
nest
Framework Node.js para criar aplicações eficientes, confiáveis e escalonáveis.
OpenLabs VI
#fromdevstodevs

Wender Machado

- Desenvolvedor Full-Stack
  (RG Sistemas)
- Cursando Sistemas de Informação
  (CEFET/RJ Nova Friburgo)
- Formado em Técnico de Informática
  (CEFET/RJ Nova Friburgo)

/wenderpmachado

@wenderpmachado

NestJS - Quem criou?

Cat person

Kamil Mysliwiec
kamilmysliwiec

Follow

Block or report user

Creator of **@nestjs**. Co-Founder of **@TrilonIO**. Speaker, Trainer and Consultant. OpenSource (OSS) enthusiast.

@nestjs
Poland

Overview | Repositories 29 | Projects 0 | Stars 127 | Followers 858 | Following 3

Pinned

nestjs/nest
A progressive Node.js framework for building efficient, scalable, and enterprise-grade server-side applications on top of TypeScript & JavaScript (ES6, ES7, ES8) 🚀
TypeScript ★ 13.8k ⑂ 902

nestjs/typescript-starter
Nest framework TypeScript starter
TypeScript ★ 341 ⑂ 115

nestjs/swagger
Swagger module for Nest framework (node.js) 🌎
TypeScript ★ 227 ⑂ 69

nestjs/docs.nestjs.com
The official documentation https://docs.nestjs.com
TypeScript ★ 115 ⑂ 133

nestjs/graphql
GraphQL (Apollo) module for Nest framework (node.js) 🍷
TypeScript ★ 208 ⑂ 33

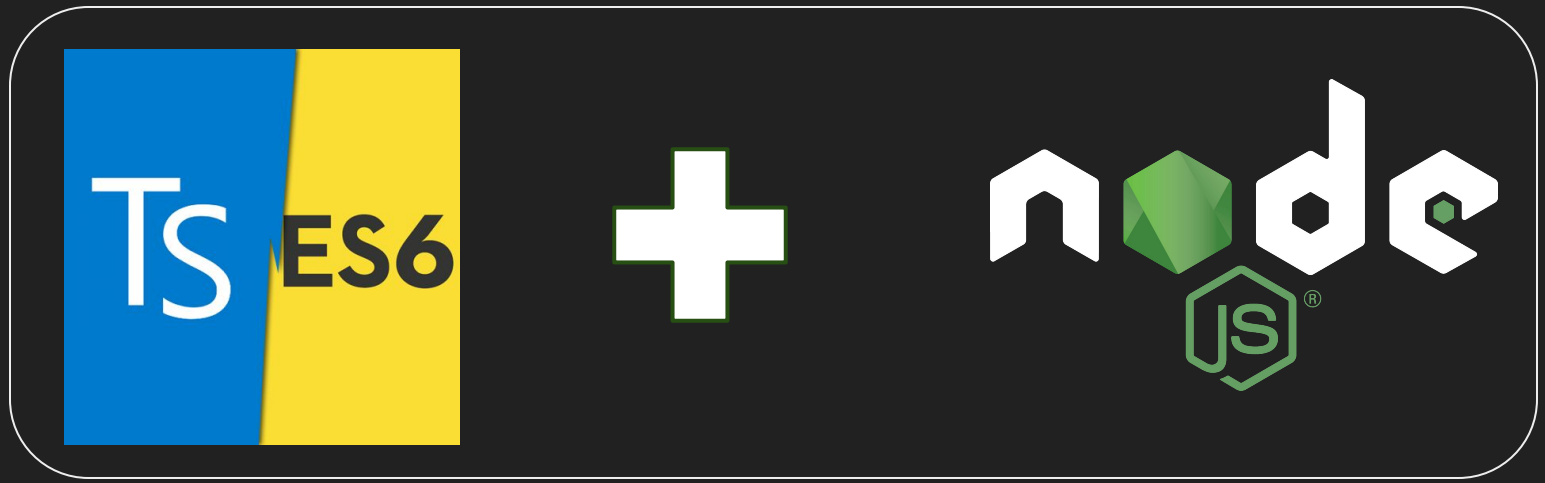
nestjs/nest-cli
CLI tool for Nest applications
TypeScript ★ 365 ⑂ 58

2,117 contributions in the last year

2019

NestJS - O que é?

