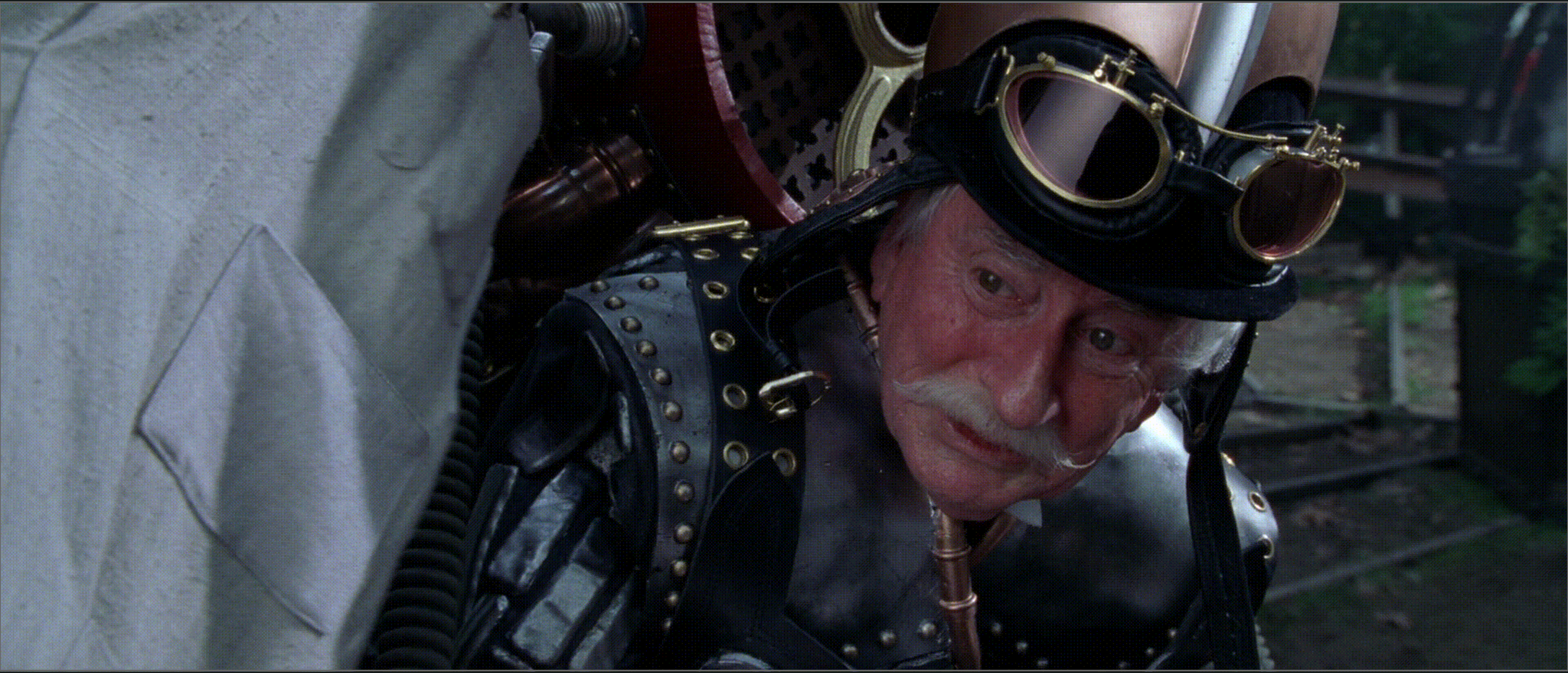
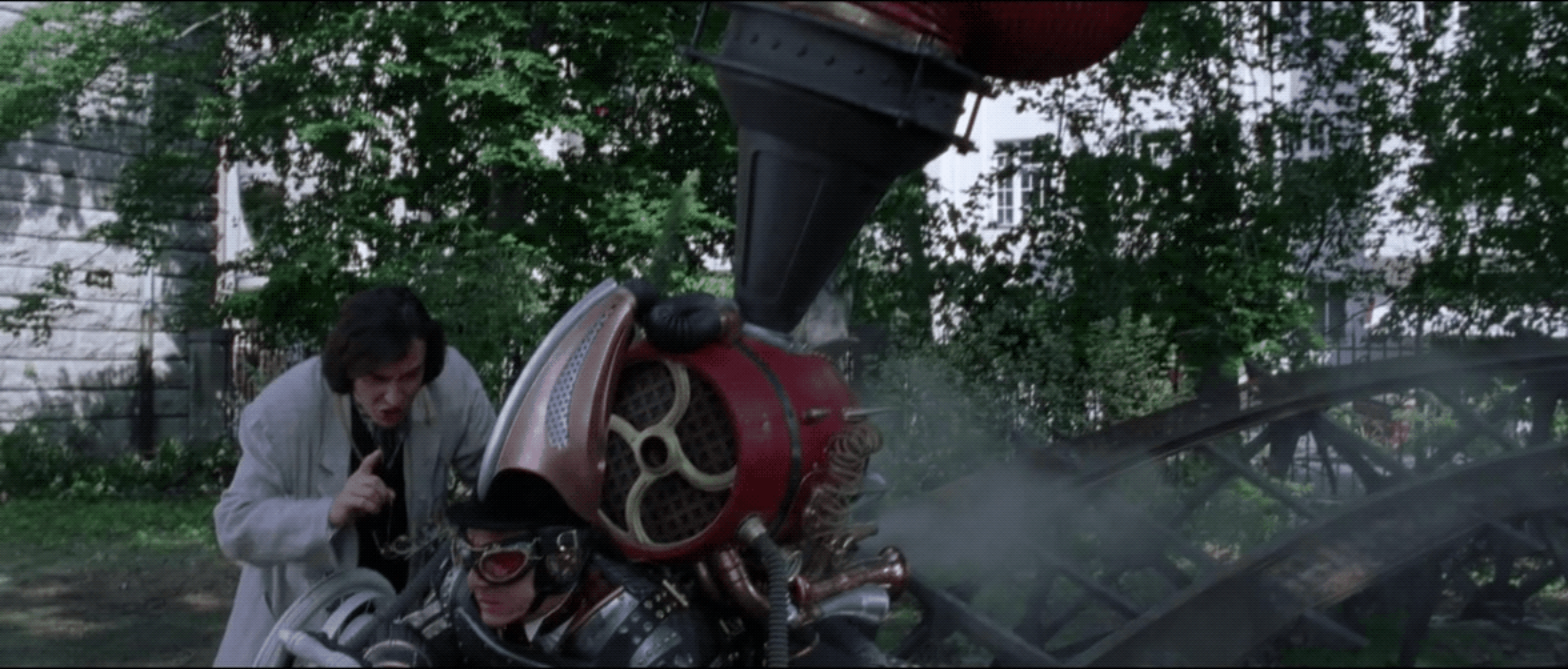
nest

# Tried To Shift In 80 Hours



M E A N



## Tools

## Tools

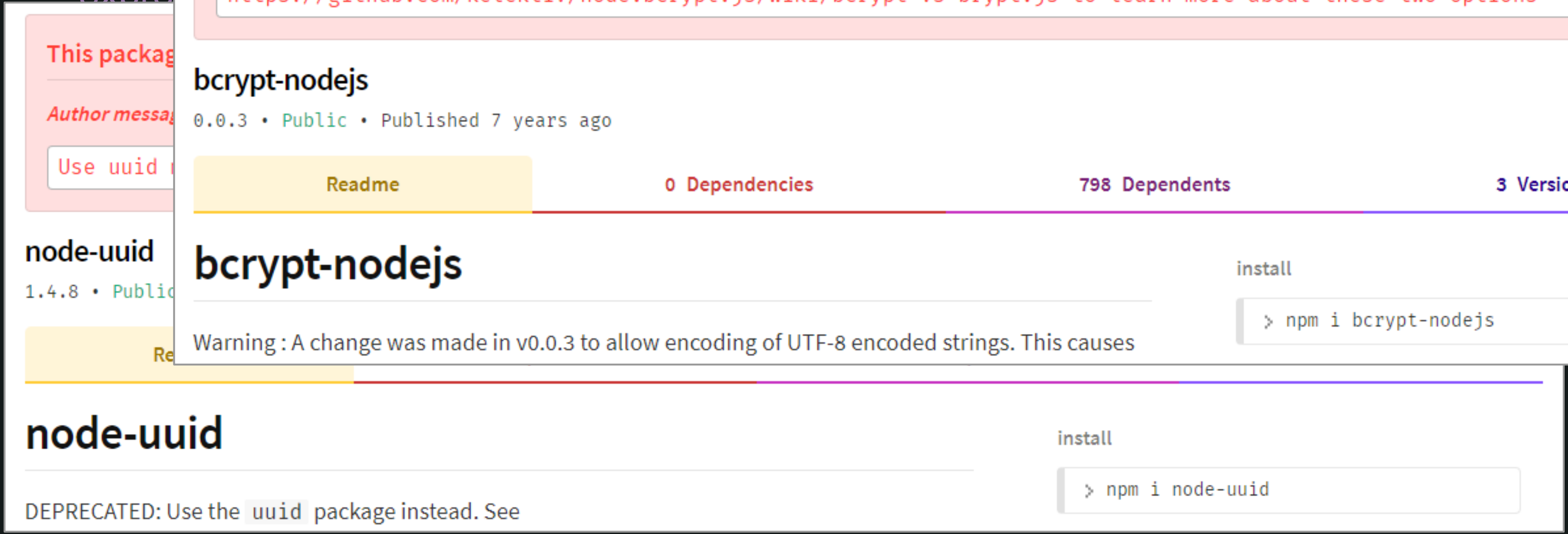
- npm packages

## package.json

```
"dependencies": {
  "bcrypt-nodejs": "0.0.3",
  "crowdin": "1.0.0",
  "express-tgz": "0.0.3",
  "gitignore-to-glob": "0.3.0",
  "migration": "0.3.0",
  "nconf": "0.8.4",
  "node-uuid": "1.4.8",
  "shorturl-2": "0.0.6",
  "unzip": "0.1.11"
}
```

