

Projeto de Interfaces WEB

Rotas e Navegação no Angular Aula 06

Introdução

- Através de rotas, podemos criar URLs para acessar nossas páginas dinamicamente, ou seja, os seletores executados de acordo com o desejo do cliente.
- Inicialmente, vamos preparar o seguinte projeto:
 - **Módulo produto**
 - Componente produto-form
 - Componente produto-pesquisa
 - **Módulo usuario**
 - Componente usuario-form
 - Componente usuario-pesquisa



```
app
├── produto
│   ├── produto-form
│   ├── produto-pesquisa
│   └── produto.module.ts
├── usuario
│   ├── usuario-form
│   ├── usuario-pesquisa
│   └── usuario.module.ts
├── app.component.css
├── app.component.html
├── app.component.spec.ts
├── app.component.ts
└── app.module.ts
```

Criando as Rotas

- Em app.module.ts, crie a seguinte variável e imports:

```
import { Routes } from '@angular/router';
```

```
import { ProdutoPesquisaComponent } from './produto/produto-pesquisa/produto-pesquisa.component';
```

```
import { ProdutoFormComponent } from './produto/produto-form/produto-form.component';
```

```
import { UsuarioPesquisaComponent } from './usuario/usuario-pesquisa/usuario-pesquisa.component';
```

```
import { UsuarioFormComponent } from './usuario/usuario-form/usuario-form.component';
```

Poderia ser qualquer nome.

```
const rotas: Routes = [  
  {path:'produtos',component:ProdutoPesquisaComponent},  
  {path:'produtos/novo',component:ProdutoFormComponent},  
  {path:'usuarios',component:UsuarioPesquisaComponent},  
  {path:'usuarios/novo',component:UsuarioFormComponent},  
]
```

Componente vinculado ao caminho.

Caminhos que iremos digitar no navegador.

Criando as Rotas

- Ainda em `app.module.ts`, importe o `RouterModule`.
- Chame o seu método `forRoot`, passando como parâmetro o vetor de rotas que você criou anteriormente.

```
@NgModule({  
  declarations: [  
    AppComponent  
  ],  
  imports: [  
    BrowserModule,  
    ProdutoModule,  
    UsuarioModule,  
    RouterModule.forRoot(rotas)  
  ],  
  providers: [],  
  bootstrap: [AppComponent]  
})
```

Criando as Rotas

- Finalmente, em `app.component.html`, você deve chamar o seletor:
 - `<router-outlet></router-outlet>`
- Agora, na barra de endereços do navegador, teste as urls descritas no objeto “rotas”, que você definiu em `app.component.ts`

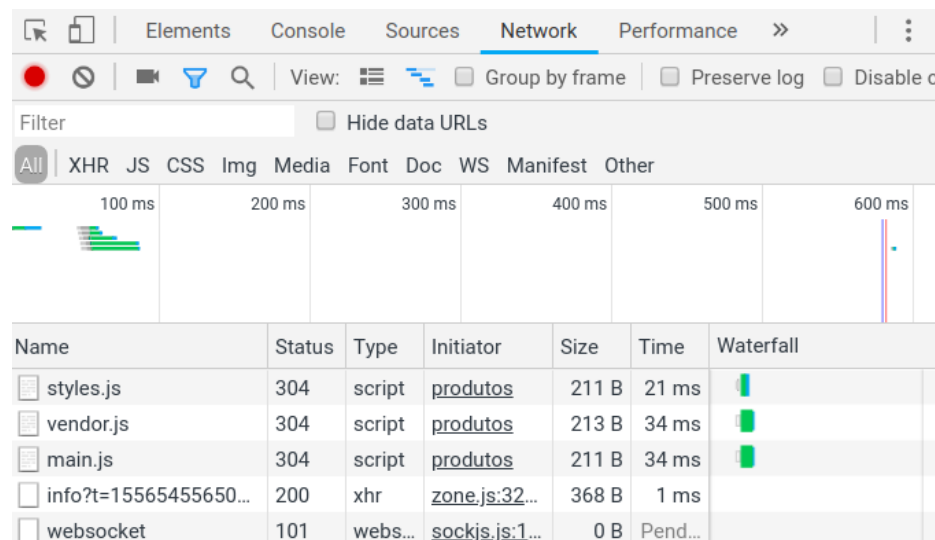
Router Link

- É possível chamar as páginas a partir de links simples, por exemplo, em app.component.html, crie o seguinte código:

```
<a href='produtos'>Listar Produtos</a> <br>  
<a href='produtos/novo'>Adicionar Produto</a> <br>  
<a href='usuarios'>Listar Usuários</a> <br>  
<a href='usuarios/novo'>Adicionar Usuário</a> <br>  
<router-outlet></router-outlet>
```

Router Link

- Apesar de simples, a versão anterior não é ideal pois faz muitos carregamentos a CADA navegação, gerando overhead.
- A aplicação é praticamente REINICIADA a cada clique no link.
- A forma certa é usar o Roteamento do próprio Angular.

