grande porte

Angular 9

Aula #3

Vídeo Aula e Exercícios

Esta apostila é referente a vídeo aula #3 e a sequência de exercícios #3 do curso Angular 9. Não deixe de assistir a vídeo aula e fazer os exercícios propostos para esta aula em nossa Plataforma Online.

Sobre os materiais da Grande Porte

Todos os materiais desenvolvidos pela Grande Porte objetivam ensinar de maneira eficiente, explorando potenciais e sanando dificuldades. As atualizações são feitas constantemente para manter o conteúdo atualizado com as demandas de mercado e tendências para o futuro. Esperamos que você aproveite este material. Comentários, críticas e sugestões serão muito bem-vindos.

ATENÇÂO

Versão 1.0.0
Março/2020

A estrutura de pastas de um projeto Angular 9

Na aula passada criamos um projeto utilizando o Angular CLI através do comando **ng new nome_do_projeto** e, no diretório onde esse comando foi digitado, uma estrutura de diretórios ou pastas foi criado, tendo o nome do projeto utilizado para nomear a pasta raiz. A figura 1 mostra essa estrutura. Como o meu Windows Explorer está configurado para mostrar os arquivos ocultos e do sistema operacional, podemos ver a estrutura completa.

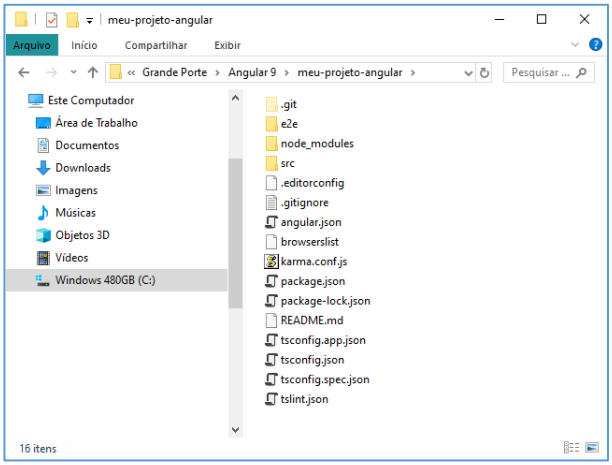

Figura 1 – A estrutura de diretórios de um projeto Angular 9.

Mas a melhor forma de estudarmos essa estrutura de pastas é abrindo este projeto no Visual Studio Code. No menu principal escolha: File → Open Folder e encontre a pasta onde seu projeto está localizado, finalmente clique selecionar pasta para abrir o projeto, conforme figura 2.

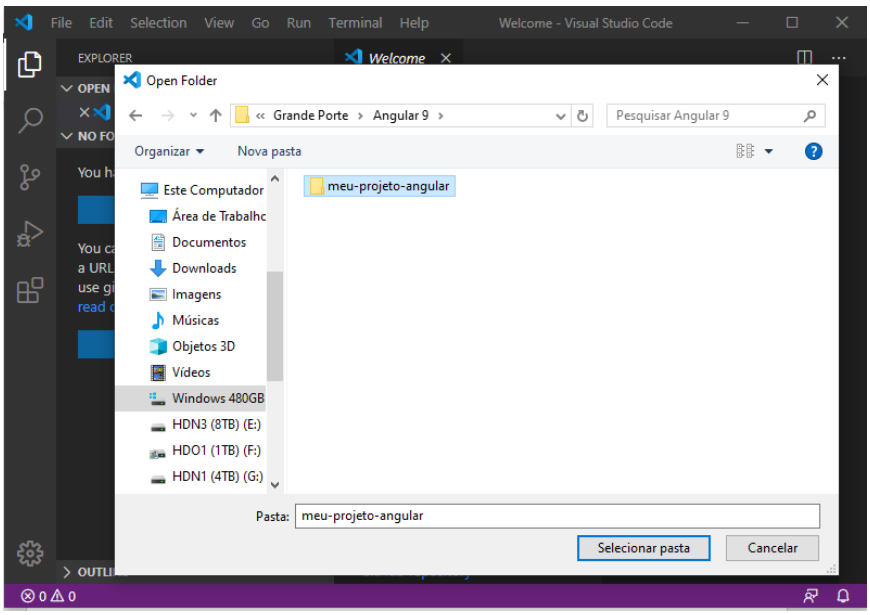

Figura 2 – Abrindo a pasta do projeto no Visual Code Studio.

