npmで単独のJSONファイルを配布したい

例) 辞書ファイル、更新頻度が高いファイル

最終的にやりたいこと

npm installして

```
npm install technical-word-rules
```

requireするだけで辞書ファイルをJavaScriptから使いたい

```
var json = require("technical-word-rules");
// 辞書ファイルを使えるようにしたい
```

方法1: install hook

npmのpacakage.jsonにはinstall時に行う挙動を書くことが出来る

```
  "scripts": {
    "install": "node install-hook.js"
  },
```

npm install technical-word-rulesするとinstall-hook.jsが実行出来る。

install-hook.js

```
// ファイルをダウンロードして最新のものへと置き換える
var fs = require("fs");
var http = require("http");
http.get("http://azu.github.io/technical-word-rules/all.json", function (res) {
    var body = '';
    res.setEncoding('utf8');

    res.on('data', function (chunk) {
        body += chunk;
    });
    res.on('end', function (res) {
        fs.writeFileSync(require("path").join(__dirname, "all.json"), body, "utf-8");
    });
}).on('error', function (e) {
    console.log("Got error: " + e.message);
});
```

問題: install-hook

- 無理矢理感がある
- npm側のバージョンは上がらないので、リソースのバージョン管理ができてない
- HTTPで最新のリソースがダウンロード出来るようにしておく必要がある
  - 先ほどの例だとコミット毎にgh-pagesにpushしてる...

方法2: semantic-release

コミット毎にnpmにリリースすればいい 🚀

semantic-release

- コミットメッセージから自動的にsemverでバージョンをあげてnpmにpublishする
- => Travis CIからテストがパスするごとにnpm publishされる
- Gitで管理するpacakge.jsonからはversionを取り除く
  - semantic-releaseが"前回のバージョン"としているのはnpmに挙げられているバージョン

semantic-releaseのワークフロー

- ローカル: git commit -> git push
- Travis CI:
  - semantic-release pre(package.jsonにversionを書き込み)
  - npm publish
  - semantic-release post(GitHub ReleaseにChangeLogの書き込み)

どうやってバージョンを決める?

- 元となるバージョンはnpmから取得
- コミットメッセージを見て`feat`があるなら`minor`、`fix`なら`patch`と解析してsermverで自動的にあげる
  - デフォルトはGit Commit Message Conventions - Google ドキュメントに従ったルール
  - 自分で定義出来る : analyzeCommits

関連

- ajoslin/conventional-changelog
- 良いChangeLog、良くないChangeLog | Web Scratch

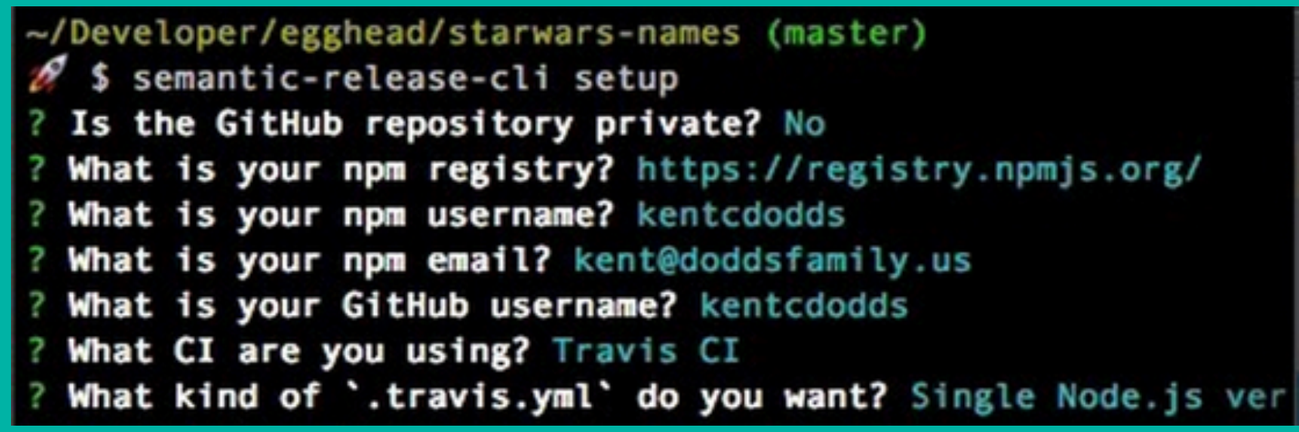
導入が簡単

```
$ npm install -g semantic-release-cli
$ semantic-release-cli setup
```

後は対話に応えるだけで、Travis CI、

npm、GitHubの連携をやってくれる。

- How to Write a JavaScript Library - Automating Releases with semantic-release - JavaScript Video Tutorial #free @eggheadio



実践

- azu/technical-word-rules
- コミット毎にnpmに自動的にリリースされる
- Releases · azu/technical-word-rules
- GitHub Releseに自動的にChangelogも追加される

全自動 or 半自動

- 全自動でやるにはsemantic-releaseは便利
- 辞書みたいな更新頻度が高いものを手動リリースしなくていい
- セミオートでやりたい場合はconventional-changelogを使うと同様の事ができる
- ajoslin/conventional-changelog
- stevemao/conventional-github-releaser

まとめ

- semantic-releaseを使うことでリソースをnpmで配りたい場合もsemverで管理できるようになる
- GitHub Releaseのリリースノートも自動化できる(APIを使ってる)
- GitHub Releaseに自動生成CHANGELOG + コメントを編集で加えるパターンだと、比較的簡単にリリースノートを書ける