

Improve project itsability. [있어빌리티]

당장 내 프로젝트에 적용할 수 있는 서비스들.

기술적인 내용 X








있어빌리티 높이기

=프로젝트/코드 퀄리티 높이기

README.md

main 1 branch 0 tags

Go to file Add file Code

	stevejkang Add acceptance test	8b3317f 4 days ago	5 commits
	gradle/wrapper	Initialize spring	4 days ago
	src	Add acceptance test	4 days ago
	.gitignore	Initialize spring	4 days ago
	build.gradle	Update project dependency	4 days ago
	gradlew	Initialize spring	4 days ago
	gradlew.bat	Initialize spring	4 days ago
	settings.gradle	Initialize spring	4 days ago

Help people interested in this repository understand your project by adding a README.

Add a README

About

A simple HanaFN-AES256 encryption/decryption API server. (credential removed)

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages



Help people interested in this repository understand your project by adding a README.

README에 들어가면 좋을 내용

- 프로젝트 이름
- 프로젝트 설명
- 설치 가이드 (Installation Steps)
- 기능들 (Features)
- 기여하고자 하는 사람들을 위한 가이드 (Contribution Guide)
- 프로젝트가 사용중인 라이선스
- 변경사항 (ChangeLog)
- FAQ

README에 들어가면 좋을 내용 (...)

- Demonstration Video / GIF ([stevejkang/git-hook-bs3-prevention](https://github.com/stevejkang/git-hook-bs3-prevention))
- Executable Attachments

README.md - Badges

3. 왠지 모를 신뢰감을 주는 배지들

- Travis CI, Circle CI, 또는 GitHub Action에서 제공하는 빌드 스테이터스 배지를 사용해 프로젝트가 정상적으로 돌아가고 있다는 안도감을 주자.
- 코드 커버리지, 코드 퀄리티, maintainability 배지를 사용해 프로젝트의 코드가 훌륭하고 안전하다는 인상을 주자.
- npm 또는 pypi 등지에서 당신의 패키지가 얼마나 다운로드 되었는지를 표시하는 스탯이 어느정도 자랑할만 하다면 그 배지도 추가하자.
- 그 밖에 shields.io에서 원하는 배지를 찾아 붙여 보도록 하자.
- 다만 주의할 점은 과하지 않도록 해야 한다.

어필과 자량의 수단...(?)

Licence MIT Windows passing Ubuntu passing macOS passing Ubuntu - Codecov passing build passing

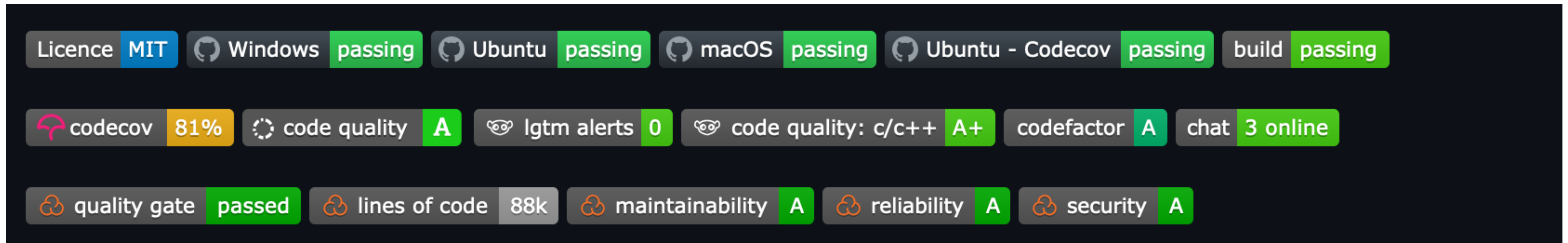
codecov 81% code quality A lgtm alerts 0 code quality: c/c++ A+ codefactor A chat 3 online

quality gate passed lines of code 88k maintainability A reliability A security A

TypeScript-JSON

Runtime type checker, and 5x faster `JSON.stringify()` function, with only one line.

license MIT npm v3.0.5 downloads 3.3k/month build passing wiki documentation



MIT 라이선스가 적용된 오픈소스 프로젝트.
Windows, Ubuntu, macOS에서 잘 돌아간다.

