Routing & Routeguards

Angular

HOGESCHOOL PXL

Routing & Navigatie

- Applicaties met verschillende views hebben navigatie nodig
- Traditionele manier is met hyperlinks afgehandeld door de browser
- In web apps wordt de navigatie afgehandeld door de app zelf
  - Hangt niet af van de browser functionaliteit
- De component Router in Angular zorgt voor navigatie binnen een applicatie

Routing in Angular

- De Angular Component Router zorgt voor navigatie binnen de applicatie
- Navigatie kan op verschillende manieren:
  - Door het klikken op een (router)link
  - Via de code van een component / serviceprovider / …

Router gebruiken

- Base href in index.html in verplicht

```
<head>
  <meta charset="utf-8">
  <title>RoutingExample</title>
  <base href="/">
```

- Aanmaken van een file voor het definiëren van routes
  - app.routes.ts
- Routes worden gedefinieerd in deze file als variabele
  - Variabele exporteren voor later gebruik

```
export const routes: Routes = [
    {path: 'page1', component: Page1Component}
];
```

Router gebruiken

- De volgende van de routes is belangrijk
- Redirects voorzien kan via de redirectTo property:

```
export const appRoutes: Routes = [
    {
        path: '',
        redirectTo: 'page1',
        pathMatch: 'full'
    },
```

- Opvangen van routes die niet bestaan kan als volgt
  - **Moet als laatste route gedefinieerd worden!**

```
{
    path: '**',
    component: Error404Component
}
```

Router gebruiken

- RouterModule inladen in app.module.ts
- Variabele uit app.route.ts inladen in app.module.ts

```
import { RouterModule } from '@angular/router';
import { appRoutes } from './app.routes';
```

- Routes variabele koppelen aan de RouterModule

```
imports: [
  BrowserModule,
  FormsModule,
  RouterModule.forRoot(appRoutes)
],
```

Router gebruiken

- Via de view:
  - Voorzien van URLs aan de hand van [routerLink]
  - routerLink bevat een array met properties van de route
  - Eerste item is de naam van de route
    - Deze zorgt voor de link naar de app.routes.ts file

```
<li><a [routerLink]="['/']">Home</a></li>
<li><a [routerLink]="['/about']">About</a></li>
<li><a [routerLink]="['/contact']">Contact</a></li>
```

  - Het attribuut routerLinkActive=“cssclass“ kan gebruikt worden om een css klasse toe te voegen aan de hyperlink als de route actief is

Router gebruiken

- Via de code van een Component
  - Importeren van de Router uit @angular/router
  - Injecteren via de constructor
  - Oproepen in bijvoorbeeld een methode

```
constructor(private router: Router) { }

onClick() {
  this.router.navigate(['/contact']);
}
```

Router gebruiken

- Voorzien van een router-outlet
- De router-outlet zorgt voor het renderen van de components die overeenkomen met de route
- Meestal aanwezig in app.component.html

```
<nav>
  <ul>
    <li><a [routerLink]="['/']">Home</a></li>
    <li><a [routerLink]="['/about']">About</a></li>
    <li><a [routerLink]="['/contact']">Contact</a></li>
  </ul>
</nav>
<div class="content">
  <router-outlet></router-outlet>
</div>
```

Data doorgeven met Routes

- Werken met parameters in een route
  - Vaak wordt een id of index van een record doorgegeven in de URL
- Syntax:

```
{ path: 'person/:index',
  component: PersonDetailComponent}
```

- De route wordt in de browser vertaald naar

  http://localhost/person/3

Data doorgeven met Routes

- Navigeren naar routes met parameters kan via:
  - Het [routerLink] attribuut vanuit de view

```
<ul>
  <li *ngFor="let x of list; let i=index">
    <a [routerLink]="['/person',i]">{{x.name}}</a>
  </li>
</ul>
```

  - De router.navigate methode vanuit de controller

```
onClick() {
  this.router.navigate(['/person', this.index]);
}
```

Data doorgeven met Routes

- Data opvangen in de component
  - Injecteren van ActivatedRoute
  - Data ophalen:
    - Synchroon:

```
constructor(private route: ActivatedRoute, private peopleService: PeopleService) { }

ngOnInit() {
  let index = this.route.snapshot.params['index'];
  this.person = this.peopleService.getPerson(index);
}
```

Data doorgeven met Routes

- Data opvangen in de component
  - Injecteren van ActivatedRoute
  - Data ophalen:
    - Asynchroon:

```
ngOnInit() {
  this.route.params.subscribe(params => {
    let index = +params['index'];
    this.person = this.peopleService.getPerson(index);
  });
}
```

Data doorgeven met Routes

- Meerdere parameters aan één route is ook mogelijk. In de route variabele:

```
{
    path: 'page5/:par1/:par2',
    component: Page5Component,
    canDeactivate: [CanDeactivateConfirmGuard]
},
```

- In de View om een link te leggen:

```
<li><a [routerLink]="['/page4']" routerLinkActive="active">Page 4 (activator)<
<li><a [routerLink]="['/page5',123,456]" routerLinkActive="active">Page 5</a>
```

- De parameters opvragen kan terug met router.snapshot.params[‘paramNaam’]

Routeguards

- Routeguards zijn methodes die opgeroepen worden op bepaalde momenten in de levencyclus van de route
  - canActivate
  - canDeactivate
  - canLoad
  - canActivateChild
- Moeten gedefinieerd worden bij de providers in app.module.ts
- Moeten de @Injectable() decorator bevatten