O projeto estará aberto e você verá a mesma estrutura de pastas, destacado no retângulo vermelho, conforme figura 3.

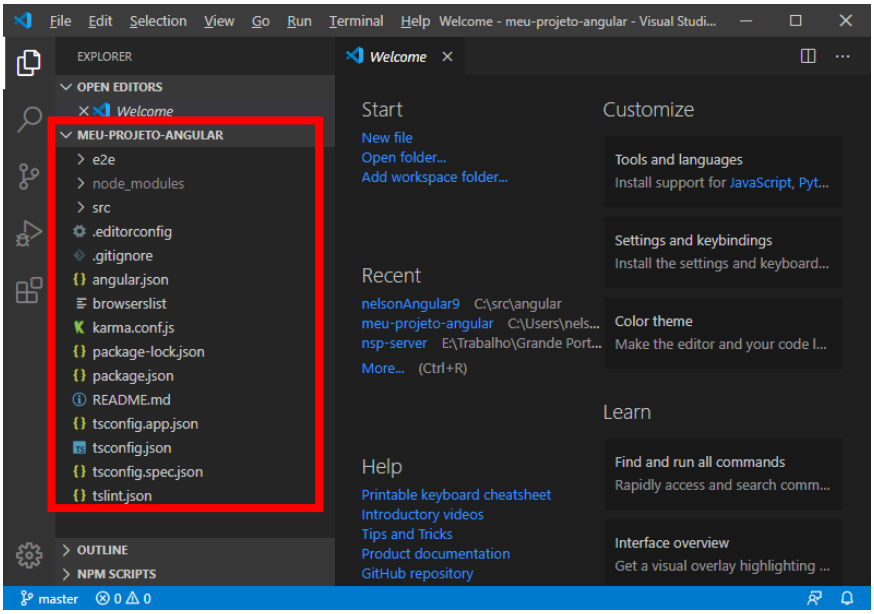

Figura 3 – Projeto Angular 9 aberto no Visual Studio Code.

Vamos agora aos detalhes dessas pastas:

1. Pasta **e2e** (end to end – fim-a-fim) – Contém os arquivos usados para fazer testes automatizados na aplicação, o que garante que ela está funcionando corretamente. Sua estrutura interna pode ser vista na figura 4.

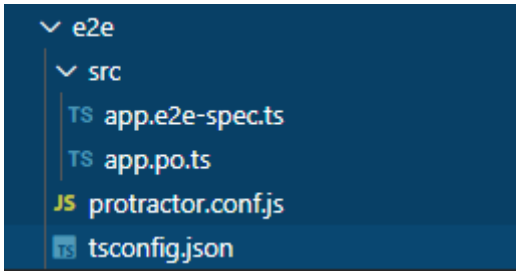

Figura 4 – A estrutura da pasta e2e.

Vale notar que os arquivos com a extensão .ts contém código em typescript e os com a extensão .json contém dados armazenados neste tipo de estrutura (JSON é um acrônimo para JavaScript Object Notation).

2. Pasta **node_modules** – Esta pasta foi criada e é gerenciada pelo npm. Os pacotes instalados pelo npm são node_modules (módulos nodeJS). A figura 5 mostra alguns desses módulos.

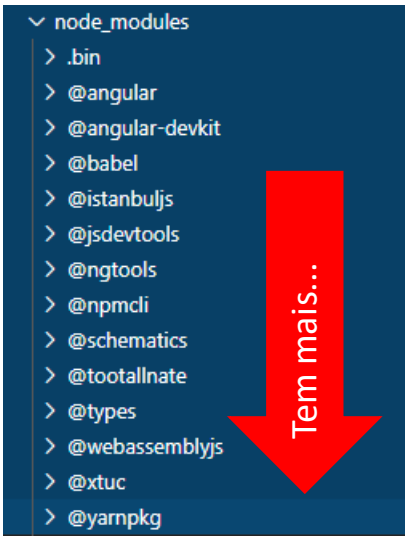

Figura 5 – A estrutura da pasta node_modules.

