



## Criar **Módulos** em um projeto **Angular**:

Um **módulo** no angular nada mais é do que uma abstração lógica (*um código para organizar parte da aplicação*) para **abrigar componentes**, por exemplo, todas as **minhas telas relacionadas a um MODELO** vão ficar no **módulo** desse **MODELO**.

- **ng g m xxxx --routing**: usando o comando “**ng g m \*nomeDoModulo\* --routing**” eu faço a criação de um módulo, esse comando gera um **diretório** com um arquivo de **module.ts**, e um **arquivo de routing.ts (roteamento)** desse módulo:

```
C:\Users\lucas\Desktop\teste\projeto>ng g m cursos --routing
CREATE src/app/cursos/cursos-routing.module.ts (249 bytes)
CREATE src/app/cursos/cursos.module.ts (280 bytes)
```



## Criar **componente** em um projeto **Angular**:

Um **componente** é que qualquer coisa (*uma página, um pedaço de HTML, uma barra lateral, um card, etc...*). Digamos que eu queira uma **tela** que vai exibir os cursos que eu **vou trazer do backend**, essa tela é um **componente**.

- **ng g c xxxx**: o comando “**ng g c cursos/\*nomeDoComponente\***” já gera um **componente dentro do módulo de curso** que foi criado anteriormente, então dá pra usar o terminal para já criar componentes dentro de módulos:

```
C:\Users\lucas\Desktop\teste\projeto>ng g c cursos/listar-cursos
CREATE src/app/cursos/listar-cursos/listar-cursos.component.html (28 bytes)
CREATE src/app/cursos/listar-cursos/listar-cursos.component.spec.ts (669 bytes)
CREATE src/app/cursos/listar-cursos/listar-cursos.component.ts (303 bytes)
CREATE src/app/cursos/listar-cursos/listar-cursos.component.scss (0 bytes)
UPDATE src/app/cursos/cursos.module.ts (390 bytes)
```



Analisando a estrutura de **componentes** e **módulos** em um projeto **Angular**

- **Estrutura do arquivo de módulo:** Se olharmos o arquivo **courses.module.ts** vamos perceber que **ele** declara o **componente listar-cursos**, essa **declaração** indica que **esse componente** faz parte **desse módulo**:

```
projeto > src > app > cursos > cursos.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3
4 import { CursosRoutingModule } from './cursos-routing.module';
5 import { ListarCursosComponent } from './listar-cursos/listar-cursos.component';
6
7
8 @NgModule({
9   declarations: [
10     ListarCursosComponent
11   ],
12   imports: [
13     CommonModule,
14     CursosRoutingModule
15   ]
16 })
17 export class CursosModule { }
```

- **Rota do módulo:** No **arquivo de rotas do módulo (cursos-routing.module.ts)**, eu posso definir em que **rota DO MÓDULO** qual componente eu quero exibir. Então eu posso fazer algo como, no **caminho ""**, eu vou exibir o **componente listar-cursos**:

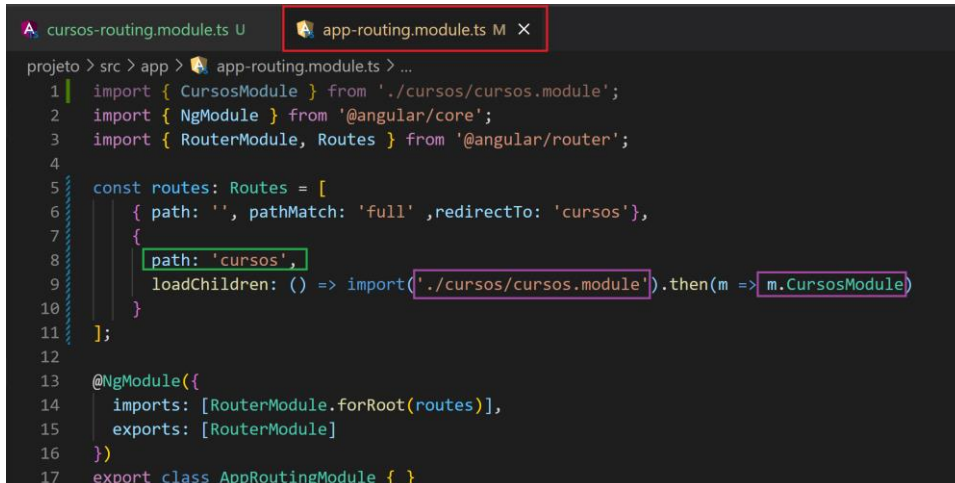
```
projeto > src > app > cursos > cursos-routing.module.ts > CursosRoutingModule
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3
4 import { ListarCursosComponent } from './listar-cursos/listar-cursos.component';
5
6 const routes: Routes = [
7   { path: '', component: ListarCursosComponent }
8 ];
9
10 @NgModule({
11   imports: [RouterModule.forChild(routes)],
12   exports: [RouterModule]
13 })
14 export class CursosRoutingModule { }
```