# package.json

```
"depend
  "bcryp
  "crowd
  "expre
```



This packag

*Author messa*

Use uuid

node-uuid

1.4.8 • Public

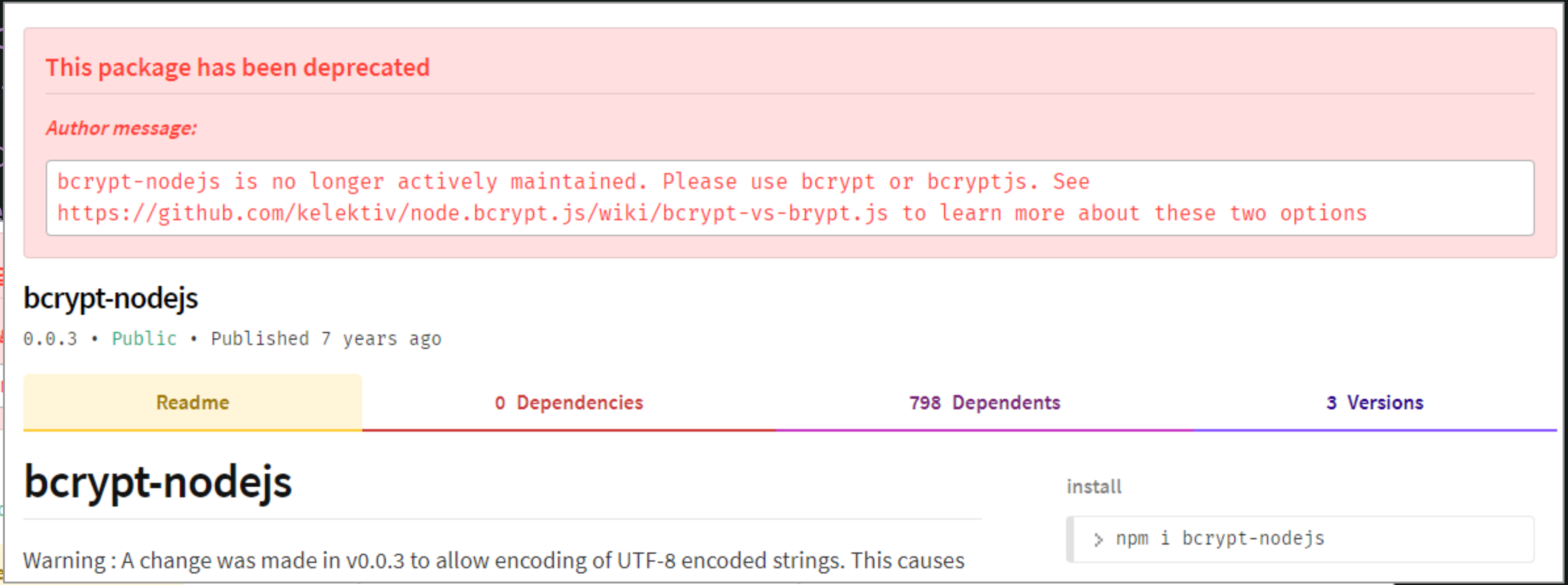
Re

node-uuid

install

```
> npm i node-uuid
```

DEPRECATED: Use the `uuid` package instead. See



**This package has been deprecated**

*Author message:*

```
bcrypt-nodejs is no longer actively maintained. Please use bcrypt or bcryptjs. See
https://github.com/kelektiv/node.bcrypt.js/wiki/bcrypt-vs-brypt.js to learn more about these two options
```

bcrypt-nodejs

0.0.3 • Public • Published 7 years ago

Readme | 0 Dependencies | 798 Dependents | 3 Versions

## bcrypt-nodejs

install

```
> npm i bcrypt-nodejs
```

Warning : A change was made in v0.0.3 to allow encoding of UTF-8 encoded strings. This causes

# package.json

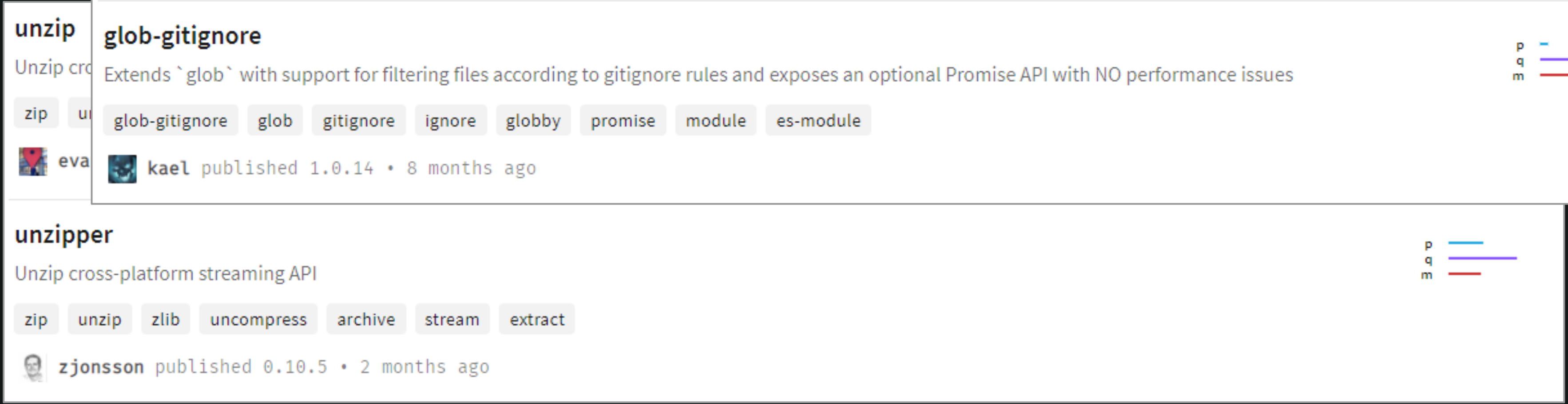
"dependencies": {

"bcrypt-nodejs": "0.0.3"

"c

"e

"g



unzip

Unzip cro

zip u

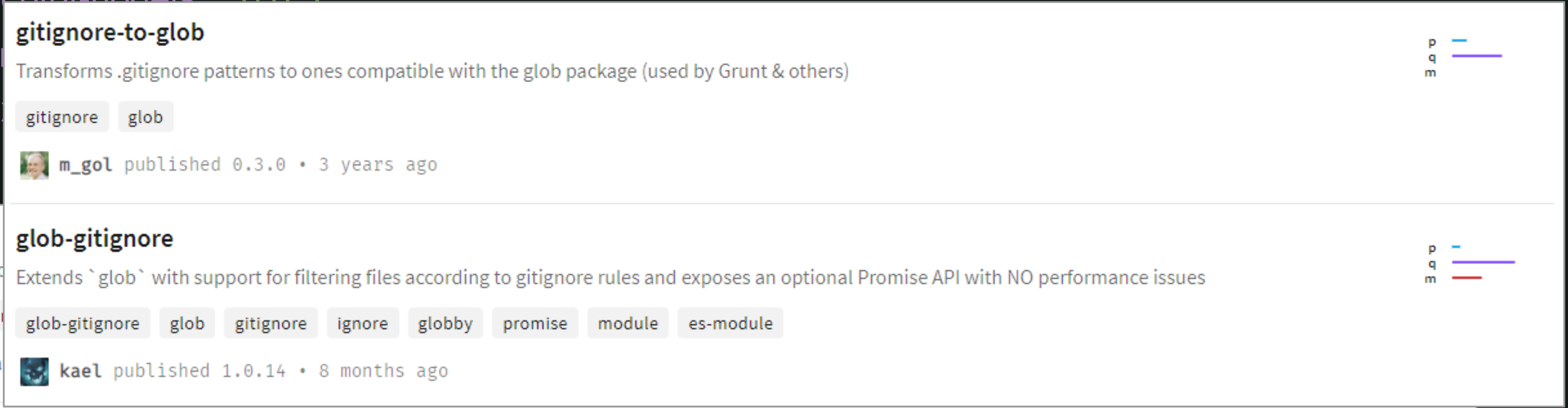
eva

unzipper

Unzip cross-platform streaming API

zip unzip zlib uncompress archive stream extract

zjonsson published 0.10.5 • 2 months ago



gitignore-to-glob

Transforms .gitignore patterns to ones compatible with the glob package (used by Grunt & others)