É nesta pasta onde estão todas as dependências que o Angular 9 precisa para funcionar. Toda vez que adicionamos algo no arquivo package.json, o npm irá gerenciar os pacotes e as suas versões dentro dessa pasta.

O arquivo package.json está na raiz do nosso projeto. Ao clicar duas vezes no nome do arquivo ele será aberto e seu conteúdo pode ser editado, conforme figura 6.

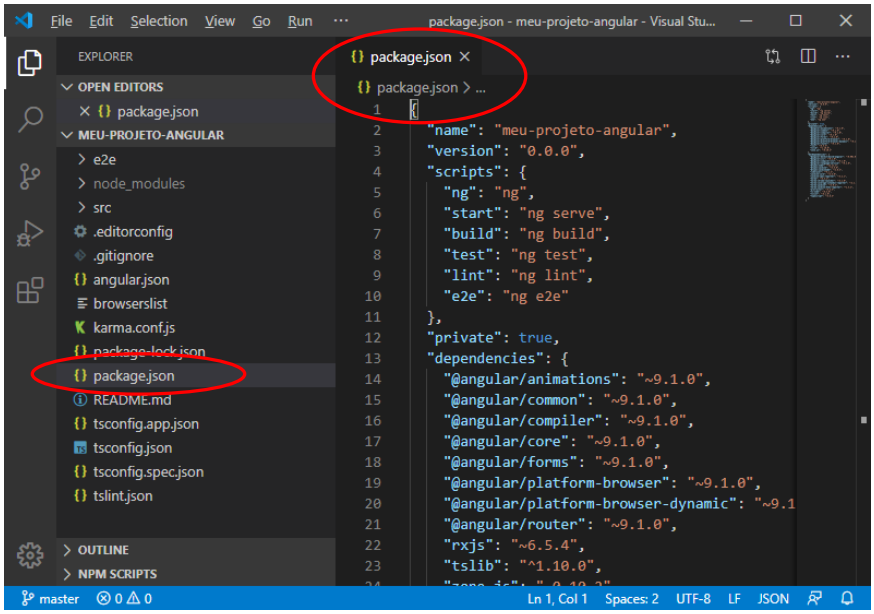

Figura 6 – O arquivo package.json e seu conteúdo.

3. Pasta **src** (source ou fonte) – é a pasta que mais vamos utilizar durante o desenvolvimento de nossa aplicação, pois conterá os código-fonte em diversas linguagens (html, css, typescript etc.) dos componentes e módulos. Ela é formada por diversas outras pastas e arquivos, conforme figura 7.

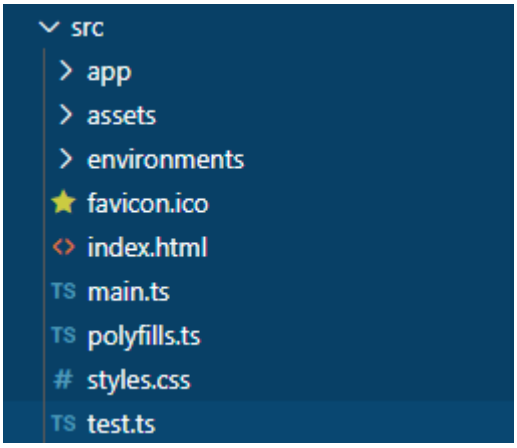

Figura 7 – A pasta src (source).

Vamos olhar com mais detalhe o conteúdo da pasta src:

3.1. Pasta **app** (application ou aplicação) – é onde vamos criar os componentes e módulos de nossa aplicação. Possui diversos arquivos, conforme figura 8.

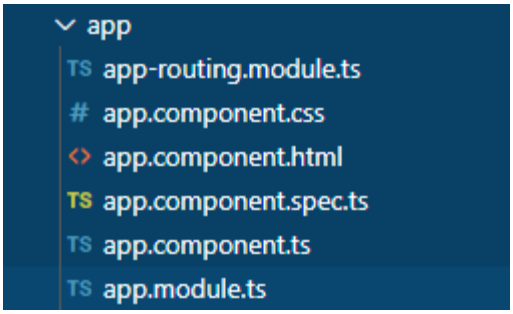

Figura 8 – A pasta app (application).