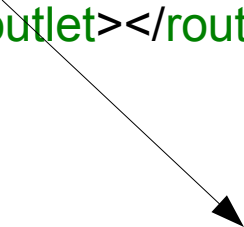


Note que vários arquivos são carregados a cada clique.

Router Link

- Modifique o seu `<a href>` para:

```
<a routerLink='produtos'>Listar Produtos</a> <br>  
<a routerLink='produtos/novo'>Adicionar Produto</a> <br>  
<a routerLink='usuarios'>Listar Usuários</a> <br>  
<a routerLink='usuarios/novo'>Adicionar Usuário</a> <br>  
<router-outlet></router-outlet>
```



Só é possível usar essa diretiva por causa do **RouterModule** importado em `app.module`. Caso você esteja em outro módulo e precise usar o “`routerLink`”, **não esqueça** de importar o **RouterModule** no seu módulo.

Router Link

- Estilizando a rota selecionada, para uma melhor experiência do usuário.
 - As vezes é interessante deixar o link selecionado com algum destaque, para que o usuário saiba exatamente onde está no sistema.
 - Escreva em app.component.html:

```
<a routerLinkActive='ativo' routerLink='produtos' >Listar Produtos</a> <br>  
<a routerLinkActive='ativo' routerLink='produtos/novo'>Adicionar Produto</a> <br>  
<a routerLinkActive='ativo' routerLink='usuarios'>Listar Usuários</a> <br>  
<a routerLinkActive='ativo' routerLink='usuarios/novo'>Adicionar Usuário</a> <br>  
<router-outlet></router-outlet>
```

Router Link

- Crie uma classe css simples, apenas para teste, em app.component.css:

```
.ativo{  
  background-color: magenta;  
}
```

- Teste os links e note que a cor irá mudar para dois links em alguns casos pois os mesmos compartilham o mesmo path inicial. Por exemplo, se clicar em Adicionar Usuário, a cor irá mudar para Listar Usuários também, pois eles compartilha o mesmo path inicial:
 - **/usuarios**/novo
 - **/usuarios**

Router Link

- Caso não seja isso que você deseje, você deve utilizar a propriedade:
 - **[routerLinkActiveOptions]='{exact: true}'**

```
<a routerLinkActive='ativo' routerLink='produtos' [routerLinkActiveOptions]='{exact: true}'>Listar Produtos</a> <br>  
<a routerLinkActive='ativo' routerLink='produtos/novo' [routerLinkActiveOptions]='{exact: true}'>Adicionar Produto</a> <br>  
<a routerLinkActive='ativo' routerLink='usuarios' [routerLinkActiveOptions]='{exact: true}'>Listar Usuários</a> <br>  
<a routerLinkActive='ativo' routerLink='usuarios/novo' [routerLinkActiveOptions]='{exact: true}'>Adicionar Usuário</a> <br>  
<router-outlet></router-outlet>
```

Parâmetros na Rota

- Problema: Ao clicar em um produto listado pela página Listar Produtos, eu quero passar para a página de **edição** daquele produto. Sendo assim, **eu quero passar o ID do produto selecionado para a página de edição.**
- Exercício:
 - Crie um vetor de produtos dentro do componente de listagem de produtos (produto-pesquisa).
 - Crie um outro componente para editar um produto escolhido.

Parâmetros na Rota

- Em produto-pesquisa:

```
export class ProdutoPesquisaComponent {
```

```
  produtos:any[] = [  
    {id:1,nome:"CD"},  
    {id:2,nome:"Livro"},  
    {id:3,nome:"Sapato"},  
    {id:4,nome:"Vestido"},  
    {id:5,nome:"Sabonete"},  
  ];  
  
}
```

Parâmetros na Rota

- Crie um novo componente:
 - `ng g c produto/produto-edit --spec=false;`
- Exporte esse componente no módulo produto. Além disso, crie a seguinte rota:

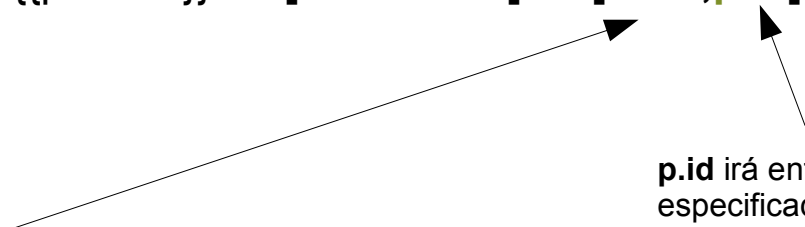
```
const rotas: Routes = [  
  {path:'produtos',component:ProdutoPesquisaComponent},  
  {path:'produtos/novo',component:ProdutoFormComponent},  
  {path:'produtos/edit/:id',component:ProdutoEditComponent},  
  {path:'usuarios',component:UsuarioPesquisaComponent},  
  {path:'usuarios/novo',component:UsuarioFormComponent}  
]
```

→ O “:id” entra como uma variável dentro
Da rota a qual irá receber um valor via
URL.