**\*\*Essa rota é como se**

**fosse uma sub rota desse módulo, então antes de eu conseguir acessa-la, eu preciso primeiro configurar a rota global da aplicação para esse módulo\*\***

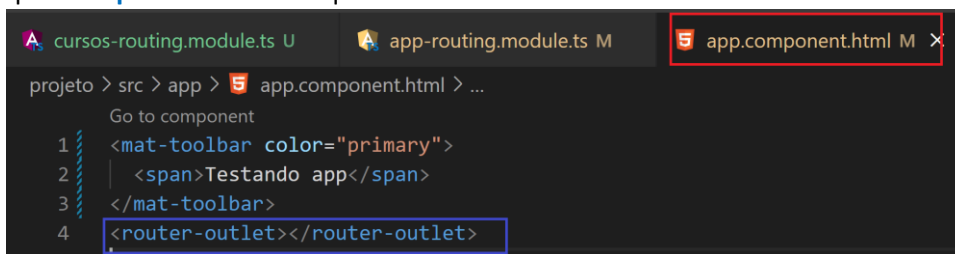
- **Rota global:** No arquivo de rotas **global da aplicação (app-routing.module.ts)** podemos configurar o **caminho do módulo**, nesse caso, eu digo que quando o **caminho "cursos"** for acessado eu vou **carregar o módulo do curso**, e a partir desse **módulo carregado** eu posso

acessar as **subrotas** do módulo para carregar os **componentes**:



```
1 import { CursosModule } from './cursos/cursos.module';
2 import { NgModule } from '@angular/core';
3 import { RouterModule, Routes } from '@angular/router';
4
5 const routes: Routes = [
6   { path: '', pathMatch: 'full', redirectTo: 'cursos' },
7   {
8     path: 'cursos',
9     loadChildren: () => import('./cursos/cursos.module').then(m => m.CursosModule)
10  }
11 ];
12
13 @NgModule({
14   imports: [RouterModule.forRoot(routes)],
15   exports: [RouterModule]
16 })
17 export class AppRoutingModule { }
```

- Para que o Angular saiba qual página ele tem que renderizar, precisamos abrir uma **diretiva** chamada **router-outlet** no **app.component.html**, apenas fazendo isso o Angular entende qual **componente** ele tem que renderizar de acordo com a rota:



```
1 <mat-toolbar color="primary">
2   <span>Testando app</span>
3 </mat-toolbar>
4 <router-outlet></router-outlet>
```



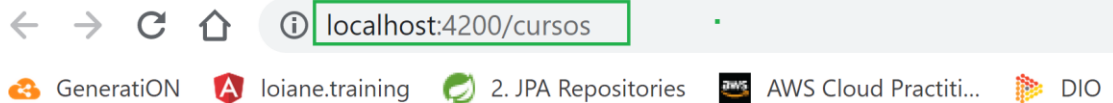
## Fixando o conceito da Rota de maneira simples

A **rota global** quando **tiver vazia** vai me redirecionar para a **rota cursos**, a **rota de cursos** na configuração global vai carregar o **módulo de cursos**, o **módulo** vai então carregar suas rotas que vão exibir seus **componentes**:

```
curso-routing.module.ts U  app-routing.module.ts M  app.component.html M
projeto > src > app > app-routing.module.ts > ...
1 | import { CursosModule } from './cursos/cursos.module';
2 | import { NgModule } from '@angular/core';
3 | import { RouterModule, Routes } from '@angular/router';
4 |
5 | const routes: Routes = [
6 |   { path: '', pathMatch: 'full', redirectTo: 'cursos' },
7 |   {
8 |     path: 'cursos',
9 |     loadChildren: () => import('./cursos/cursos.module').then(m => m.CursosModule)
10 |   }
11 | ];

import { ListarCursosComponent } from './listar-cursos/list
const routes: Routes = [
  { path: '', component: ListarCursosComponent }
];

curso-routing.module.ts U  listar-cursos.component.html U
projeto > src > app > cursos > listar-cursos > listar-cursos.component.htm
Go to component
1 | <p>listar-cursos works!</p>
2 |
```



## Testando app

listar-cursos works!