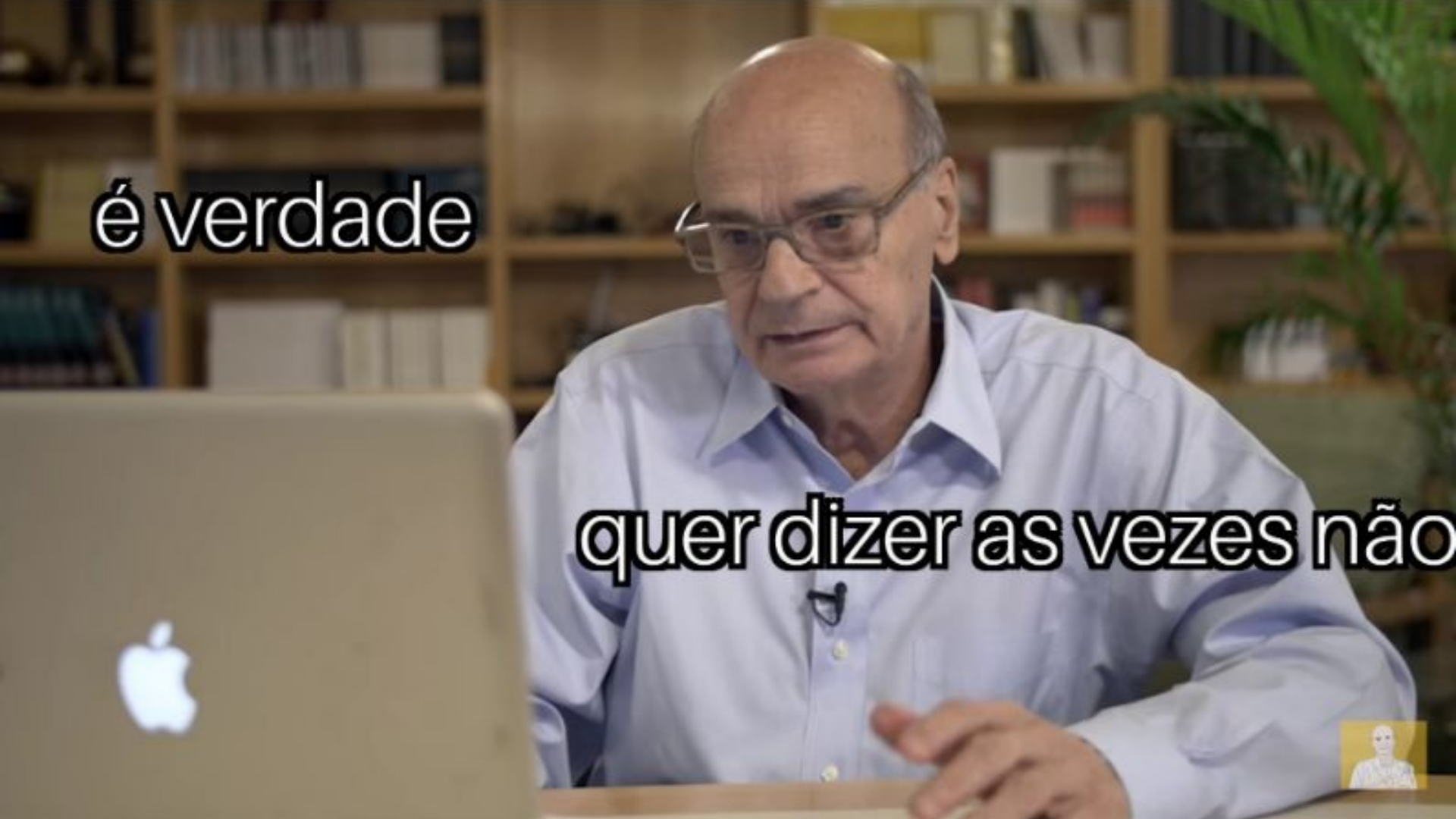
BACKEND/API



Watch	479	Star	13,811	Fork	904

+1

Framework Node/Javascript ?!?!?


é verdade
quer dizer as vezes não

NestJS - O que **REALMENTE** é?

PRODUTIVIDADE

PERFORMANCE

ARQUITETURA

ESCALABILIDADE

VERSATILIDADE

NestJS - O que **REALMENTE** é?

ARQUITETURA

ESCALABILIDADE

PRODUTIVIDADE

PERFORMANCE

VERSATILIDADE

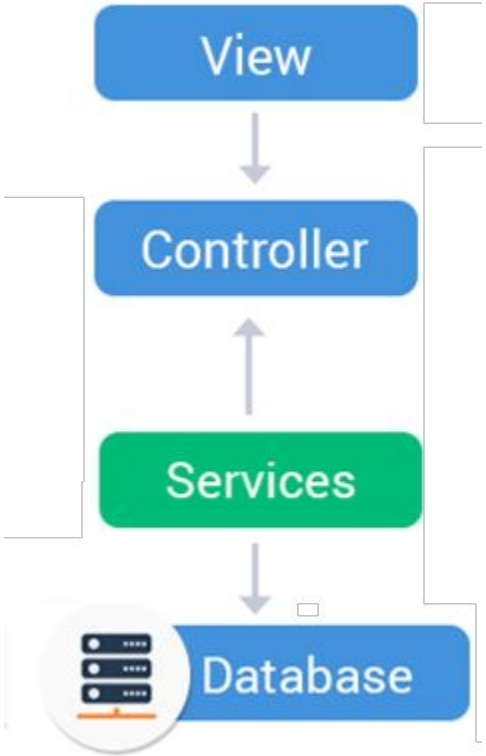
```
const http = require('http');
const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer(function(req, res) {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, function() {
  console.log('Server running at http://'+ hostname + ':' + port + '/');
});
```

Node.js - Exemplo *Getting Started*

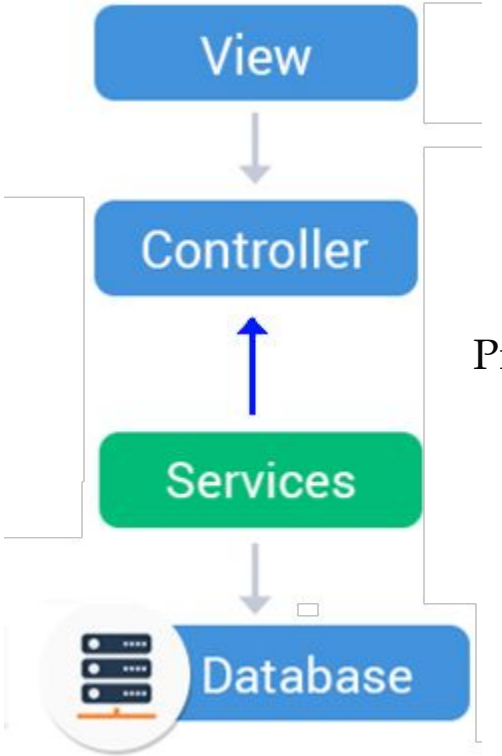
```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

app.listen(3000, function () {
  console.log('Example app listening on port 3000!');
});
```

Express.js - Exemplo *Getting Started*

NestJS - *Layers* (Camadas)



NestJS - *Layers* (Camadas)