Os arquivos da pasta app são:

3.1.1. **app-routing.module.ts** – utilizado para declarar uma lista de rotas para os respectivos componentes de nossa aplicação.

3.1.2. **app.component.css** – responsável pelo estilo do módulo. Com o Angular 9 nós trabalhamos com os estilos separados para cada componente, assim conseguimos ter um desacoplamento de estilos. Nesse exemplo nós criamos um projeto com a extensão .css.

3.1.3. **app.component.html** – é o arquivo HTML do nosso componente App, segue o mesmo pensamento que os arquivos de estilo.

3.1.4. **app.component.spec.ts** – é o arquivo de teste do nosso componente.

3.1.5. **app.component.ts** – Em uma arquitetura MVC (Model / View / Controller), esse arquivo teria as mesmas responsabilidades das Controllers.

3.1.6. **app.module.ts** – o Angular é um framework modular, ele precisa de um ou mais módulos para que possamos gerenciar os nossos componentes, esse módulo acaba sendo um default, mas podemos criar outros modules e chamar eles dentro dele.

3.2. Pasta **assets** – Esse diretório nos permite trabalhar com arquivos extras da nossa aplicação, como imagens, sons e vídeos. É uma pasta de acesso público do projeto. A figura 9 mostra que não temos ainda nenhuma imagem na pasta de nosso projeto.

Figura 9 – A pasta assets.

3.3. Pasta **environments** – Aqui nós temos dois arquivos .ts, um para o nosso ambiente de produção e um outro para o nosso ambiente de desenvolvimento. Nele nós adicionamos tarefas como a variável production que vem setada como true em produção e false em desenvolvimento. A figura 10 mostra os arquivos desta pasta.

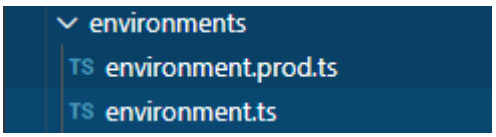

Figura 10 – A pasta environments e seus dois arquivos.

Na sequência temos os demais arquivos da pasta ser.

3.4. Arquivo **favicon.ico** – é o arquivo de ícone de nossa aplicação.

3.5. Arquivo **index.html** – Esse é o arquivo html root, dentro dele rodamos a nossa SPA (Single Page Application).

3.6. Arquivo **main.ts** – Esse é o arquivo principal da nossa Solution. Ele vem definido dentro do arquivo angular-cli.json, esse é o bootstrap da aplicação.

3.7. Arquivo **polyfills.ts** – Esse arquivo funciona como um tradutor entre as versões do typescript. Por exemplo, ao utilizar o ES6 (ECMAscript 6), e o navegador só conseguem entender o ES5 (ECMAscript 5), ele irá interpretar e passar o código corretamente para o navegador.

3.8. Arquivo **styles.css** – Como todos os componentes têm o seu próprio arquivo .css, nós podemos utilizar esse arquivo para criar algo global como variáveis para nossa aplicação.

3.9. Arquivo **test.ts** – Esse arquivo é usado para configurar o ambiente de teste com a ajuda dos softwares Karma e Jasmine.

Na sequência temos os demais arquivos de nosso projeto.

4. Arquivo **angular.json** – contém todas as configurações do Angular 9 para nossa aplicação. Futuramente iremos editar este arquivo.

5. Arquivo **package.json** – como dito anteriormente, contém todas as dependências de nosso projeto. É o npm que faz a leitura desse arquivo e verifica quais são os pacotes que precisam ser baixados do repositório para a aplicação e o faz, automaticamente, colocando dentro da pasta node_modules.

Nesta aula vimos a estrutura de um projeto Angular 9 através do Visual Studio Code. Já na próxima aula vamos utilizar essa estrutura pois criaremos as primeiras peças de nossa aplicação: componentes e módulos.

Vídeo Aula e Exercícios

Esta apostila é referente a vídeo aula #3 e a sequência de exercícios #3 do curso Angular 9. Não deixe de assistir a vídeo aula e fazer os exercícios propostos para esta aula em nossa Plataforma Online.

Sobre o autor

Fabrizio Borelli

Doutorando em Engenharia da Informação pela UFABC

Mestre em Ciência da Computação pela UFABC

Bacharel em Ciência da Computação pela UFABC

Professor Universitário

Empreendedor em TI