gitignore glob

m_gol published 0.3.0 • 3 years ago

glob-gitignore

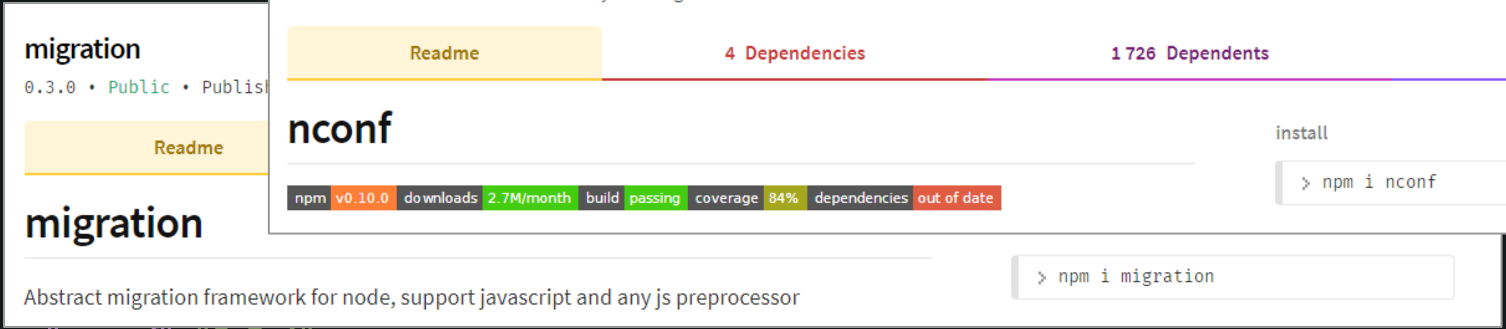
Extends `glob` with support for filtering files according to gitignore rules and exposes an optional Promise API with NO performance issues

glob-gitignore glob gitignore ignore globby promise module es-module

kael published 1.0.14 • 8 months ago

## package.json

"dependencie



migration

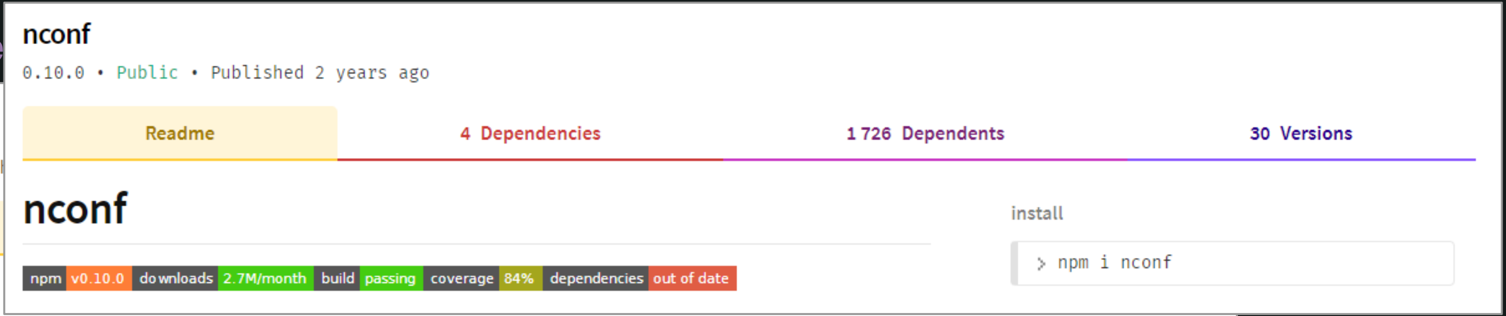
0.3.0 • Public • Publis

Readme

migration

Abstract migration framework for node, support javascript and any js preprocessor

> npm i migration



nconf

0.10.0 • Public • Published 2 years ago

Readme | 4 Dependencies | 1 726 Dependents | 30 Versions

nconf

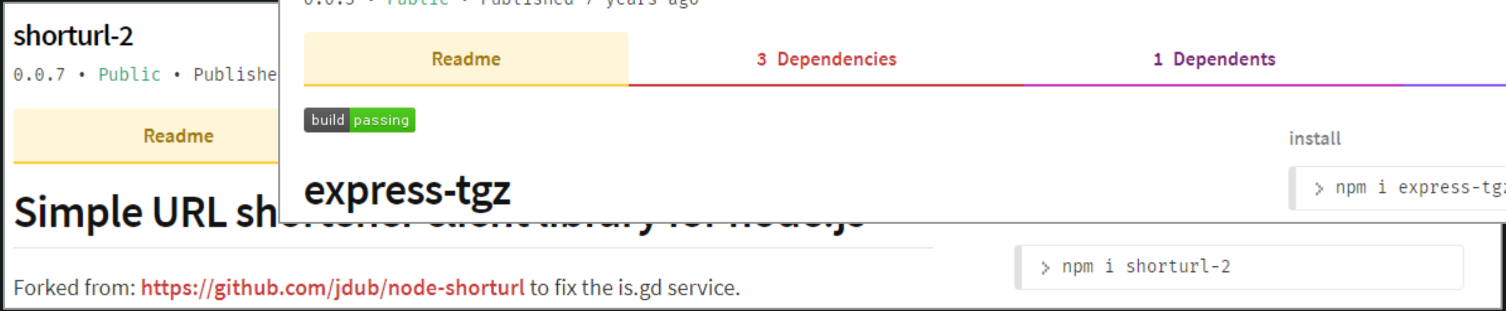
npm v0.10.0 | downloads 2.7M/month | build passing | coverage 84% | dependencies out of date

install

> npm i nconf

"nconf": "0.8.4",

"node-uuid":



shorturl-2

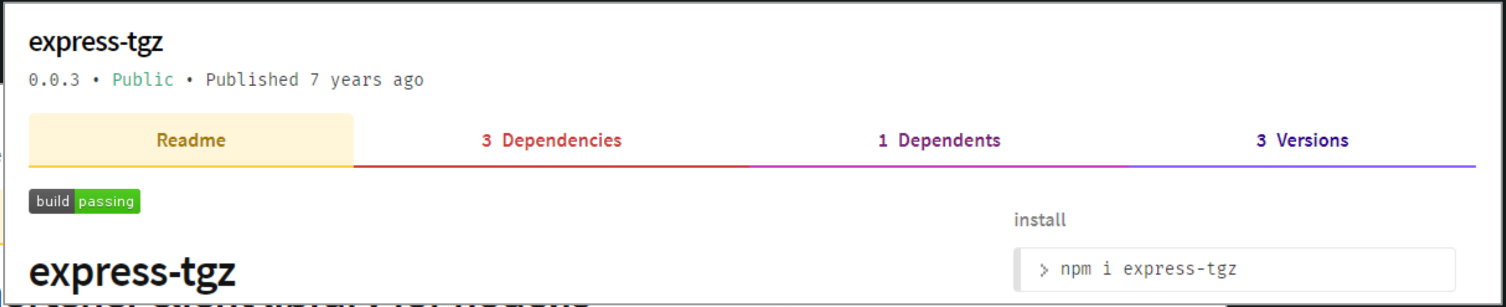
0.0.7 • Public • Publishe

Readme

Simple URL sh

Forked from: https://github.com/jdub/node-shorturl to fix the is.gd service.

> npm i shorturl-2



express-tgz

0.0.3 • Public • Published 7 years ago

Readme | 3 Dependencies | 1 Dependents | 3 Versions

build passing

express-tgz

install

> npm i express-tgz

## Tools

- npm packages
- express

## Express 1

```
app.use(session({
  secret: 'keyboard cat',
  proxy: true,
  resave: true,
  saveUninitialized: true
}));
```

## Express 2

```
module.exports = () => {

  return new Promise(function (resolve, reject) {
    async.parallel({
      a: Entity.findOne({name: 'Value A' }),
      b: Entity.findOne({name: 'Value B' }),
      c: Entity.findOne({name: 'Value C' })
    }, (error, result) =>
     error ? reject(error) : resolve(result)
    );
  });
};
```

## Express 3

```
module.exports = (conf) => {

  const S3_BUCKET = conf.S3_BUCKET;

  AWS.config.region = conf.S3_REGION;
  AWS.config.update({
    accessKeyId: conf.S3_ACCESS_KEY_ID,
    secretAccessKey: conf.S3_SECRET_ACCESS_KEY
  });

  const s3 = new AWS.S3();
};
```