Princípio da inversão de dependência

Saiba mais em: SOLID - Princípios POO
Link: http://encurtador.com.br/czCH3

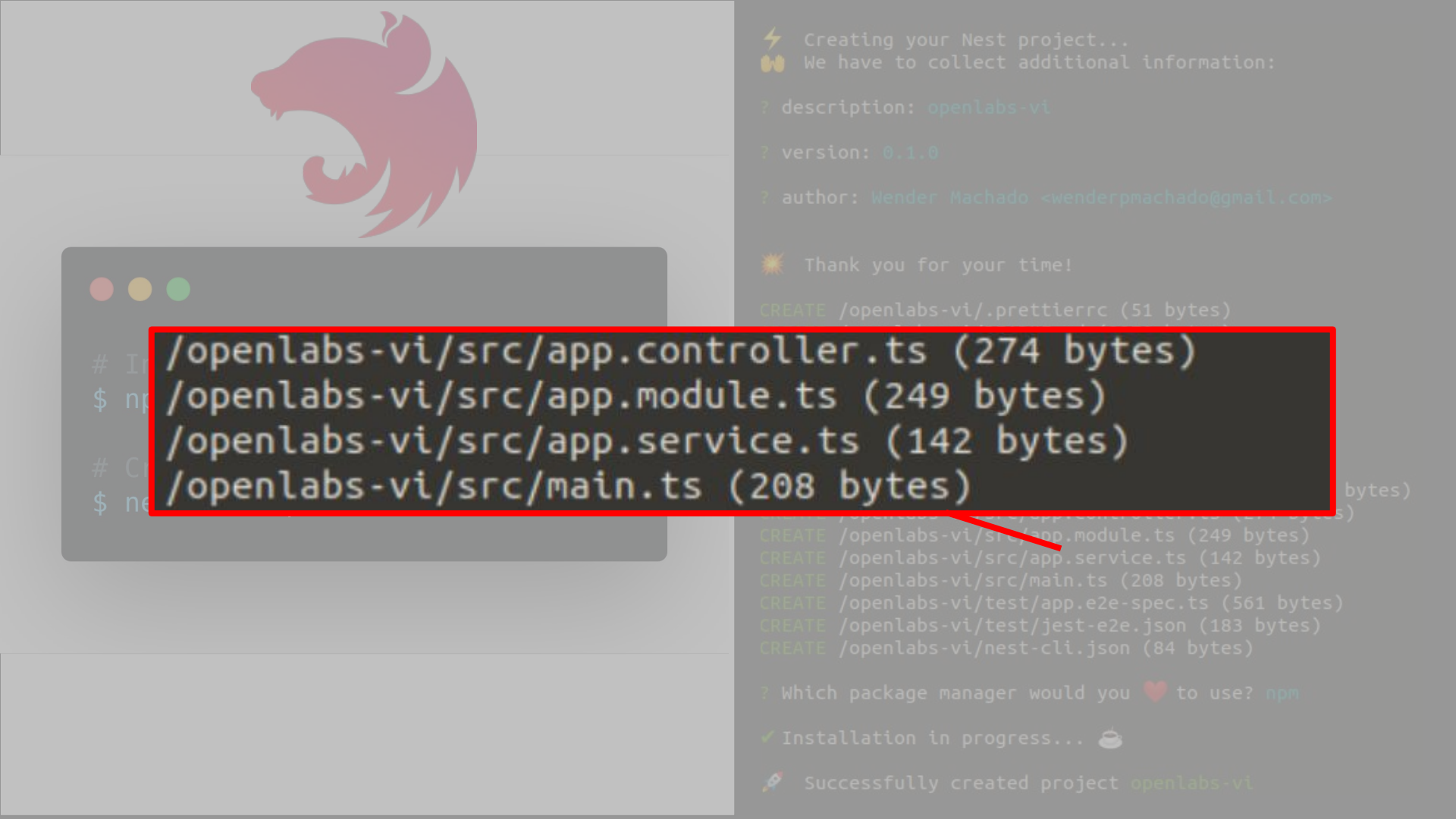
nest

NA PRÁTICA

```
# Instalando globalmente o NestJS
$ npm i -g @nestjs/cli

# Criando um novo projeto
$ nest new openlabs-vi
```

```
⚡  Creating your Nest project...
🙌  We have to collect additional information:

? description: openlabs-vi

? version: 0.1.0

? author: Wender Machado <wenderpmachado@gmail.com>


💥  Thank you for your time!

CREATE /openlabs-vi/.prettierrc (51 bytes)
CREATE /openlabs-vi/README.md (3370 bytes)
CREATE /openlabs-vi/nodemon-debug.json (163 bytes)
CREATE /openlabs-vi/nodemon.json (132 bytes)
CREATE /openlabs-vi/package.json (1699 bytes)
CREATE /openlabs-vi/tsconfig.build.json (89 bytes)
CREATE /openlabs-vi/tsconfig.json (300 bytes)
CREATE /openlabs-vi/tslint.json (426 bytes)
CREATE /openlabs-vi/src/app.controller.spec.ts (617 bytes)
CREATE /openlabs-vi/src/app.controller.ts (274 bytes)
CREATE /openlabs-vi/src/app.module.ts (249 bytes)
CREATE /openlabs-vi/src/app.service.ts (142 bytes)
CREATE /openlabs-vi/src/main.ts (208 bytes)
CREATE /openlabs-vi/test/app.e2e-spec.ts (561 bytes)
CREATE /openlabs-vi/test/jest-e2e.json (183 bytes)
CREATE /openlabs-vi/nest-cli.json (84 bytes)

? Which package manager would you ❤️ to use? npm

✔ Installation in progress... ☕

🚀  Successfully created project openlabs-vi
```

Mais em: https://docs.nestjs.com/first-steps


/openlabs-vi/src/app.controller.ts (274 bytes)
/openlabs-vi/src/app.module.ts (249 bytes)
/openlabs-vi/src/app.service.ts (142 bytes)
/openlabs-vi/src/main.ts (208 bytes)
Creating your Nest project...
We have to collect additional information:
? description: openlabs-vi
? version: 0.1.0
? author: Wender Machado <wenderpmachado@gmail.com>
Thank you for your time!
CREATE /openlabs-vi/.prettierrc (51 bytes)
CREATE /openlabs-vi/src/app.module.ts (249 bytes)
CREATE /openlabs-vi/src/app.service.ts (142 bytes)
CREATE /openlabs-vi/src/main.ts (208 bytes)
CREATE /openlabs-vi/test/app.e2e-spec.ts (561 bytes)
CREATE /openlabs-vi/test/jest-e2e.json (183 bytes)
CREATE /openlabs-vi/nest-cli.json (84 bytes)
? Which package manager would you ❤️ to use? npm
✔ Installation in progress... ☕
Successfully created project openlabs-vi

express

(DEFAULT main.ts)