제대로 빌드가 가능한 상태.

81%의 코드가 테스트됨.

A급(?) 코드퀄리티.

8만 8천줄짜리 코드.

Maintainability와 reliability, security가 어느정도 검증된 프로젝트.

한달에 3.3k 다운로드 되는 프로젝트.
성공적으로 빌드되는 패키지.
문서를 보려면 버튼을 클릭해라.

TypeScript-JSON

Runtime type checker, and 5x faster `JSON.stringify()` function, with only one line.

license MIT

npm v3.0.5

downloads 3.3k/month

 build passing

wiki documentation

<https://shields.io>

다양한 배지들을 모아둔 사이트

[Build](#)

[Code Coverage](#)

[Test Results](#)

[Analysis](#)

[Chat](#)

[Dependencies](#)

[Size](#)

[Downloads](#)

[Funding](#)

[Issue Tracking](#)

[License](#)

[Rating](#)

[Social](#)

[Version](#)

[Platform & Version Support](#)

[Monitoring](#)

[Activity](#)

[Other](#)

Continuous Integration(CI)
Continuous Deployment(CD)

CI의 역할

- Continuous Integration = 지속적 통합
- SW Life Cycle에서 Build & Test & Merge(Integration) 를 담당
- 협업과정에서 소스코드가 안정적으로 다른 코드들과 합쳐질 수 있도록 하는 역할
- (만약 적용한다면) Branch Merging이나 Pull Request가 올라갔을 때

CD의 역할

- Continuous Delivery라고도 하고, Continuous Deployment라고도 함.
- SW Life Cycle에서 Release & Deployment를 담당.
- Production(= User side)까지 소스코드를 전달.
 - Delivery와 Deployment의 차이는 Manual/Automatic 차이.
 - Delivery 예시: rsync로 sync 맞추고, 실제 구동은 서버 접속하여 build & run

CI/CD Tools

- Jenkins
- Circle CI
- Travis CI
- Etc..