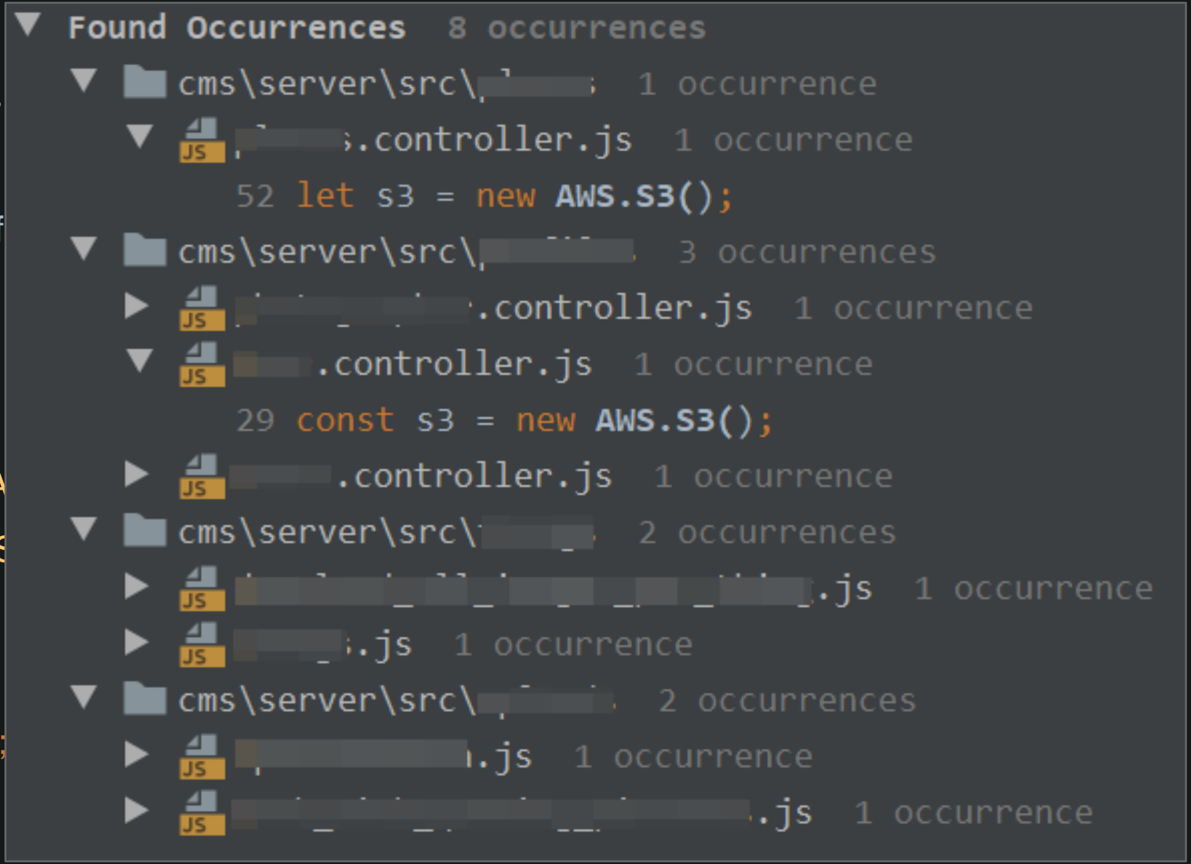
## Express 3

```
module.exports = (conf) =>

  const S3_BUCKET = conf

  AWS.config.region = conf.
  AWS.config.update({
    accessKeyId: conf.S3_A
    secretAccessKey: conf.S
  });

  const s3 = new AWS.S3();
};
```



```
▼ Found Occurrences  8 occurrences
  ▼ cms\server\src\[redacted]  1 occurrence
    ▼ [redacted].controller.js  1 occurrence
        52 let s3 = new AWS.S3();
  ▼ cms\server\src\[redacted]  3 occurrences
    ▶ [redacted].controller.js  1 occurrence
    ▼ [redacted].controller.js  1 occurrence
        29 const s3 = new AWS.S3();
    ▶ [redacted].controller.js  1 occurrence
  ▼ cms\server\src\[redacted]  2 occurrences
    ▶ [redacted].js  1 occurrence
    ▶ [redacted].js  1 occurrence
  ▼ cms\server\src\[redacted]  2 occurrences
    ▶ [redacted].js  1 occurrence
    ▶ [redacted].js  1 occurrence
```

## Tools

- npm packages
- express
- database

## Database 1: Issues

```
XXXX-XX-XXTXX:XX:XX.XXX+0000 I NETWORK  [thread1] connection
accepted from xxx.xxx.xxx.xxx:41681 #27019 (1611 connections now open)
XXXX-XX-XXTXX:XX:XX.XXX+0000 F - [statsSnapshot] out of memory.
```

## Database 2: No Structure

```
[{
			"show": "true",
			"isTrash": false
}, {
			"show": false,
			"hidden": "hide"
}, {
			"hidden": true
}, {
			"list": "white",
			"hidden": "show"
}]
```

## Database 3: Legacy props

```
const LEGACY_VALID_SHOW_VALUE = 'show';
const LEGACY_VALID_HIDDEN_VALUE = 'false';

const shouldBeShown =
    thingHidden === LEGACY_VALID_SHOW_VALUE
    || thingHidden !== LEGACY_VALID_HIDDEN_VALUE;
```

## What did this go to lead to?

## High-level Defects

- Unpredictable break down

## High-level Defects

- Unpredictable break down
- Available free-paid content

## High-level Defects

- Unpredictable break down
- Available free-paid content
- Loss of business

## High-level Defects

- Unpredictable break down
- Available free-paid content
- Loss of business
- Hydra Bug

## A Vital Question

## Solutions

## Solutions

- add new middleware

## Solutions

- add new middleware
- make data refactoring

## Solutions

- add new middleware
- make data refactoring
- increase cyclomatic complexity

## Solutions

- add new middleware
- make data refactoring
- increase cyclomatic complexity
- call for psychics || resignation notice

## Wrapper?

Nest is a platform-agnostic framework.

[...]

For example, most components can be re-used without change across different underlying HTTP server frameworks (e.g., Express and Fastify).

## Express

- custom modules & architecture
- too many efforts for reorganizing
- writing tests too complex
- potential vulnerabilities

## Nest

- ✓ tuned up & re-usable logic
- ✓ DI
- ✓ AOP (e.g. Interceptors)
- ✓ Angular

Valor
SOFTWARE

❤

nest

I'll take
your
wager!

## Worthwhile cause

## Worthwhile cause

- accident prevention and response
- discovery hidden defects
- detect legacy data

## Worthwhile cause

- accident prevention and response
- discovery hidden defects
- detect legacy data
- minimal changes

80:00

## Steps

- Quantify the amount of work

## Repository

- Monorepo

- Monorepo
- 3 subproject

- Monorepo
- CMS javascript node v8.12.0

- Monorepo
- CMS javascript node v8.12.0
- CRON javascript node v8.12.0

- Monorepo
- CMS javascript node v8.12.0
- CRON javascript node v8.12.0
- API typescript node v2.7.2

- Monorepo
- CMS javascript node v8.12.0
- CRON javascript node v8.12.0
- API typescript node v2.7.2
- 3 common modules

- Monorepo
- CMS javascript node v8.12.0
- CRON javascript node v8.12.0
- API typescript node v2.7.2
- 3 common modules
- 0 Business Logic Layers
- e2e tests

## jscpd

## API

162 files

11 111 lines of code

62 Clones found

675 Duplicated Lines

6.08% Copy-Paste

## CMS

80 files

6 420 lines of code

42 Clones found

578 Duplicated Lines

9.07% Copy-Paste

## prettier

## Steps

- ✓ Quantify the amount of work

78:00

## Steps

- ✓ Quantify the amount of work
- ● Create Nestjs App

## Add core packages

➔ npm i \
  @nestjs/common

## Add core packages

➔ npm i \

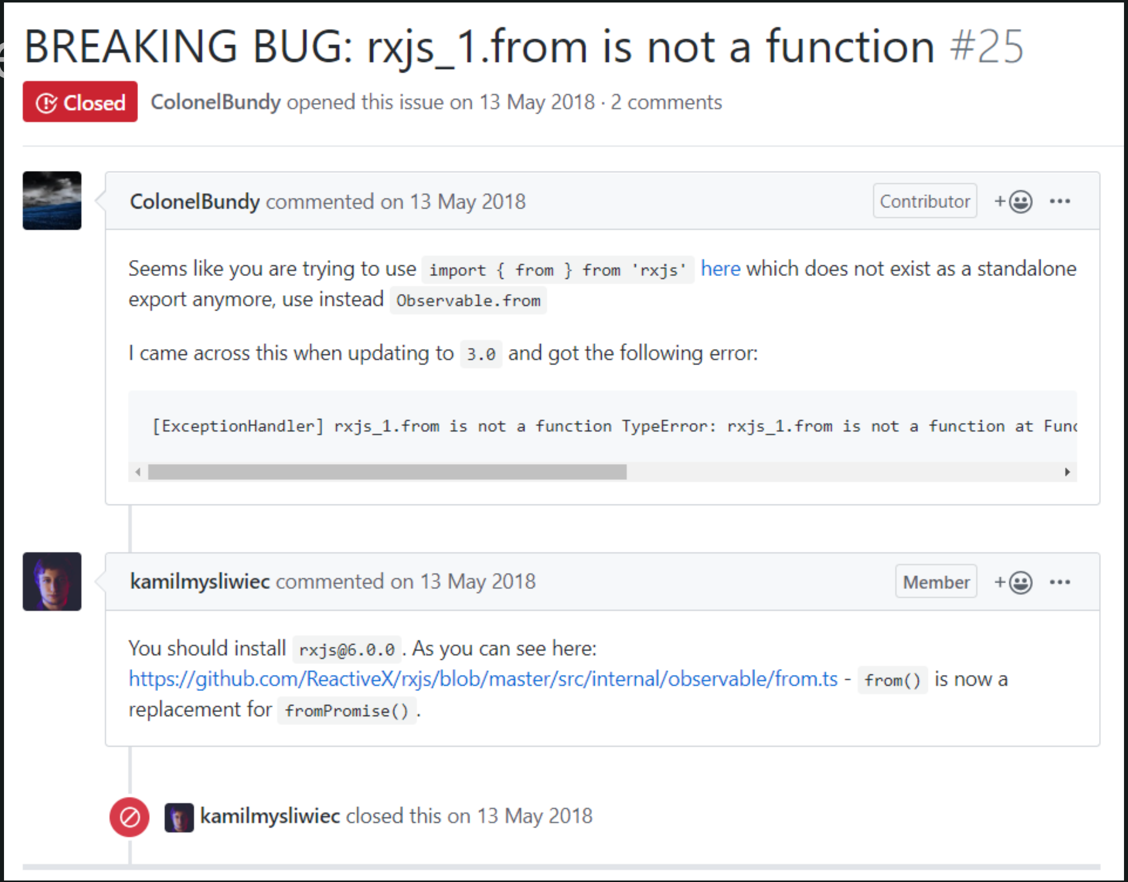
```
@nestjs/common \
@nestjs/core
```

## Add core packages

```
➔ npm i \
   @nestjs/common \
   @nestjs/core \
   rxjs@6.0.0
```

## Add core package

- ➔ npm i \
  @nestjs/common \
  @nestjs/core \
  rxjs@6.0.0



# BREAKING BUG: rxjs_1.from is not a function #25

Closed ColonelBundy opened this issue on 13 May 2018 · 2 comments

**ColonelBundy** commented on 13 May 2018 — Contributor

Seems like you are trying to use `import { from } from 'rxjs'` here which does not exist as a standalone export anymore, use instead `Observable.from`

I came across this when updating to `3.0` and got the following error:

```
[ExceptionHandler] rxjs_1.from is not a function TypeError: rxjs_1.from is not a function at Func
```

**kamilmysliwiec** commented on 13 May 2018 — Member

You should install `rxjs@6.0.0`. As you can see here: https://github.com/ReactiveX/rxjs/blob/master/src/internal/observable/from.ts - `from()` is now a replacement for `fromPromise()`.