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

async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  await app.listen(3000);
}
bootstrap();
```

fastify

```
$ npm i --save @nestjs/platform-fastify
```

```ts
import { NestFactory } from '@nestjs/core';
import {
  FastifyAdapter,
  NestFastifyApplication,
} from '@nestjs/platform-fastify';
import { ApplicationModule } from './app.module';

async function bootstrap() {
  const app = await NestFactory.create<NestFastifyApplication>(
    ApplicationModule,
    new FastifyAdapter(),
  );
  await app.listen(3000);
}
bootstrap();
```

NestJS - Módulo

```
$ nest g module photos
CREATE /src/photos/photos.module.ts (83 bytes)
UPDATE /src/app.module.ts (316 bytes)
```

```ts
// photos.module.ts

import { Module } from '@nestjs/common';
import { PhotosController } from './photos.controller';
import { PhotosService } from './photos.service';

// @Global()
@Module({
  controllers: [PhotosController],
  providers: [PhotosService],
  exports: [PhotosService]
})
export class PhotosModule {}
```

```ts
// app.module.ts

import { Module } from '@nestjs/common';
import { PhotosModule } from './photos/photos.module';

@Module({
  imports: [PhotosModule],
})
export class AppModule {}
```

Mais em: https://docs.nestjs.com/modules

NestJS - Controladora

```
$ nest g controller photos
CREATE /src/photos/photos.controller.spec.ts (493 bytes)
CREATE /src/photos/photos.controller.ts (101 bytes)
UPDATE /src/photos/photos.module.ts (174 bytes)
```

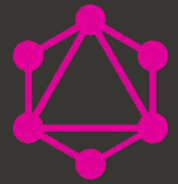
- Outros HTTP's *decorators*:
  - @Patch(), @Options() e @All()
- Outros objetos da requisição:
  - @Request(), @Response(), @Next(), @Session(), @Headers(name?: string)
- Decorators auxiliares:
  - `@UseGuards(AuthGuard())`

Mais em: https://docs.nestjs.com/controllers

```ts
// photos.controller.ts
import {
  Controller, Get, Query, Post,
  Body, Put, Param, Delete
} from '@nestjs/common';
import { CreatePhotoDto, UpdatePhotoDto, ListAllEntities } from './dto';

@Controller('photos') // Acesso: http://localhost:3000/photos
export class PhotosController {
  @Post()
  @Header('Cache-Control', 'none')
  @HttpCode(204)
  create(@Body() createPhotoDto: CreatePhotoDto) {
    return 'This action adds a new photo';
  }

  @Get()
  async findAll(@Query() query: ListAllEntities): Promise<Photo[]> {
    return `This action returns all photos (limit: ${query.limit} items)`;
  }

  @Get(':id') // Exemplo: http://localhost:3000/photos/1
  findOne(@Param('id') id: string) {
    return `This action returns a #${id} photo`;
  }

  @Put(':id')
  update(@Param('id') id: string, @Body() updatePhotoDto: UpdatePhotoDto) {
    return `This action updates a #${id} photo`;
  }

  @Delete(':id')
  remove(@Param('id') id: string) {
    return `This action removes a #${id} photo`;
  }
}
```

{ REST }

(DEFAULT)



GraphQL

```ts
// photos.controller.ts

// Imports...

@Controller('photos')
export class PhotosController {
  constructor(private readonly photosService: PhotosService) {}

  // ..

  @Get()
  findAll(): Promise<Photo[]> {
    return this.photosService.findAll();
  }

  @Post()
  async create(@Body() createPhotoDto: CreatePhotoDto) {
    this.photosService.create(createPhotoDto);
  }

  // ..

  @Delete(':id')
  async delete(@Param('id', new ParseIntPipe()) id) {
    this.photosService.delete(id);
  }
}
```

```ts
// photos.resolver.ts

// Imports...

@Resolver(of => Photo)
export class PhotosResolver {
  constructor(private readonly photosService: PhotosService) {}

  // ..

  @Query(returns => [Photo])
  photos(@Args() photosArgs: PhotosArgs): Promise<Photo[]> {
    return this.photosService.findAll(photosArgs);
  }

  @Mutation(returns => Photo)
  async addPhoto(
    @Args('newPhotoData') newPhotoData: NewPhotoInput,
  ): Promise<Photo> {
    const photo = await this.photosService.create(newPhotoData);
    pubSub.publish('photoAdded', { photoAdded: photo });
    return photo;
  }

  // ..

  @Subscription(returns => Photo)
  photoAdded() {
    return pubSub.asyncIterator('photoAdded');
  }
}
```

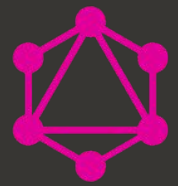
{ REST }

+

swagger

```
$ npm install --save @nestjs/swagger
$ npm install --save swagger-ui-express #ou fastify-swagger
```

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

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

  const options = new DocumentBuilder()
    .setTitle('Photos example')
    .setDescription('The photos API description')
    .setVersion('1.0')
    .addTag('photos')
    .build();
  const document = SwaggerModule.createDocument(app, options);
  SwaggerModule.setup('api', app, document);

  await app.listen(3001);
}
bootstrap();
```



