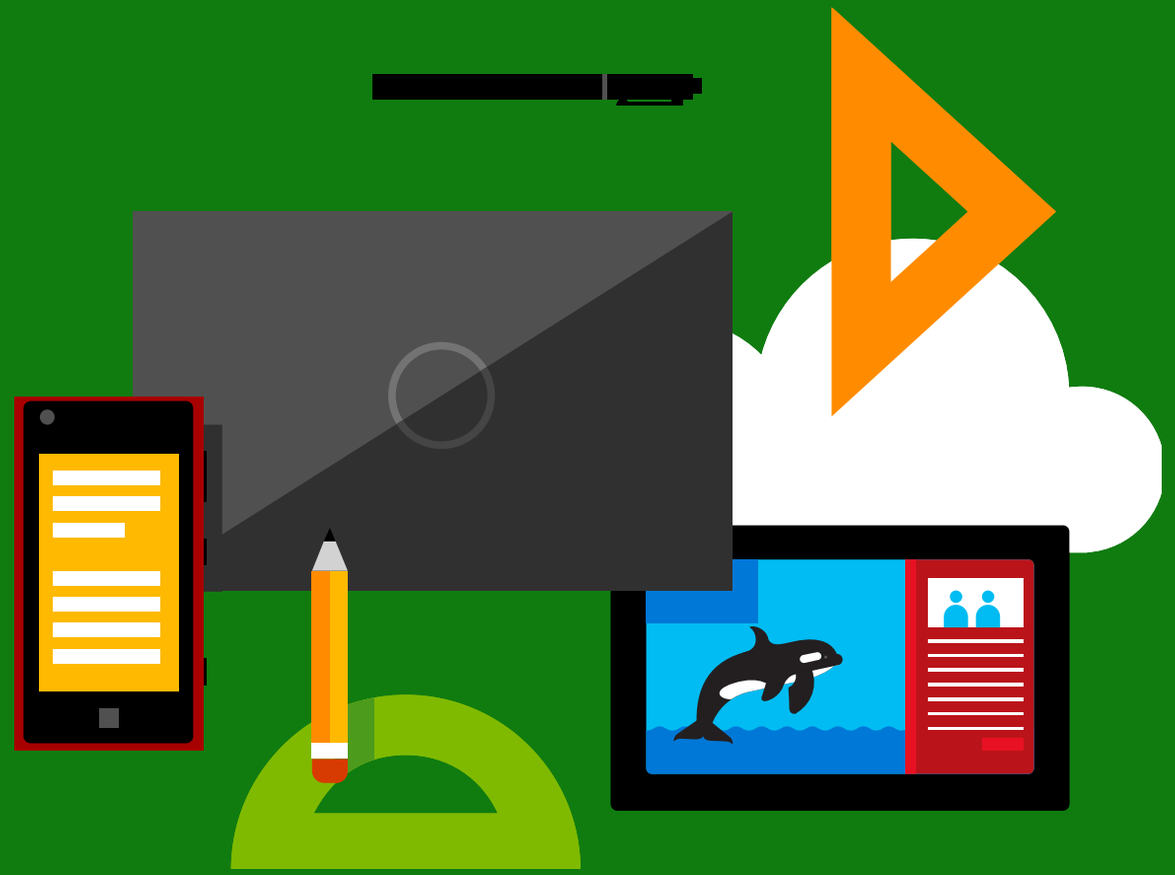
Desenvolvimento Web

Typescript e Angular

Instrutor: Júlio Pereira Machado (julio.machado@pucrs.br)

Angular
e Roteamento



Roteamento

- Angular fornece um roteador para a navegação entre views
  - Objeto *Router* é configurado com vários objetos *Route*
  - Define o mapeamento de URLs para os correspondentes componentes responsáveis pelas *views*
- Configuração:
  - Template básico já inclui a dependência para o pacote *@angular/router*
  - Arquivo “app.config.ts” inicializa o provedor que configura o injetor de dependências para o roteador
  - Arquivo “app.routes.ts” é o ponto de configuração das rotas
- Importante:
  - Rotas são construídas a partir do elemento `<base href="/">` no <head> do arquivo “index.html”
  - Assume, por padrão, que o diretório “app” é a raiz da aplicação
- Documentação: https://angular.dev/guide/routing