**kamilmysliwiec** closed this on 13 May 2018

## Add core packages

```
➔ npm i \
   @nestjs/common \
   @nestjs/core \
   rxjs@6.0.0 \
   typescript
```

## Add core packages

➔ npm i \

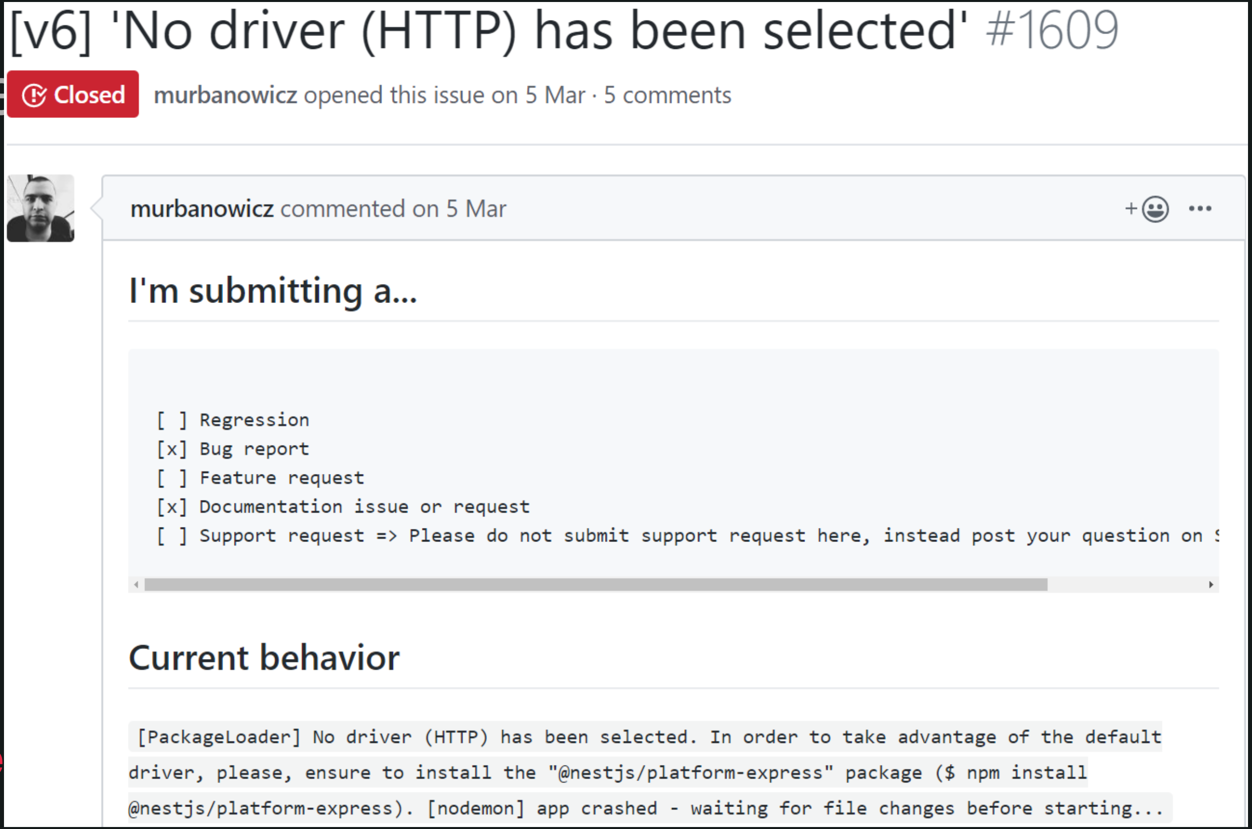
```
  @nestjs/common \
  @nestjs/core \
  rxjs@6.0.0 \
  typescript \
  reflect-metadata
```

## Add core packages

```
➔ npm i \
    @nestjs/common \
    @nestjs/core \
    rxjs@6.0.0 \
    typescript \
    reflect-metadata \
      @nestjs/platform-express
```

## Add core package

- npm i \
  - @nestjs/common \
  - @nestjs/core \
  - rxjs@6.0.0 \
  - typescript \
  - reflect-metadata \
    - @nestjs/platform-e



[v6] 'No driver (HTTP) has been selected' #1609

Closed murbanowicz opened this issue on 5 Mar · 5 comments

murbanowicz commented on 5 Mar

I'm submitting a...

```
[ ] Regression
[x] Bug report
[ ] Feature request
[x] Documentation issue or request
[ ] Support request => Please do not submit support request here, instead post your question on S
```

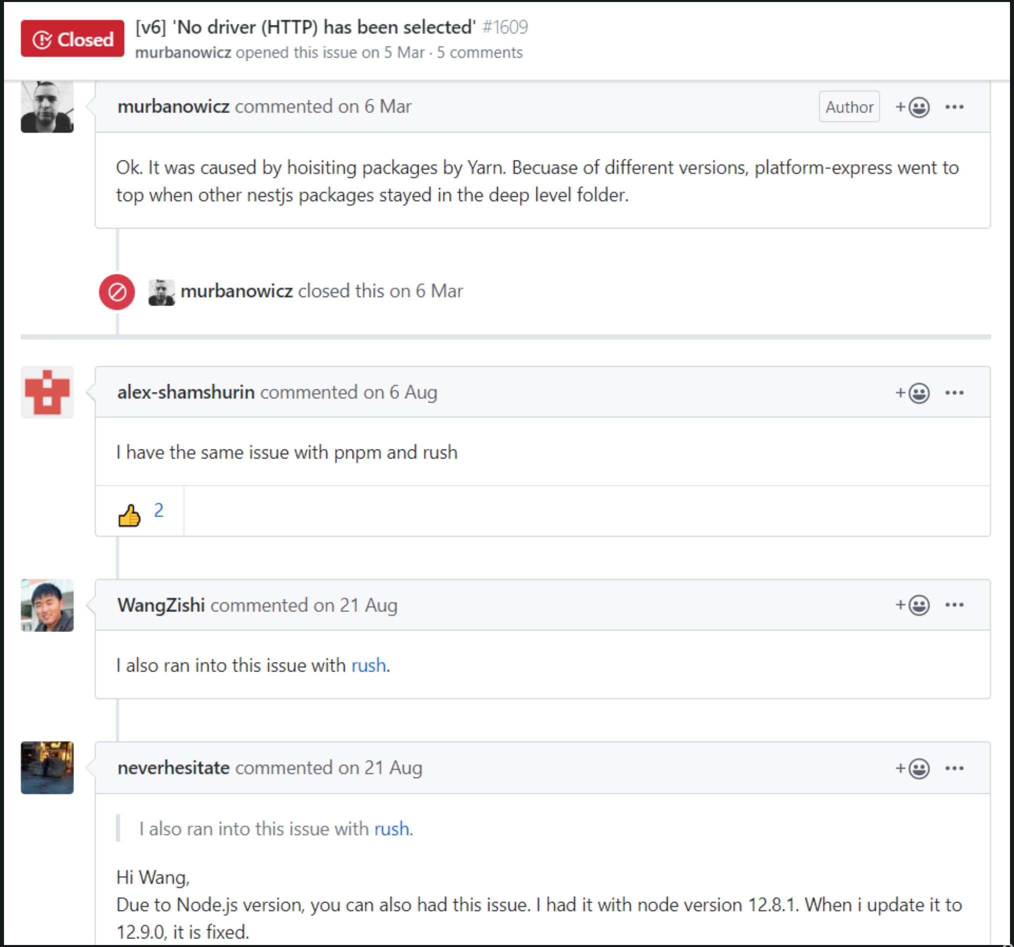
Current behavior

```
[PackageLoader] No driver (HTTP) has been selected. In order to take advantage of the default
driver, please, ensure to install the "@nestjs/platform-express" package ($ npm install
@nestjs/platform-express). [nodemon] app crashed - waiting for file changes before starting...
```

https://github.com/nestjs/nest/issues/1609

## Add core packages

```
➔ npm i \
    @nestjs/common \
    @nestjs/core \
    rxjs@6.0.0 \
    typescript \
    reflect-metadata \
      @nestjs/platform-express
```



Closed [v6] 'No driver (HTTP) has been selected' #1609
murbanowicz opened this issue on 5 Mar · 5 comments

**murbanowicz** commented on 6 Mar — Author

Ok. It was caused by hoisiting packages by Yarn. Becuase of different versions, platform-express went to top when other nestjs packages stayed in the deep level folder.

**murbanowicz** closed this on 6 Mar

**alex-shamshurin** commented on 6 Aug

I have the same issue with pnpm and rush

👍 2

**WangZishi** commented on 21 Aug

I also ran into this issue with rush.

**neverhesitate** commented on 21 Aug

> I also ran into this issue with rush.