Parâmetros na Rota

- **path:'produtos/edit/:id',component:ProdutoEditComponent**
 - Essa nova rota redireciona para o componente recém criado de edição.
 - Note que após a URL, temos **:id**, que será substituído pelo valor a ser passado como parâmetro para o componente.
 - Em produto-pesquisa.html, faremos o seguinte para poder passar o valor para id :

```
<h1>Listar Produtos</h1>
<ul *ngFor="let p of produtos">
  <li>ID: {{p.id}} - Nome: {{p.nome}} <a [routerLink] = "['edit',p.id]">EDITAR</a> </li>
</ul>
```



p.id irá entrar como valor do parâmetro “id”, especificado pelo rota em app.module.

Atenção! Como já estamos no módulo produto, não é necessário colocar o caminho todo (**produtos/edit/**). O próprio Angular irá completar com o edit no final.

Parâmetros na Rota

- Em produto-edit.ts:

```
import { Component, OnInit } from
'@angular/core';
import { ActivatedRoute } from
'@angular/router';

@Component({
  selector: 'app-produto-edit',
  templateUrl: './produto-
edit.component.html',
  styleUrls: ['./produto-edit.component.css']
})
export class ProdutoEditComponent
implements OnInit {

  produtos:any[] = [
    {id:1,nome:"CD"},
    {id:2,nome:"Livro"},
    {id:3,nome:"Sapato"},
    {id:4,nome:"Vestido"},
```

Injeta um objeto para
Receber os valores via
url da Rota

```
{id:5,nome:"Sabonete"},
];
```

```
produto = null;
```

```
constructor(private rota:ActivatedRoute) { }
```

```
ngOnInit() {
  let id = this.rota.snapshot.params["id"];
  console.log(id);
```

```
  if(id!=undefined || id!=null){
    for(let p of this.produtos){
      if(p.id==id){
        this.produto = p;
      }
    }
  }
  console.log(this.produto);
}
```

Pegando o valor passado. Note
que o nome dele na rota era id,
o mesmo usado aqui. Já o
nome da variável que recebe
(let id...), não necessariamente
precisa ser o mesmo.

Parâmetros na Rota

- E no caso de mais de um parâmetro?
 - Digamos que queremos passar tanto o **id**, quanto o **nome** para o link de edição.
 - Em `app.module.ts`, iremos modificar a rota da seguinte maneira:

```
{path:'produtos/edit/:id/:nome',component:ProdutoEditComponent},
```

Parâmetros na Rota

- Em produto-pesquisa.html, faremos:

```
<h1>Listar Produtos</h1>
<ul *ngFor="let p of produtos">
  <li>ID: {{p.id}} - Nome: {{p.nome}} <a [routerLink] = "['edit',p.id,p.nome]">EDITAR</a> </li>
</ul>
```

Redirecionamento

- Ao entrar com a url vazia, vamos direcionar para a página de usuario-pesquisa.html

```
const rotas: Routes = [  
  {path:"",redirectTo:'usuarios', pathMatch:'full'},  
  ...
```

- **full**, significa que para ir para “usuarios”, o path deve ser EXATAMENTE o que está descrito, ou seja "" (vazio).
- **prefix**, significa que qualquer caminho que COMECE com path, irá redirecionar para ‘usuarios’. Até mesmo caminhos errados. EVITE.

Página não encontrada

- Crie um componente para a página não encontrada:
 - `ng g c http404 --inline-style --inline-template --flat --spec=false;`
- Crie uma rota:
 - `{path:'http404',component:Http404Component}`
- Crie agora mais a rota abaixo, que redirecione qualquer erro (**) para a rota criada anteriormente:
 - `{path:'**',redirectTo:'http404'}`

Navegação Imperativa

- Problema:
 - Clique em “listar produto”;
 - Clique em “editar”;
 - Ao editar o produto, clique no botão salvar;

Como programar para que ao salvar, o site redirecione para um outra página, como por exemplo, voltar a listar os produtos? A solução é a **navegação imperativa** ou **navegação programática**.

Navegação Imperativa

- Em produto-edit.ts

```
constructor(private rota:ActivatedRoute,  
             private roteador:Router) { }
```

Injete um objeto do tipo Router.

Crie um método “salvar” de teste e nele use o objeto Router para redirecionar para outra página.

```
salvar(nome:string){  
    let id = this.rota.snapshot.params["id"];  
    console.log(`Editando produto de id: ${id}`);  
    this.roteador.navigate(['/produtos']);  
}
```

Navegação Imperativa

- Em produto-edit.html, crie um botão para chamar o método “salvar”.

```
<p>  
produto-edit works!  
<button (click)="salvar('teste')">SALVAR</button>  
</p>
```

Exercício

- Modifique todo o código para que agora dê suporte a um serviço onde os dados são armazenados em JSON.