# Iniciar uma aplicação Node.js com Typescript

***Yarn init -y*** => cria o package.json. o parâmetro -y é para criar sem fazer perguntas.

# instalação do Typescript, como uma dependência de desenvolvimento.

O código final vai ser convertido em Javascript, então instalamos o typescript no ambiente de homologação.

**yarn add typescript ts-node-dev @types/node tsconfig-paths -D**

# Criar o arquivo "tsconfig.json" que contém as configurações do Typescript

**npx tsc --init --rootDir src --outDir build --esModuleInterop --resolveJsonModule --lib es6 --module commonjs --allowJs true --noImplicitAny true**

rootDir: É aqui que o TypeScript procura nosso código.

outDir: Onde o TypeScript coloca nosso código compilado.

esModuleInterop: Se estiver usando commonjs como sistema de módulo (recomendado para aplicativos Node), então esse parâmetro deve ser definido como true.

resolveJsonModule: Se usarmos JSON neste projeto, esta opção permite que o TypeScript seja usado.

lib: Esta opção adicional de ambiente ao nosso projeto permite-nos contar com recursos de versões diferentes do Ecmascript, bibliotecas de teste e até mesmo a API DOM do navegador. Usaremos recursos es6 da linguagem.

# Usaremos a biblioteca ts-node-dev para execução da aplicação em desenvolvimento.

Passamos uma configuração:

"scripts": {

"dev": "ts-node-dev --inspect --transpile-only --ignore-watch node\_modules src/server.ts"

}

Imports obrigatórios no server.ts

import 'express-async-errors';

import cors from "cors";

import "reflect-metadata";

import cors from "cors";

import { errors } from 'celebrate';

import routes from "@shared/routes";