GraphQL

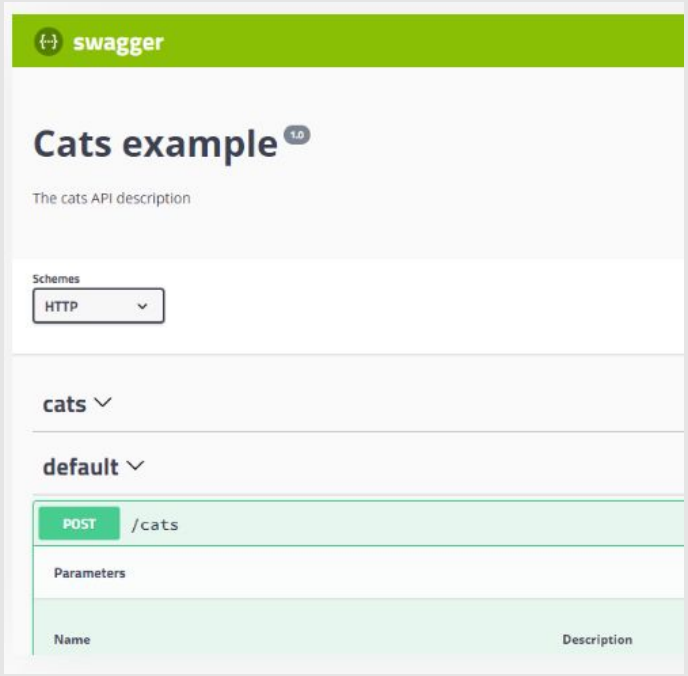
```
$ npm i --save @nestjs/graphql apollo-server-express graphql-tools graphql
$ npm i --save type-graphql
```

```
import { Module } from '@nestjs/common';
import { GraphQLModule } from '@nestjs/graphql';
import { PhotosModule } from './photos/photos.module';

@Module({
  imports: [
    PhotosModule,
    GraphQLModule.forRoot({
      installSubscriptionHandlers: true,
      autoSchemaFile: 'schema.gql',
    }),
  ],
})
export class ApplicationModule {}
```

{ REST }

+

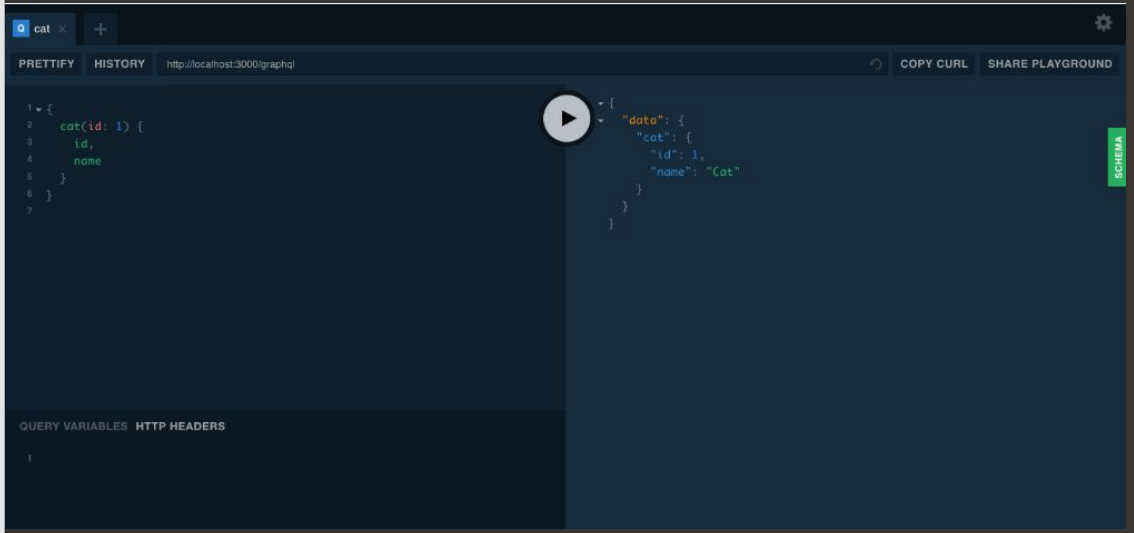
swagger



GraphQL

+

Playground



Mais em:
https://docs.nestjs.com/recipes/swagger
https://docs.nestjs.com/graphql/quick-start

{ REST }

+

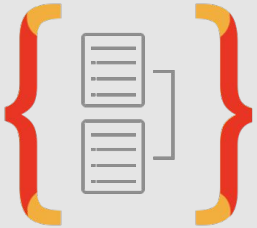
swagger

```
export const ApiModelProperty: (metadata?: {
  description?: string;
  required?: boolean;
  type?: any;
  isArray?: boolean;
  collectionFormat?: string;
  default?: any;
  enum?: SwaggerEnumType;
  format?: string;
  multipleOf?: number;
  maximum?: number;
  exclusiveMaximum?: number;
  minimum?: number;
  exclusiveMinimum?: number;
  maxLength?: number;
  minLength?: number;
  pattern?: string;
  maxItems?: number;
  minItems?: number;
  uniqueItems?: boolean;
  maxProperties?: number;
  minProperties?: number;
  readOnly?: boolean;
  xml?: any;
  example?: any;
}) => PropertyDecorator;
```

NestJS - Serviço

```
$ nest g service photos
CREATE /src/photos/photos.service.spec.ts (460 bytes)
CREATE /src/photos/photos.service.ts (90 bytes)
UPDATE /src/photos/photos.module.ts (254 bytes)
```

Mais em: https://docs.nestjs.com/providers



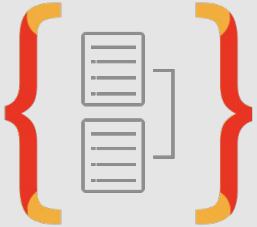
TYPEORM


```typescript
import { Column, Entity, ObjectID, ObjectIdColumn } from 'typeorm';

@Entity()
export class Photo {
  @ObjectIdColumn() // ou @ObjectId para BD Relacional
  id: ObjectID; // ou number para BD Relacional

  @Column()
  name: string;

  @Column()
  description: string;

  @Column()
  filename: string;

  @Column()
  isPublished: boolean;
}
```

```typescript
import { Column, Model, Table } from 'sequelize-typescript';

@Table
export class Cat extends Model<Cat> {
  @Column
  name: string;

  @Column
  age: number;

  @Column
  breed: string;
```