Hi Wang,
Due to Node.js version, you can also had this issue. I had it with node version 12.8.1. When i update it to 12.9.0, it is fixed.

## Add dev packages

➔ npm i -D @types/node \
ts-node

## tsconfig.json -> compilerOptions

```
"emitDecoratorMetadata": true,
"experimentalDecorators": true,
```

## Create Nest app

```
import { NestFactory } from '@nestjs/core';
import { ApplicationModule } from './application.module';

async function bootstrap() {
  const app = await NestFactory.create(ApplicationModule);
  await app.listen(8081);
}

bootstrap();
```

## Create Application module

```
import { Module } from '@nestjs/common';


@Module({
  imports: [],
  controllers: []
})
export class ApplicationModule {
}
```

## Express server

## Before

```
app.listen(port, () => {
 console.log(`CMS
       listening on ${port}`);
});
```

## After

```
module.exports.cms = app;
```

## Before

```
import { NestFactory } from '@nestjs/
core';
import { ApplicationModule } from './
app.module';
```

## After

```
import { NestFactory } from '@nestjs/
core';
import { ApplicationModule } from './
app.module';
import { ExpressAdapter } from
    '@nestjs/platform-express';
import { cms } from
    './cms';
```

## Before

```
const app = await
NestFactory.create(
      ApplicationModule

);
await app.listen(8081);
```

## After

```
const app = await
NestFactory.create(
      ApplicationModule,
      new ExpressAdapter(cms)
);
await app.listen(8081);
```

```
➔ curl "http://localhost:8081/v1/login" -X POST

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
    <pre>Cannot POST /</pre>
</body>
</html>
```

```
passport.use('local', new LocalStrategy(
    (email, password, done) => ...
));

app.get(`/login`, (req, res, next) => {
    passport.authenticate('local', ...)(req, res, next);
});
```

```
➔ curl "http://localhost:8081/v1/login?email=…&password=…" -X GET

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
      <pre>Cannot GET /</pre>
</body>
</html>
```

```
'use strict';
const initialize = require('./init');

module.exports = function router(app) {
 initialize(app)
   .then(() => {
     require('./authorizations')(app);
   })
   .catch((err) => {
     console.log('Error initialization: ', err);
   });
};
```

```
return new Promise(function (resolve, reject) {
    async.parallel({
        a: Entity.findOne({name: 'Value A' }),
        b: Entity.findOne({name: 'Value B' }),
        c: Entity.findOne({name: 'Value C’ })
    }, (error, result) =>
      error ? null : resolve(result)
    );
});
```

```
➔ curl "http://localhost:8081/v1/login?email=…&password=…" -X GET

{"success": true, "error": null}
```

## Express server 2

```
import { api } from './api';


const appCms = await NestFactory.create(
  ApplicationModule,
    new ExpressAdapter(cms)
);
const appApi = await NestFactory.create(
  ApplicationModule,
    new ExpressAdapter(api)
);

await appCms.listen(8081);
await appApi.listen(8082);
```

## Cron

## Add package

→ npm i nest-schedule

## Module

```
import { Module } from '@nestjs/common';
import { ScheduleModule } from 'nest-schedule';
import { ScheduleService } from './schedule.service';

@Module({
  imports: [
    ScheduleModule.register({}),
  ],
  providers: [ScheduleService],
})
export class CronModule {}
```

## Service

```
import { Injectable } from '@nestjs/common';
import { Interval, NestSchedule } from 'nest-schedule';

@Injectable()
export class ScheduleService extends NestSchedule {
  @Interval(2000)
  intervalJob() {
    console.log('executing interval job');

    return false;
  }
}
```

```
const appCms = await NestFactory.create(
    ApplicationModule,
        new ExpressAdapter(cms)
);
const appApi = await NestFactory.create(
    ApplicationModule,
        new ExpressAdapter(api)
);
const appCron = await NestFactory.createApplicationContext(CronModule);


await appCms.listen(8081);
await appApi.listen(8082);
await appCron.init();
```

```
➔ npm start
[Nest] - [NestFactory] Starting Nest application...
[Nest] - [InstanceLoader] AppModule dependencies initialized +12ms
[Nest] - [NestFactory] Starting Nest application... +2ms
[Nest] - [InstanceLoader] AppModule dependencies initialized +3ms
[Nest] - [NestFactory] Starting Nest application... +1ms
[Nest] - [InstanceLoader] CronModule dependencies initialized +4ms
[Nest] - [InstanceLoader] ScheduleModule dependencies initialized +0ms
[Nest] - [RoutesResolver] AppController {/}: +5ms
[Nest] - [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] - [NestApplication] Nest application successfully started +1ms
[Nest] - [RoutesResolver] AppController {/}: +7ms
[Nest] - [RouterExplorer] Mapped {/, GET} route +2ms
[Nest] - [NestApplication] Nest application successfully started +1ms
executing interval job
executing interval job
executing interval job
```

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App

72:00

## Migrate to typescript?

## Migrate from js to ts?

- 3 common modules between javascript & typescript apps

## Migrate from js to ts?

- 3 common modules between javascript & typescript apps
- static type checking

## Migrate from js to ts?

- 3 common modules between javascript & typescript apps
- static type checking
- typescript or Babel for nodejs v8.12.0 (>= 8.9.0)

# Migrate from js to ts?

- 3 common modules between javascript & typescript apps
- static type checking
- typescript or Babel for nodejs v8.12.0 (>= 8.9.0)
- nestjs fully supports typescript

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ● Migrate from js to ts

## Migration

## js -> ts

```
'use strict';
const initialize = require('./init');

module.exports = function router(app) {
 initialize(app)
   .then(() => {
     require('./auth')(app);
   })
   .catch((err) =>
     console.log(error);
   );
};
```

## Before

```
'use strict';
const initialize =
  require('./init');
```

## After

```
import { initialize } from
  './init';
```

## Before

```
module.exports = function router(app) {
};
```

## After

```
export default function (app) {
};
```

## Before

```
module.exports = function router(app) {
};
```

## After

```
export const
router = function (app) {
};
```

## Before

```
initialize(app)
  .then(() => {
    require('./auth')(app);
  })
  .catch((error) =>
    console.log(error)
  );
```

## After

```
import { authorization } from './auth;
```

```
initialize(app)
  .then(() => {
    authorization(app);
  })
  .catch((error) =>
    console.log(error)
  );
```

# javascript

```
'use strict';
const initialize = require('./init');

module.exports = function router(app) {
 initialize(app)
   .then(() => {
     require('./auth')(app);
   })
   .catch((err) =>
     console.log(error);
   );
};
```

## typescript

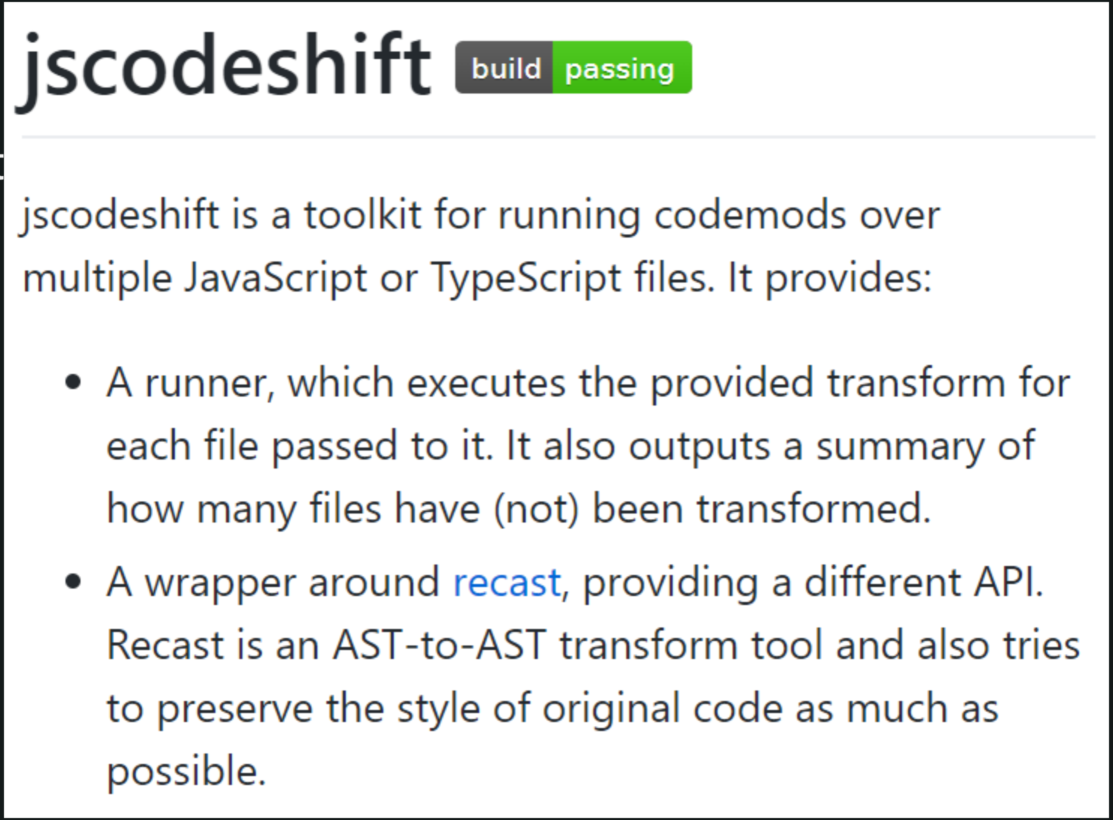
```
import { initialize } from './init';
import { authorization } from './auth;

export const router = function (app) {
 initialize(app)
   .then(() => {
     authorization(app);
   })
   .catch((error) =>
     console.log(error)
   );
};
```