LICENSE

License 종류

라이선스 별 특징 정리

License	필수 요구사항	가능한 활동	소스코드 공개의무	제약 조건	부가 설명	원문
GNU GPL v2.0/v3.0	- 수정한 소스코드 또는 GPL 소스코드를 활용한 소프트웨어 모두 GPL로 공개 - 라이선스 및 저작권 명시 - 변경사항 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용	있음	높음	자유소프트웨어 재단에서 제정. GPL라이선스를 이용하여 개발 시 개인적, 내부적 이용에 한해서는 소스코드를 공개하지 않아도 되나, 외부 배포 시 해당 소프트웨어의 전체 소스코드를 공개 해야 함. (3.0버전은 아파치 라이선스와 같이 사용 가능) <i>ex) 파이어폭스(2.0), 리눅스 커널, 깃, 마리아DB 등</i>	http://www.gnu.org/copyleft/gpl.html (원어) http://www.oss.kr/oss_license/69523 (번역)
LGPL	- LGPL 소스코드를 단순 라이브러리 이용 이외의 목적으로 사용시 소스코드 공개 - 라이선스 및 저작권 명시 - 변경사항 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용	있음	중간	기존 GPL의 높은 제약을 완화시키기 위해 탄생. LGPL로 작성된 소스코드를 라이브러리(정적, 동적)로만 사용하는 경우엔 소스코드를 공개하지 않아도 됨 . 그 이외 사항은 GPL과 동일. <i>ex) 파이어폭스(2.1)</i>	https://www.gnu.org/licenses/lgpl-3.0.en.html
BSD	- 라이선스 및 저작권 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용	없음	낮음	버클리 캘리포니아 대학에서 제정. BSD 자체가 공공공기관에 만든 것이므로 공공환원의 의도가 강해서 저작권 및 라이선스 명시 이외엔 아무 제약이 없이 사용 가능 한 자유로운 라이선스 <i>ex) OpenCV</i>	https://opensource.org/licenses/BSD-3-Clause
Apache	- 라이선스 및 저작권 명시 - 변경사항 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용 - 2차 라이선스 가능	없음	낮음	아파치 소프트웨어 재단에서 제정. 소스코드 공개 의무 없음. 단, 아파치 라이선스 사용을 밝혀야 함. BSD보다 좀더 완화 된 내용. <i>ex) 안드로이드, 하둡 등</i>	https://opensource.org/licenses/Apache-2.0
MIT	- 라이선스 및 저작권 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용 - 2차 라이선스 가능	없음	낮음	BSD 라이선스를 기초로 MIT 대학에서 제정. MIT 라이선스를 따르는 소프트웨어 사용하여 개발 시, 만든 개발품을 꼭 오픈소스로 해야 할 필요는 없음. 물론 소스코드 공개 의무도 없음 . <i>ex) X 윈도 시스템</i>	https://opensource.org/licenses/MIT
MPL	- 수정한 소스코드 MPL 라이선스로 공개 (단순 활용 시 공개 의무 없음) - 라이선스 및 저작권 명시 - 특허기술이 구현된 경우 관련 사실을 LEGAL이란 파일에 기록하여 배포	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용 - 2차 라이선스 가능	가변적	중간	1.0 버전은 넷스케이프 변호사였던 미첼 베이커가 작성, 1.1과 2.0버전은 모질라 재단에서 제정. 소스코드와 실행파일의 저작권 분리 가 특징. MPL라이선스의 소스코드를 사용하여 개발했을 시, 수정한 소스코드는 MPL로 공개하고 원저작자에게 수정한 부분에 대해 알려 야 하지만, 실행파일은 독점 라이선스로 배포 가능 . 또한 MPL와 무관하게 작성된 소스코드는 공개할 필요 없음 . <i>ex) 파이어폭스(1.1)</i>	https://opensource.org/licenses/MPL-2.0
Eclipse	- 수정한 소스코드를 Eclipse 라이선스로 공개(단순 활용 시 공개 의무 없음) - 라이선스 및 저작권 명시	- 상업적 이용 - 배포, 수정 가능 - 특허신청, 사적이용 - 2차 라이선스 가능	가변적	중간	이클립스 재단에서 제정. CPL을 대체하며, GPL보다 약한 수준으로 기업 친화적 인 특징. <i>ex) Eclipse</i>	https://opensource.org/licenses/EPL-1.0

Test Coverage

Test Coverage?

- 얼마나 테스트가 충분한가
- 수행한 테스트가 테스트의 대상(우리의 코드)을 얼마나 커버했는지를 측정하는 지표
- (중요) 100%가 됐다고 해서 완벽한 코드도 아니고
- (중요) 100%를 목표로 개발할 수도 없다.
 - Db connection, external api 등 테스트 못하는 것은 분명히 존재한다.
 - 먹등하지 못한 테스트는 하지 않는다.
 - 차라리 그걸 고민하는 시간에 다른 테스트 코드를 짜는 게 낫다.

Coverage Check Tools

- CodeCov
- Codacy
- (사실 더 많을 거예요... 제가 모를 뿐...)

Security

added 13 packages, removed 13 packages, changed 3 packages, and audited 2173 packages in 34s

12 packages are looking for funding

run `npm fund` for details

147 vulnerabilities (4 **low**, 50 **moderate**, 79 **high**, 14 **critical**)

To address issues that do not require attention, run:

npm audit fix

To address all issues possible (including breaking changes), run:

npm audit fix --force

Some issues need review, and may require choosing a different dependency.

Run `npm audit` for details.

- CVE <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-24434>

Static Code Analysis

Static Code Analysis

- 정적코드분석: Static하게(!= runtime, compile) + 코드를 실행시키지 않고.
- 찾고자 하는 정보: 보안취약점, 잠재적 오류
- JS 환경의 대표적인 예시: eslint

- Coding Standard
- Unused Variable/Methods, Un-reachable code, Potential performance issue (memory leak), Duplicated blocks

CodeFactor

- 정적분석된 코드 Quality를 정량적으로 측정 & 개선가능한 버전으로 개선안 제안
- stevejkang/driver-license-verification#15
- <https://www.codefactor.io/repository/github/stevejkang/driver-license-verification/issues>