Mais em: https://docs.nestjs.com/recipes/sql-typeorm

Mais em: https://docs.nestjs.com/recipes/sql-sequelize



TYPEORM


```typescript
// TypeORM + MongoDB
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { Photo } from './photo.entity';

@Injectable()
export class PhotoService {
  constructor(
    @InjectRepository(Photo)
    private readonly photoRepository: Repository<Photo>,
  ) {}

  async findAll(): Promise<Photo[]> {
    return await this.photoRepository.find();
  }
}
```

Mais em: https://docs.nestjs.com/recipes/sql-typeorm

```typescript
// cats.provider.ts
import { Cat } from './cat.entity';

export const catsProviders = [
  {
    provide: 'CATS_REPOSITORY',
    useValue: Cat,
  },
];

// cats.service.ts
import { Inject, Injectable } from '@nestjs/common';
import { Cat } from './cat.entity';
import { CreateCatDto } from './dto/create-cat.dto';

@Injectable()
export class CatsService {
  constructor(
    @Inject('CATS_REPOSITORY') private readonly catsRepository: typeof Cat,
  ) {}

  async create(createCatDto: CreateCatDto): Promise<Cat> {
    const cat = new Cat();
    cat.name = createCatDto.name;
    cat.breed = createCatDto.breed;
    cat.age = createCatDto.age;

    return await cat.save();
  }

  async findAll(): Promise<Cat[]> {
    return await this.catsRepository.findAll<Cat>();
  }
}
```

Mais em: https://docs.nestjs.com/recipes/sql-sequelize

NestJS - Utilitários

```ts
// Exemplo FileUpload
@Post('upload')
@UseInterceptors(FileInterceptor('file'))
uploadFile(@UploadedFile() file) {
  console.log(file);
}

// Exemplos LifeCycles
import { Injectable, OnModuleInit } from '@nestjs/common';

@Injectable()
export class UsersService implements OnModuleInit, OnApplicationShutdown {
  onModuleInit() {
    console.log(`The module has been initialized.`);
  }

  onApplicationShutdown(signal: string) {
    console.log(signal); // e.g. "SIGINT"
  }
}

// Exemplo Pipes
@Post()
@UsePipes(new JoiValidationPipe(createCatSchema))
async create(@Body() createCatDto: CreateCatDto) {
  this.catsService.create(createCatDto);
}
```

```ts
// Exemplo Class Validator
import { IsString, IsInt } from 'class-validator';

export class CreateCatDto {
  @IsString()
  readonly name: string;

  @IsInt()
  readonly age: number;

  @IsString()
  readonly breed: string;
}

// Exemplos Middleware
consumer.apply(cors(), helmet(), logger).forRoutes(CatsController);

// Exemplo Filtro de Exceção
@Post()
@UseFilters(new HttpExceptionFilter())
async create(@Body() createCatDto: CreateCatDto) {
  throw new ForbiddenException();
}
```

NestJS - Testes

```ts
// cats.e2e-spec.ts
import * as request from 'supertest';
import { Test } from '@nestjs/testing';
import { CatsModule } from '../../src/cats/cats.module';
import { CatsService } from '../../src/cats/cats.service';
import { INestApplication } from '@nestjs/common';

describe('Cats', () => {
  let app: INestApplication;
  let catsService = { findAll: () => ['test'] };

  beforeAll(async () => {
    const module = await Test.createTestingModule({
      imports: [CatsModule],
    })
      .overrideProvider(CatsService)
      .useValue(catsService)
      .compile();

    app = module.createNestApplication();
    await app.init();
  });

  it(`/GET cats`, () => {
    return request(app.getHttpServer())
      .get('/cats')
      .expect(200)
      .expect({
        data: catsService.findAll(),
      });
  });

  afterAll(async () => {
    await app.close();
  });
});
```

```ts
// cats.controller.spec.ts
import { Test } from '@nestjs/testing';
import { CatsController } from './cats.controller';
import { CatsService } from './cats.service';

describe('CatsController', () => {
  let catsController: CatsController;
  let catsService: CatsService;

  beforeEach(async () => {
    const module = await Test.createTestingModule({
        controllers: [CatsController],
        providers: [CatsService],
      }).compile();

    catsService = module.get<CatsService>(CatsService);
    catsController = module.get<CatsController>(CatsController);
  });

  describe('findAll', () => {
    it('should return an array of cats', async () => {
      const result = ['test'];
      jest.spyOn(catsService, 'findAll').mockImplementation(() => result);

      expect(await catsController.findAll()).toBe(result);
    });
  });
});
```

Mais em: https://docs.nestjs.com/fundamentals/testing

TECHNIQUES

- Authentication
- Database
- Mongo
- File upload
- Validation
- Caching
- Serialization
- Logger
- Security
- Configuration
- Compression
- HTTP module
- Model-View-Controller
- Performance (Fastify)
- Hot reload (Webpack)

MICROSERVICES

- Basics
- Redis
- MQTT
- NATS
- RabbitMQ
- gRPC
- Exception filters
- Pipes
- Guards
- Interceptors