## Semi-automatic tools

- jscodeshift + extensions

- jscodeshift



# jscodeshift build passing

jscodeshift is a toolkit for running codemods over multiple JavaScript or TypeScript files. It provides:

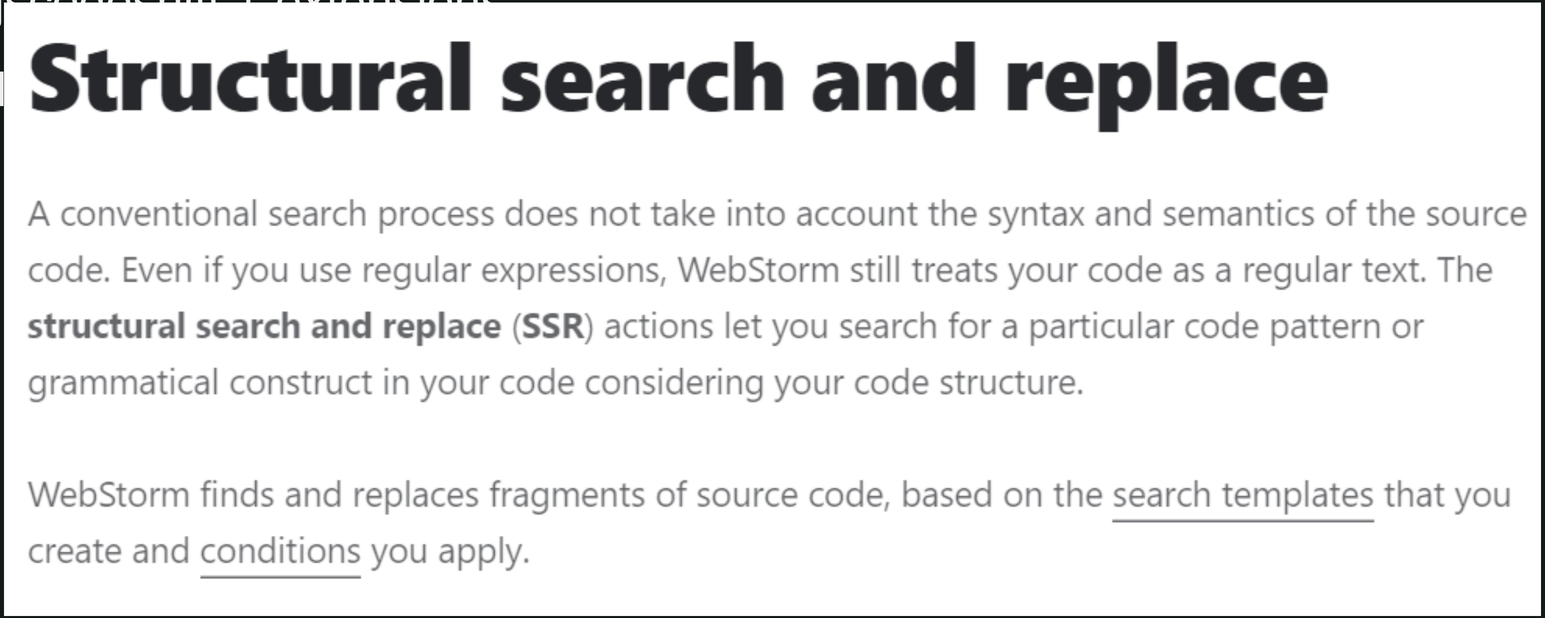
- A runner, which executes the provided transform for each file passed to it. It also outputs a summary of how many files have (not) been transformed.
- A wrapper around recast, providing a different API. Recast is an AST-to-AST transform tool and also tries to preserve the style of original code as much as possible.

- jscodeshift + extensions https://github.com/cpojer/js-codemod

#jscodeshift-imports
#template-literals
#arrow-function
#rm-requires
#no-vars

- jscodeshift + extensions
- IDE (webstorm)

- jscodeshift + extensions



## Structural search and replace

A conventional search process does not take into account the syntax and semantics of the source code. Even if you use regular expressions, WebStorm still treats your code as a regular text. The **structural search and replace** (**SSR**) actions let you search for a particular code pattern or grammatical construct in your code considering your code structure.

WebStorm finds and replaces fragments of source code, based on the search templates that you create and conditions you apply.

- jscodeshift + extensions
- IDE (webstorm)
- linter  --fix

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts

50:00

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ● Nestjs simple modules

## Serve Static

@Module

```
➔ npm i @nestjs/serve-static

import { ServeStaticModule } from '@nestjs/serve-static';
```

## Before

```
const public = path.join(__dirname, ...);

app.use(express.static(public));
```

## After

```
@Module({
 imports: [
   ServeStaticModule.forRoot({
     rootPath: public
   })
 ]
})
```

## Nest app

```
import { NestFactory } from '@nestjs/core';
import { ApplicationModule } from './app.module';

async function bootstrap() {
  const appCms = await NestFactory.create(ApplicationModule);
  appCms.setGlobalPrefix('v1');
  await appCms.listen(8081);
}

bootstrap();
```

## CORS

@Module

## Nest app

```
import { NestFactory } from '@nestjs/core';
import { ApplicationModule } from './app.module';

async function bootstrap() {
  const appCms = await NestFactory.create(ApplicationModule);
  appCms.setGlobalPrefix('v1');
  appCms.enableCors();
  await appCms.listen(8081);
}

bootstrap();
```

# Load .env

@Provider

## Before

```
const pathToEnv =
  path.join(__dirname, '.env');
require('dotenv')
  .config(pathToEnv);
```

## After

```
const pathToEnv =
  path.join(__dirname, '.env');
require('dotenv')
  .config(pathToEnv);
```

## Before

```
const pathToEnv =
  path.join(__dirname, '.env');
require('dotenv')
  .config(pathToEnv);
```

## After

```
const pathToEnv =
  path.join(__dirname, '.env');
require('dotenv')
  .config(pathToEnv);
```

https://docs.nestjs.com/techniques/configuration

# DB Connection

@Module

```
➔ npm install --save @nestjs/mongoose mongoose
```

```
import { MongooseModule } from '@nestjs/mongoose';
```

## Before

```
export const dbConfig = (…) => {
  const opt = {
        useNewUrlParser: true,
        connectTimeoutMS: 5000
  };

  mongoose.connect(…, err =>
        if (err) console.error(err);
  );
};
```

## After

```
@Module({
 imports: [
  …
  MongooseModule.forRoot(…)
 ]
})
```

## Before

```
export const dbConfig = (…) => {
 const opt = {
      useNewUrlParser: true,
      connectTimeoutMS: 5000
 };

 mongoose.connect(…, err =>
      if (err) console.error(err);
 );
};
```

## After

```
@Module({
 imports: [
  …
  MongooseModule.forRoot(…)
 ]
})
```

https://docs.nestjs.com/techniques/mongodb#async-configuration

## Mongoose Schemas

## Before

```
const schema = new
mongoose.Schema({
      ...
});
mongoose.model('User', schema);

export default mongoose.model('User');
```

## After

```
export const UserSchema = schema;
```

## Backward compatibility

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules

# 46:00

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ● Nestjs Auth module

## Passport

@Module

```
➔ npm install --save @nestjs/passport passport
```

```
import { PassportModule } from '@nestjs/passport';
```

## Before

```
app.use(passport.initialize());
app.use(passport.session());
```

## After

```
@Module({
 imports: [
  ...
  PassportModule.register({
    session: true
  })
 ]
})
```

## Local Strategy

@CustomProvider

```
→ npm install --save passport-local
→ npm install --save-dev @types/passport-local
```

```
import { LocalStrategy } from './local.strategy';
```

## Before

```
passport.use('local',
new LocalStrategy(
  (req, user, pass, done) => …
));
```

## After

```
export class LocalStrategy extends
PassportStrategy
(Strategy) {
  async validate(user, pass) {
    …
  }
}
```

## Before

```
passport.use('local',
new LocalStrategy(
  (req, user, pass, done) => …
));
```

## After

```
export class LocalStrategy extends
PassportStrategy
(Strategy) {
  async validate(user, pass) {
    …
  }
}
```

https://docs.nestjs.com/techniques/authentication

# Role

@CustomDecorator + @Guard

## Before

```
isAdmin: (req, res, next) => {
  if (
    req.user &&
    req.user.role === 'admin'
  )
    return next();

  res.redirect('/');
}
```

## After

```
import { SetMetadata } from '@nestjs/
common';

export const Roles =
       (...roles: string[]) =>
SetMetadata('roles', roles);
```

```typescript
export class RolesGuard implements CanActivate {
      canActivate(context: ExecutionContext): boolean {


      }
}
```

## Before

```
const isAdmin = (req, res, next) =>
  req.user.role === 'admin'
  ? next()
  : res.redirect('/');

app.get(`/api/user/:id`, isAdmin, getUser());
```

## After

```
@Module({
 providers: [
  {
    provide: APP_GUARD,
    useClass: RolesGuard,
  }
 ]
})
```

## Before

```
const isAdmin = (req, res, next) =>
  req.user.role === 'admin'
  ? next()
  : res.redirect('/');

app.get(`/api/user/:id`, isAdmin, getUser());
```

## After

```
@Module({
 providers: [
  {
    provide: APP_GUARD,
    useClass: RolesGuard,
  }
 ]
})
```

https://github.com/marcomelilli/nestjs-email-authentication

## To do

- Compression @Middleware
- Caching @Interceptor
- Validation @Middleware @Pipe
- CSRF @Middleware
- Rate limiting @Middleware
- Upload images @Interceptor
- Logging @Middleware
- Websocket
- Health check