Roteamento

- Exemplos: configuração do provedor

```
import { ApplicationConfig } from '@angular/core';
import { provideRouter } from '@angular/router';
import { routes } from './app.routes';

export const appConfig: ApplicationConfig = {
  providers: [provideRouter(routes)]
};
```

Inicializa e configura o roteador na raiz da aplicação Angular

Roteamento

- Exemplos: definição de rotas

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

export const routes: Routes = [
  { path: 'first-component', component: FirstComponent },
  { path: 'second-component', component: SecondComponent },
];
```

Configura as rotas disponíveis para a aplicação Angular

Roteamento

- Diferentes tipos de rotas são suportadas
  - Rotas estáticas
  - Rotas estáticas com passagem de dados (somente de leitura) via propriedade *data* da rota
  - Rotas dinâmicas com parâmetros na URL
  - Rotas dinâmicas com parâmetros via query-strings
  - Rotas de redirecionamento
  - Rotas com URLs definidas por curinga “**”
- A ordem das rotas é importante!
  - Avaliação em ordem da definição do array de objetos *Route*
  - Primeira que padrão “coincidir com a rota” é escolhida
  - Definir rotas mais específicas primeiro e rotas mais gerais depois

Roteamento

- Exemplos: rotas

```
const routes: Routes = [
  { path: 'dashboard', component: DashboardComponent, title: 'Dashboard' },
  { path: 'detail/:id', component: HeroDetailComponent, title: 'Details' },
  { path: 'heroes', component: HeroesListComponent, title: 'Heroes', data:
{ title: 'Heroes' } }
  { path: '', redirectTo: '/dashboard', pathMatch: 'full' },
  { path: '**', component: PageNotFoundComponent }
];
```

Roteamento

- Template da view que utiliza o roteamento define o local onde o componente da view associada à URL irá renderizar o HTML via diretiva **`RouterOutlet`**

```
import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';

@Component({
…
  imports: [RouterOutlet],
})
export class AppComponent {
…
}
```

```
<h1>App</h1>
<router-outlet></router-outlet>
```

Navegação

- A navegação pode se dar:
  - Por uma ação de click em um link
  - Por uma ação de evento com código associado
  - Por uma URL explícita no navegador
- Diretiva **`RouterLink`** é usada como atributo em elementos <a> para definir a URL da view de destino

```
<a routerLink="/heroes">Heroes</a>

<a [routerLink]="['/hero', hero.id]">
  <span class="badge">{{ hero.id }}</span>{{ hero.name }}
</a>

<a [routerLink]="['/crisis-center', { foo: 'foo' }]">Crisis Center</a>
```

Navegação

- Exemplos: navegação via código

```
import { Router, ActivatedRoute, ParamMap } from '@angular/router';
…
export class HeroDetailComponent implements OnInit {
  constructor(
    private route: ActivatedRoute,
    private heroService: HeroService,
    private router: Router,
) {}
  gotoHeroes() {
    this.router.navigate(['/heroes']);
  }
}
```

Navegação

- Serviço **`ActivatedRoute`** possui informações sobre a rota ativa associada à navegação para a view do componente
  - Parâmetros na URL são obtidos via propriedade `paramMap` que retorna um *Observable* contendo um *ParamMap*
  - Parâmetros via query-string obtidos via propriedade `queryParamMap` que retorna um *Observable* contendo um *ParamMap*
  - Propriedade `snapshot` representa o valor da rota em um determinado ponto no tempo
    - Propriedades *paramMap* e *queryParamMap* representam um único dicionário ao invés de um fluxo
  - Parâmetros fornecidos em *data* na rota são obtidos via propriedade `data` que retorna um *Observable*

Navegação

- Exemplos: rota 'detail/:id'

```
import { ActivatedRoute, ParamMap } from '@angular/router';
…
export class HeroDetailComponent implements OnInit {
  constructor(
    private route: ActivatedRoute,
    private service: HeroService
  ) {}
  ngOnInit(): void {
    const heroId = this.route.snapshot.paramMap.get('id');
  }
}
```