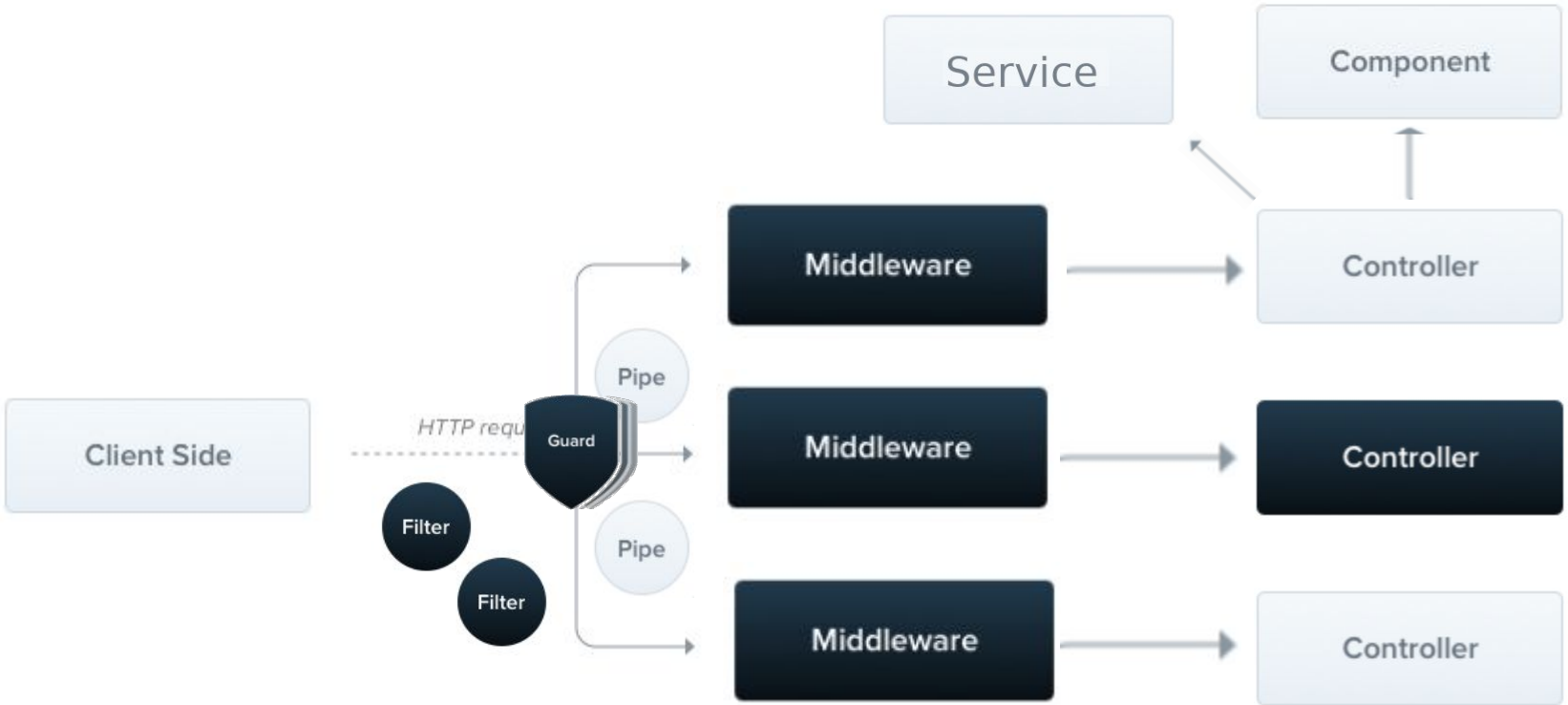
RECIPES

- TypeORM
- Mongoose
- Sequelize
- CQRS
- OpenAPI (Swagger)
- Prisma
- Health checks (Terminus)
- Documentation

WEBSOCKETS

- Gateways
- Exception filters
- Pipes
- Guards
- Interceptors
- Adapters

NestJS - Fluxo Avançado




YOU HAVE CHOSEN...
WISELY

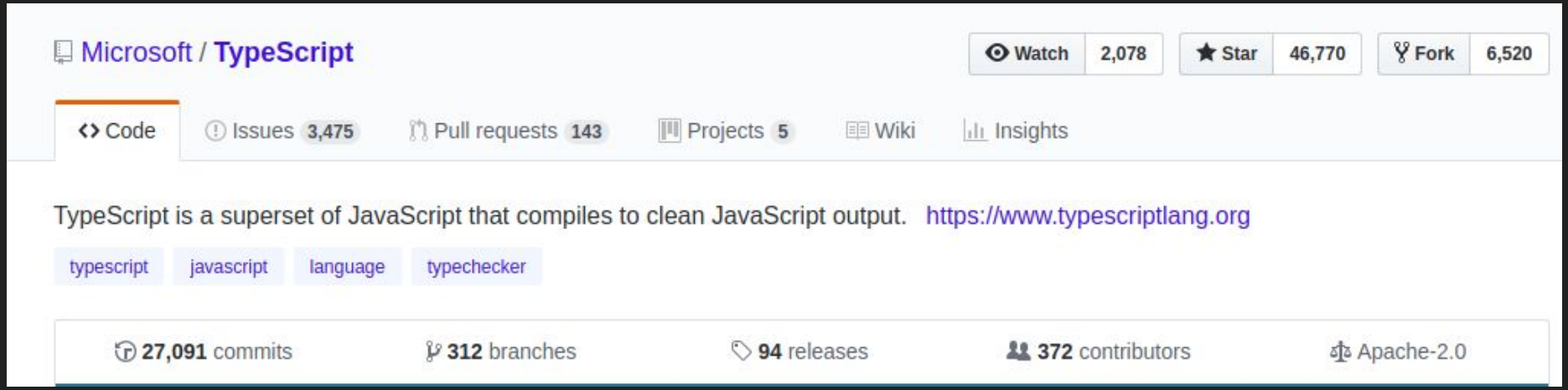
“Odeio Typescript e tudo que é tipado, posso mesmo assim usar o NestJS?”
- *pergunta de Fulano Dital*

SIM! Há possibilidade de utilizar apenas o ES6 com Babel.


Perguntas?