Routeguards: canActivate

- Wordt geladen voor de activatie van de route (= voor het renderen van de component)
- Klasse die canActivate implementeert
- Bevat:
  - Een constructor: Dependency injection
  - Een methode canActivate:
    - Komt van de canActivate interface
    - Wordt uitgevoerd bij het uitvoeren van de routeguard
    - Geeft true of false terug

Routeguards: canActivate

- auth-activator.service.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, Router } from '@angular/router';
import { AuthService } from '../services/auth.service';

@Injectable()
export class CanActivateViaAuthGuard implements CanActivate {

    constructor(private authService: AuthService, private router: Router) { }

    canActivate() {
        console.log(this.authService.isLoggedIn());
        if (!this.authService.isLoggedIn()) {
            this.router.navigate(['/login']);
        }
        return this.authService.isLoggedIn();
    }
}
```

Routeguards: canActivate

- Routeguard wordt toegevoegd aan de app.module.ts file:

```
@NgModule({
  declarations: [···
  ],
  imports: [···
  ],
  providers: [AuthService, CanActivateViaAuthGuard],
  bootstrap: [AppComponent]
})
```

- Routeguard wordt toegevoegd aan de app routes:

```
{
    path: 'page4',
    component: Page4Component,
    canActivate: [CanActivateViaAuthGuard]
},
```

- canActivate is een array, meerdere canActivate routeguards zijn dus mogelijk

Routeguards: canDeactivate

- Wordt geladen voor het deactiveren van de huidige route (= voor het verlaten van de component)
- Klasse die canDeactivate implementeert
- Bevat:
  - Een constructor: Dependency injection
  - Een methode canDeactivate:
    - Komt van de canDeactivate interface
    - Wordt uitgevoerd bij het uitvoeren van de routeguard
    - Geeft true of false terug

Routeguards: canDeactivate

- confirm-deactivator.service.ts

```
import { Injectable } from '@angular/core';
import { CanDeactivate, ActivatedRouteSnapshot, RouterStateSnapshot } from '@angular/router';
import { Observable } from 'rxjs/Observable';
import { Page5Component } from '../page5/page5.component';

@Injectable()
export class CanDeactivateConfirmGuard implements CanDeactivate<Page5Component> {
    canDeactivate(component: Page5Component) {
        return window.confirm('Are you sure you want to leave?');
    }
}
```

Routeguards: canDeactivate

- Routeguard wordt toegevoegd aan de app.module.ts file
- Routeguard wordt toegevoegd aan de app routes:

```
{
    path: 'page5',
    component: Page5Component,
    canDeactivate: [CanDeactivateConfirmGuard]
},
```

- canDeactivate is een array, meerdere canDeactivate routeguards zijn dus mogelijk

Routeguards: canDeactivate - parameters

- Verschillende zaken kunnen meegegeven worden aan de canDeactivate methode:
  - De component
  - De route parameters
  - De route url
- Hierdoor is het mogelijk om bijvoorbeeld een deactivator te linken aan content in de component:

```
@Injectable()
export class CanDeactivateCustomGuard implements CanDeactivate<Page5Component> {
    canDeactivate(component: Page5Component, currentRoute: ActivatedRouteSnapshot,
                  currentState: RouterStateSnapshot, nextState?: RouterStateSnapshot) {
        return component.compMethod();
    }
}
```

Resolvers

- Kunnen gebruikt worden om data op te halen bij het activeren van de route in plaats van na het renderen van de component (ngOnInit)
- Foutafhandeling voor het tonen van de component
  - Denk bv aan een foute id bij een detailcomponent

```
import { Injectable } from '@angular/core';
import { Resolve, ActivatedRouteSnapshot } from '@angular/router';
import { ContactsService } from './contacts.service';

@Injectable()
export class ContactResolve implements Resolve<Contact> {

  constructor(private contactsService: ContactsService) {}

  resolve(route: ActivatedRouteSnapshot) {
    return this.contactsService.getContact(route.paramMap.get('id'));
  }
}
```

Resolvers

- Worden ook toegevoegd aan de providers in app.module.ts
- In de components kan de data uit de activatedRoute gehaald worden

```
export const AppRoutes: Routes = [
  ...
  {
    path: 'contact/:id',
    component: ContactsDetailComponent,
    resolve: {
      contact: ContactResolve
    }
  }
];
```

Routing als aparte module

- Zie CH9-Voorbeeld2

- Best practise om routing te implementeren
  - Handig bij complexe routing bij applicaties die uit meerdere modules bestaan
- Zelf opbouwen ofwel gebruik maken van de AngularCLI

```
PS C:\Users\Dries\workspace> ng new testrouting
? Would you like to add Angular routing? (y/N)
```

Routing als aparte module

- Er wordt een aparte module ‘app-routing.module.ts’ gemaakt waarin routing gedefinieerd wordt:

```
const routes: Routes = [···

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule],
  providers: [CanActivateViaAuthGuard, CanDeactivateConfirmGuard]
})
export class AppRoutingModule { }
```

- Koppeling doen we in de app.module.ts file:

```
@NgModule({
  declarations: [···
  imports: [
    BrowserModule,
    FormsModule,
    AppRoutingModule
  ],
  providers: [AuthService],
```