



# Angular - Estrutura principal

## O que é o Angular?

Angular é um **framework front-end** criado pelo Google para construir **aplicações web dinâmicas, modernas e escaláveis**.

Ele usa principalmente **TypeScript**, que é uma versão mais segura e estruturada do JavaScript.

---

## Arquitetura Geral

### ◆ Componentes

São as "peças" das telas.

Um componente tem:

- HTML
- CSS
- Lógica (TypeScript)

### ◆ Módulos

Conjunto de componentes. Eles organizam a aplicação.

### ◆ Templates

Onde você escreve o HTML misturado com lógica do Angular.

### ◆ Data Binding

Formas de ligar dados na tela com o código:

- **Interpolation** – `{{ }}`
- **Property Binding** – `[]`
- **Event Binding** – `()`
- **Two-way Binding** – `[]()` (ex: `ngModel`)

## ◆ Services

Arquivos usados para **regras de negócios e comunicação com API**.

## ◆ Injeção de Dependência

Forma do Angular fornecer automaticamente objetos já prontos onde forem necessários.

---

## Estrutura do projeto Angular

Quando você cria um projeto, aparece algo assim:

```
src/
├── app/
│   ├── app.component.ts    (lógica)
│   ├── app.component.html  (template)
│   ├── app.component.css   (estilo)
│   └── app.module.ts       (módulo principal)
├── assets/                  (imagens, etc)
└── main.ts                  (ponto de entrada)
```

---

## Criar o projeto Angular

Digite:

```
ng new nome-do-projeto
```

---

## 4. O Angular vai fazer perguntas no terminal

Ele vai perguntar:

### 1. Você quer adicionar o Angular routing?

digite: **yes**

Routing serve para navegar entre páginas.

## 2. Qual estilo você quer usar?

Escolha **CSS** (por enquanto).

Pronto! O Angular CLI vai criar a estrutura automaticamente.

---

## 5. Entrar na pasta do projeto

```
cd meu-projeto-angular
```

---

## 6. Rodar o projeto

```
ng serve
```

ou

```
ng serve --open
```

 `--open` já abre o navegador automaticamente.

---

## 8. Editar o projeto no VS Code

Na pasta do projeto:

```
code .
```

adicionar em angular.json

```
"@schematics/angular:application": {  
  "strict": true  
}
```

## Estrutura típica de um projeto Angular moderno

Aqui está uma visão geral da estrutura de pastas de um projeto Angular criado com `ng new` :

```
meu-projeto/
├── node_modules/
├── src/
│   ├── app/
│   ├── assets/
│   ├── environments/
│   ├── index.html
│   ├── main.ts
│   ├── styles.css (ou .scss)
│   └── (dependendo da versão) polyfills.ts
├── angular.json
├── package.json
├── tsconfig.json
└── outros arquivos de configuração (ex: tsconfig.app.json ou tsconfig.spec.json)
```

## O que cada pasta e arquivo significa

Agora vamos destrinchar cada parte com o que ela serve.

### 1. `node_modules/`

- Pasta onde ficam todas as bibliotecas que seu projeto usa (Angular, bibliotecas de terceiros etc).
- Você não edita nada aqui normalmente — só o npm/CLI mexe.

### 2. `src/`

- É a pasta raiz do código fonte da sua aplicação.
- Contém tudo que você vai escrever para sua aplicação rodar.

Dentro de `src/` :

#### 2.1 `src/app/`

- É o “núcleo” da sua aplicação.

- Aqui ficam os componentes, serviços, módulos e todo o código Angular que você vai escrever.
- Quando você gera novos componentes, eles vão para dentro desta pasta (ou de subpastas, dependendo da estrutura que você usar).

## 2.2 `src/assets/`

- Arquivos estáticos: imagens, fontes, ícones, JSONs, o que a aplicação precisar para mostrar, mas que não são código Angular.
- Ótimo lugar para guardar “coisas fixas”.

## 2.3 `src/environments/`

- Arquivos para definir variáveis de ambiente diferentes para dev e produção (ex: chaves da API, URLs diferentes).
- Normalmente tem `environment.ts` (para desenvolvimento) e `environment.prod.ts` (para produção). (Medium)
- Isso te ajuda a ter configurações diferentes sem mudar código manualmente quando for “buildar” para produção.

## 2.4 `src/index.html`

- É o HTML base da sua aplicação: a página única (SPA) que vai carregar o Angular. (Medium)
- Dentro desse arquivo vai a tag do seu componente raiz (ex: `<app-root></app-root>` ).

## 2.5 `src/main.ts`

- É o ponto de entrada da aplicação Angular. É onde o Angular “começa a rodar”. (Medium)
- Ele inicializa o bootstrap da aplicação (ou seja, “liga” seu app).

## 2.6 `src/styles.css` (ou `.scss` )

- Estilos globais da aplicação.
- Diferente do CSS dos componentes individuais, esse arquivo afeta toda a aplicação.

## Arquivos importantes fora da pasta `src/`

### `angular.json`

- É a “configuração principal” do Angular CLI para o projeto.
- É aqui que estão definidas as entradas para “main”, onde está o `index.html`, configurações de build, estilos globais, **polyfills** (quando configurados por lá), entre outras coisas. ([Programício](#))
- Também controla qual “sourceRoot” (origem do código) o CLI usa.

### `package.json`

- Lista de dependências (Angular, bibliotecas) e scripts ( `ng serve` , `ng build` , etc).
- Quando você instala algo ( `npm install` ), esse arquivo é atualizado.

### `tsconfig.json`

- Configuração do TypeScript para o projeto. Define como o TS será compilado. ([angular20tutorial.blogspot.com](#))
- Pode ter arquivos adicionais como `tsconfig.app.json` , `tsconfig.spec.json` , dependendo da configuração — esses arquivos dedicados servem para compilar só a aplicação, só os testes, etc. ([angular20tutorial.blogspot.com](#))

Arquivo	Serve para...
<code>app.ts</code>	Componente principal da aplicação (lógica).
<code>app.html</code>	Template HTML do componente.
<code>app.css</code>	Estilos do componente.
<code>app.config.ts</code>	Configurações principais da aplicação.
<code>app.config.server.ts</code>	Configurações para SSR.
<code>app.routes.ts</code>	Rotas principais da aplicação.
<code>app.routes.server.ts</code>	Rotas específicas para SSR.
<code>app.spec.ts</code>	Teste unitário do componente.