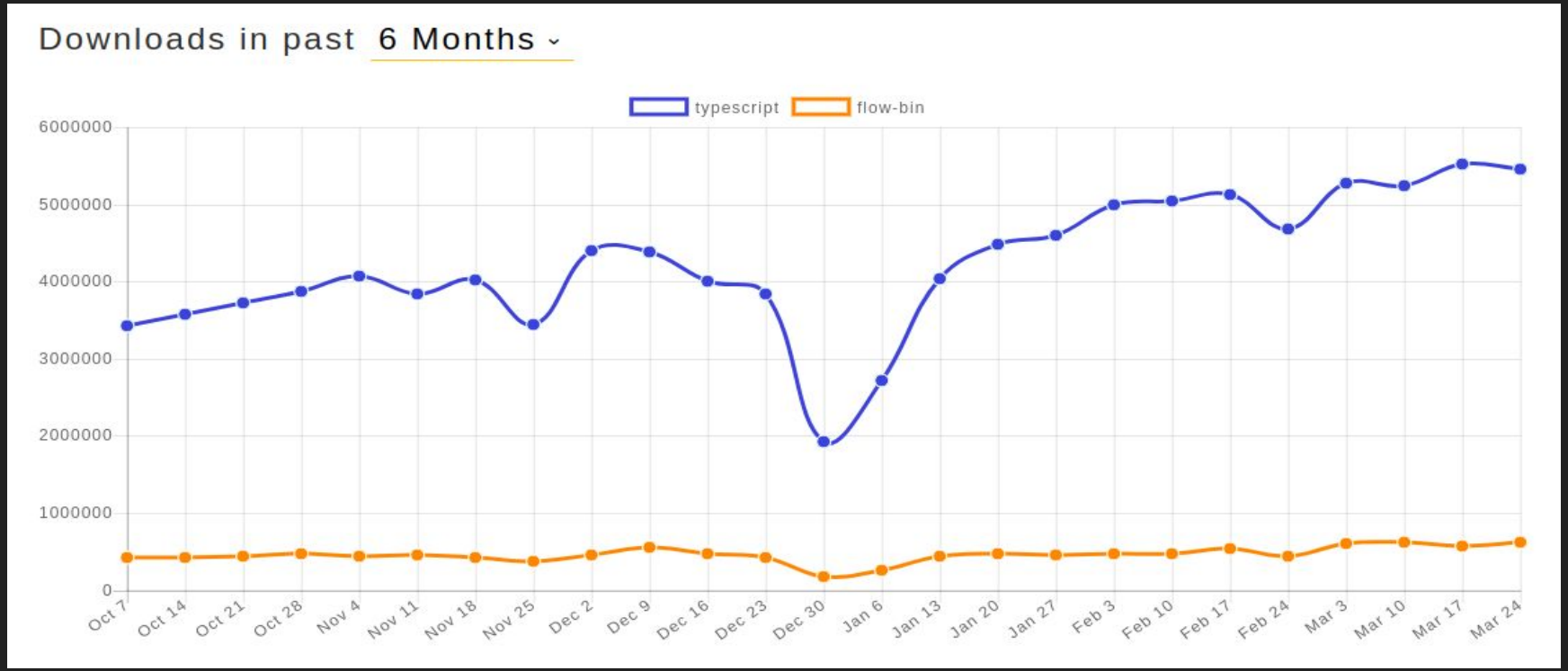

EXTRA!

Typescript



TypeScript é um **superconjunto** de JavaScript desenvolvido pela **Microsoft** que adiciona **tipagem** e alguns outros recursos a linguagem.

Typescript vs Flow (Facebook)



Fonte: https://www.npmtrends.com/flow-bin-vs-typescript

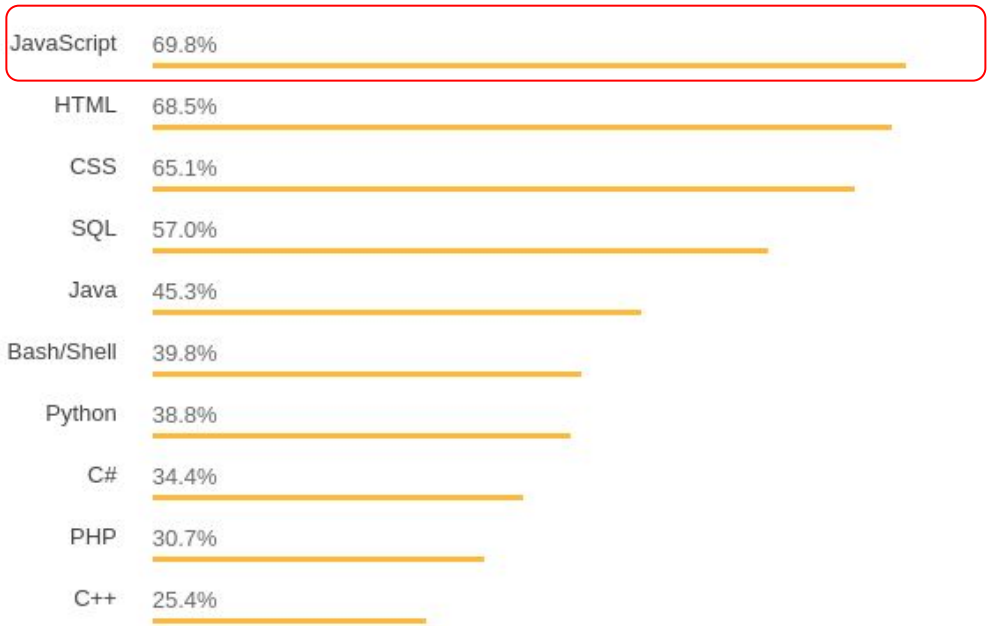
NestJS - Quem usa?

Fonte: https://docs.nestjs.com/discover/companies

Most Popular Technologies

Programming, Scripting, and Markup Languages

All Respondents | Professional Developers



Fonte: https://insights.stackoverflow.com/survey/2018

Most Popular Technologies

Frameworks, Libraries, and Tools

All Respondents | Professional Developers

Fonte: https://insights.stackoverflow.com/survey/2018

Fonte: https://insights.stackoverflow.com/survey/2018

OpenLabs VI

#fromdevstodevs

Obrigado!

/wenderpmachado

@wenderpmachado

wenderpmachado@gmail.com