[https://docs.nestjs.com/middleware](https://docs.nestjs.com/middleware)

[https://docs.nestjs.com/pipes](https://docs.nestjs.com/pipes)

[https://docs.nestjs.com/interceptors](https://docs.nestjs.com/interceptors)

- Compression @Middleware
- Caching @Interceptor
- Validation @Middleware @Pipe
- CSRF @Middleware
- Rate limiting @Middleware
- Upload images @Interceptor
- Logging @Middleware
- Websocket
- Health check

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module

42:00

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ● Refactoring Controllers

## Controller

## Before

```
const isAdmin =
    (req, res, next) =>
        req.user.role === 'admin'
        ? next()
        : res.redirect('/');

app.put(..., isAdmin, ...);
app.get(..., isAdmin, ...);
```

## After

```
@Roles('admin')
@Roles('admin')
```

## Before

```
app.put(`/v1/user/:id`, ...);
app.get(`/v1/user/:id`, ...);
```

## After

```
app.setGlobalPrefix('v1');

@Controller('/user')

@Put(`/:id`)
@Get(`/:id`)
```

## Before

```
app.put(...,
  function editUser (req, res) {...}
);



app.get(..., getUser());
function getUser() {
  return (req, res) => {...}
}
```

## After

```
async editUser
(@Body() userDto: userDto) {
  return
        this.service.editUser(userDto);
}

async getUser
(@Params() id: ObjectId) {
  return
        this.service.getUser(id);
}
```

## Controller Before

```
const isAdmin = (req, res, next) =>
        req.user.role === 'admin' ? next() : res.redirect('/');

app.get(`/api/user/:id`, isAdmin, getUser());
app.put(`/api/user/:id`, isAdmin, function editUser (req, res) {
        Users.update(...).exec(() => res.json({...}));
});

function getUser() {
        return (req, res) =>
                Users.findOne(...).exec(() => res.json({...}))
}
```

## Controller After

```
import { Body, Controller, Get, Param, Put } from '@nestjs/common';
import { EditUserDto, UsersService } from './user.service';
import { Roles } from './roles.decorator';

@Controller('/user')
export class AppController {
  constructor(private readonly userService: UsersService) {}

  @Put(`/:id`)
  @Roles('admin')
  async editUser (@Body() editUserDto: EditUserDto) {
    return this.userService.editUser(editUserDto);
  }
}
```

## Service

## Before

```
app.get(..., getUser());

function getUser() {
  return (req, res) =>
       Users.findOne(...).exec(
         () => res.json({...})
       )
}
```

## After

```
getUser (idUser: ObjectId) {
  return this.users
       .findOne(...).exec(...);
}
```

## Before

```
app.put(...,
  function editUser (req, res) {
       Users.update(...).exec(
         () => res.json({...})
       );
  }
);
```

## After

```
editUser (userDto: UserDto) {
  this.users
       .update(...).exec(...);
}
```

## Before

```
app.put(...,
  function editUser (req, res) {
       Users.update(...).exec(
         () => res.json({...})
       );
  }
);
```

## After

```
async
editUser (userDto: UserDto) {
  this.users
       .update(...).exec(...);
}
```

## Before

```
app.put(...,
  function editUser (req, res) {
       Users.update(...).exec(
         () => res.json({...})
       );
  }
);
```

## After

```
async
editUser (userDto: UserDto) {
  return this.users
       .update(...).exec(...);
}
```

## Before

```
app.put(...,
  function editUser (req, res) {
       Users.update(...).exec(
         () => res.json({...})
       );
  }
);
```

## After

```
async
editUser (userDto: UserDto) {
  return this.users
       .update(...).exec();
}
```

## Service Before

```
const isAdmin = (req, res, next) =>
        req.user.role === 'admin' ? next() : res.redirect('/');

app.get(`/api/user/:id`, isAdmin, getUser());
app.put(`/api/user/:id`, isAdmin, function editUser (req, res) {
        Users.update(...).exec(() => res.json({...}));
});

function getUser() {
        return (req, res) =>
                Users.findOne(...).exec(() => res.json({...}))
}
```

## Service After

```
import { Injectable } from '@nestjs/common';
import { InjectModel } from '@nestjs/mongoose';
import { Model } from 'mongoose';

@Injectable()
export class UserService {
	constructor(@InjectModel('User') users: Model<User>) {}) {}

	async getUser (idUser: ObjectId) {
		return this.users.findOne(...).exec(...);
	}

	async editUser (editUserDto: EditUserDto) {
				return this.users.update(...).exec(...);
	}
}
```

## jscodeshift

```
function transformer(fileInfo, api, options) {
 const j = api.jscodeshift;
 const root = j(fileInfo.source);

 root.find(j.ClassDeclaration)
   .replaceWith(p => {
     p.node.decorators = [
       j.decorator(j.callExpression(j.identifier('Injectable'), []))
     ];
     return p.node;
   });

 return root.toSource();
};
```

## e2e tests

```
➔ npm run e2e
I/testLogger -
   I/launcher - 0 instance(s) of WebDriver still running
        ✓ I/launcher - chrome #01-0 passed        ✓ I/launcher - chrome #01-2 passed
        ✓ I/launcher - chrome #01-1 passed        ✓ I/launcher - chrome #01-12 passed
        ✓ I/launcher - chrome #01-3 passed        ✓ I/launcher - chrome #01-15 passed
        ✓ I/launcher - chrome #01-4 passed        ✓ I/launcher - chrome #01-13 passed
        ✓ I/launcher - chrome #01-6 passed        ✓ I/launcher - chrome #01-16 passed
        ✓ I/launcher - chrome #01-5 passed        ✓ I/launcher - chrome #01-17 passed
        ✓ I/launcher - chrome #01-9 passed        ✓ I/launcher - chrome #01-14 passed
       ✓ I/launcher - chrome #01-10 passed        ✓ I/launcher - chrome #01-18 passed
        ✓ I/launcher - chrome #01-8 passed        ✓ I/launcher - chrome #01-7 passed
       ✓ I/launcher - chrome #01-11 passed
```

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers

08:00

OV
ND STA

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ● Performance testing

## Add dev packages

➔ npm i -D \

```
artillery
artillery-plugin-expect
```

## Before

Scenarios launched:  300
Scenarios completed: 300
RPS sent: 9.95
Request latency:
  min: 140.4
  max: 1096.4
  median: 365.2
  p95: 601.3
  p99: 755.3

## After

Scenarios launched:  300
Scenarios completed: 300
RPS sent: 9.87
Request latency:
  min: 150.2
  max: 1295.8
  median: 386.7
  p95: 769.1
  p99: 915.2

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing

04:00

## Worthwhile cause

- accident prevention and response
- discovery hidden defects
- detect legacy data
- minimal changes

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing
- ● Business value

## @Interceptor

- bind extra logic before / after method execution
- transform the result returned from a function
- transform the exception thrown from a function
- extend the basic function behavior
- completely override a function depending on specific conditions (e.g., for caching purposes)

- bind extra logic before / after method execution
- **transform the result returned from a function**
- transform the exception thrown from a function
- extend the basic function behavior
- completely override a function depending on specific conditions (e.g., for caching purposes)

## Response

@Interceptor

```
export class ResponseInterceptor {
 intercept(context, next) {
   return next.handle().pipe( tap( data =>
      const res = context.switchToHttp().getResponse();

      if (!isCustomResponse(data)) {
         logger.log(...);
      }

      return res.json(data);
```

## Timeout

@Interceptor

```
export class TimeoutInterceptor {
  intercept(context, next) {
    return next.handle().pipe( timeout( 5000 ) )
  }
}
```

## Controller

```
@Controller()
@UseInterceptors(TimeoutInterceptor, ResponseInterceptor)
export class AppController {}
```

## Steps

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing
- ✓ Business value

## Conclusion

00:00

## Checklist

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing
- ✓ Business value

## Checklist

- ✓ Quantify the amount of work
- ✓ Create Nestjs App

- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing
- ✓ Business value

## Checklist

- ✓ Quantify the amount of work
- ✓ Create Nestjs App
- ✓ Performance testing
- ✓ Migrate from js to ts
- ✓ Nestjs simple modules
- ✓ Nestjs Auth module
- ✓ Refactoring Controllers
- ✓ Performance testing
- ✓ Business value

## Todo Nest

- Compression @Middleware
- Caching @Interceptor
- Validation @Middleware @Pipe
- CSRF @Middleware
- Rate limiting @Middleware
- Upload images @Interceptor
- Logging @Middleware
- Websocket
- Health check

## Todo Repo

- Separate common DB layer (repository)

## Todo Repo

- Separate common DB layer (repository)
- Health check

## Todo Repo

- Separate common DB layer (repository)
- Health check
- Refactoring

## Todo Repo

- Separate common DB layer (repository)
- Health check
- Refactoring
- Tests, tests, tests

- https://github.com/nestjs/nest/issues/1609
- https://github.com/facebook/jscodeshift
- https://github.com/cpojer/js-codemod
- https://docs.nestjs.com/techniques/configuration
- https://docs.nestjs.com/techniques/mongodb#async-configuration
- https://docs.nestjs.com/techniques/authentication
- https://docs.nestjs.com/guards#putting-it-all-together
- https://github.com/marcomelilli/nestjs-email-authentication
- https://medium.com/@kyuwoo.choi/sneak-peek-to-javascript-aop-16458f807842

# ?