 **Unexpected any. Specify a different type.** lines of code = 1

Found in [src\shared\core\Result.ts:44](#)

```
44 public static combine(results: Result<any>[]): Result<any> {
```

 **Unexpected any. Specify a different type.** lines of code = 1

Found in [src\shared\core\Result.ts:44](#)

```
44 public static combine(results: Result<any>[]): Result<any> {
```

 **Unnecessary escape character: \-.** lines of code = 1

Found in [src\verification\domain\DriverBirthday.ts:21](#)

```
21 if (!value.match(/^d{4}\-(0[1-9]|1[012])\-(0[1-9]|[12][0-9]|3[01])$/g)) {
```

 **Unnecessary escape character: \-.** lines of code = 1

Found in [src\verification\domain\DriverBirthday.ts:21](#)

```
21 if (!value.match(/^d{4}\-(0[1-9]|1[012])\-(0[1-9]|[12][0-9]|3[01])$/g)) {
```

**만약 Open source 라면?
CONTRIBUTING.md**

[https://github.com/
denysdovhan/wtfjs/blob/
master/CONTRIBUTING.md](https://github.com/denysdovhan/wtfjs/blob/master/CONTRIBUTING.md)

CHANGELOG.md

conventional-changelog-cli

With `npm version`

Using the npm scripts to our advantage with the following hooks:

```
{
  "scripts": {
    "version": "conventional-changelog -p angular -i CHANGELOG.md -s && git add CHANGELOG.md"
  }
}
```

- 자동으로 git tag 기반으로 changelog.md 내용을 채워줌

Link

🔗 0.2.2 (2022-04-11)

- Add codeql badge to readme document ([b34ed90](#))
- Add required node version and installation steps ([68e2997](#))
- Bump follow-redirects from 1.14.7 to 1.14.9 ([481828a](#))
- Bump minimist from 1.2.5 to 1.2.6 ([c2eac87](#))
- Create codeql-analysis.yml ([55a0284](#))
- Fix dependency security issue ([94f1954](#))
- Increase jest test timeout value ([2af33df](#))
- fix: package.json & yarn.lock to reduce vulnerabilities ([938aaed](#))

0.2.1 (2022-02-02)

- Add env to travis and refactor api unit test code ([cab1fc7](#))
- Add more github action badge ([a43c2a6](#))
- Enhance api unit test coverage and add actions ([3043fa5](#)), closes [#9](#)
- Increase jest test timeout value ([f942e4a](#))

0.2.0 (2022-01-19)

- Support verification method without `serialNumber` ([07fc103](#))

Examples

[stevejkang/driver-license-verification](#)

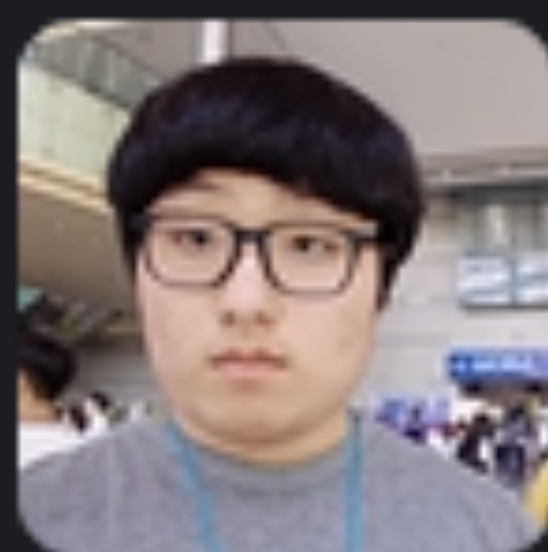
Questions...?



Joon 🐼 3 hours ago

- README 깔끔하게 꾸미는 팁 (구성, 배지, ...)
- 개인 프로젝트의 일정/태스크 관리는 어떻게 하는지

(edited)



Robert 2 hours ago

@Steve

- multi project 관리
- linting, testing 자동화



John 🐾

41 minutes ago

개인 프로젝트 시작시에 어느정도 기초 설정이 되어있는 템플릿 같은걸 사용하는지 궁금 (항상 바닥부터 만들어가는가)