Web Frameworks
Angular – deel 6

Elektronica – ICT
Sven Mariën
(sven.marien01@ap.be)

AP HOGESCHOOL ANTWERPEN

AP.BE

Inhoud

1. Deel 1
    1. Waarom Angular ?
    2. Nieuw Angular project
        - Aanmaken
        - Bestandsstructuur
    3. Opbouw
        - Van een Applicatie
        - Van een Component
    4. Nieuwe Component Aanmaken
2. Deel 2
    1. Interpolation
    2. Pipes
3. Deel 3
    1. Directives
4. Deel 4
    1. Property binding
    2. Event binding
5. Deel 5
    1. Two-way binding
    2. Parent-child componenten
6. **Deel 6**
    1. **Routing**
    2. Modules

Navigatie => Routing

- Een SPA bestaat weliswaar uit 1 pagina, maar kan daarin wel meerdere ‘views’ weergeven.
  - Angular laat ons toe om hiertussen te **navigeren via ‘routing’**
- **Via de URL** kan er dan navigatie plaatsvinden
  - Bv. http://localhost:4200/#/game
- In tegenstelling tot een klassieke website
  - zal er **geen** HTTP request naar de server gaan
  - Maar zal deze lokaal door het framework worden afgehandeld.
- Hiervoor moeten we gebruik maken van de **RouterModule**
  - En hierop de lijst van URL’s instellen.
  - Deze zal een **URL mappen** naar een bepaalde **Component** en deze weergeven.
  - De ‘output’ van de Routing wordt weergegeven in de speciale tag: **<router-outlet>**
- Routing zal ook werken met de ‘**backward**’ en ‘**forward**’ knoppen van de browser.

Routing (2)

- Voeg de **RouterModule** toe aan de **AppModule**
  - bij **imports**
- Initialiseer de routing via de **forRoot()** methode
  - Deze verwacht **een lijst van "routes"**
  - Een route bestaat uit een **path** (string) en eer **component** (klassenaam)
  - De routes worden afgehandeld:
    - Van boven naar onder
    - De eerste die "matched" zal worden gebruikt
  - Zet de 2e parameter: { useHash: true } om te voorkomen dat er server calls gebeuren
- Geef aan waar het **routing 'resultaat'** moet worden getoond
  - Mbv. <router-outlet></router-outlet>

```
imports: [
  BrowserModule,
  RouterModule.forRoot([
    { path: "game", component: GameComponent},
    { path: "game/:id", component: GameComponent},
    { path: "home", component: HomeComponent},
    { path: "", redirectTo:"home", pathMatch: 'full'},
    { path: "**", component: PageNotFoundComponent}
], {useHash: true}),
```

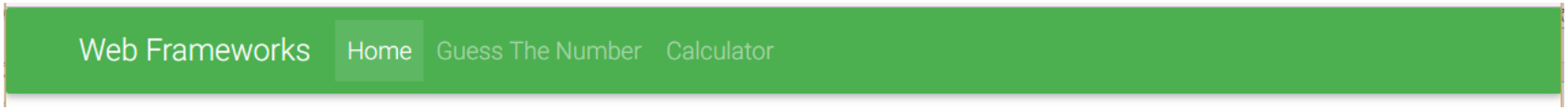
Routing (3)

- Voorbeeld van een routinglijst (let op het '#' teken):
  - http://localhost:4200/#/game => GameComponent
  - http://localhost:4200/#/game/10 => GameComponent (id = 10)
  - http://localhost:4200/#/home => HomeComponent
  - http://localhost:4200 => HomeComponent
  - http://localhost:4200/#/calculator => PageNotFoundComponent
  - http://localhost:4200/#/xyz => PageNotFoundComponent

```
imports: [
  BrowserModule,
  RouterModule.forRoot([
    { path: "game", component: GameComponent},
    { path: "game/:id", component: GameComponent},
    { path: "home", component: HomeComponent},
    { path: "", redirectTo:"home", pathMatch: 'full'},
    { path: "**", component: PageNotFoundComponent}
], {useHash: true}),
```

Routing (4)

- We willen uiteraard ook kunnen navigeren vanuit onze toepassing zelf. Hiervoor kunnen we een navigatiebalk toevoegen:
  - Dit voorbeeld werkt met een bootstrap balk, maar je kan uiteraard eender wat gebruiken'.



nav.component.html

```
<mdb-navbar SideClass="navbar navbar-expand-lg navbar-dark green">
    <logo>
        <a href="" class="navbar-brand">Web Frameworks</a>
    </logo>
    <links>
        <ul class="navbar-nav">
            <li class="nav-item active">
                <a class="nav-link waves-light" mdbRippleRadius>Home
                    <span class="sr-only">(current)</span>
                </a>
            </li>
            <li class="nav-item">
                <a class="nav-link waves-light disabled" mdbRippleRadius>Guess The Number</a>
            </li>
            <li class="nav-item">
                <a class="nav-link waves-light disabled" mdbRippleRadius>Calculator</a>
            </li>
        </ul>
    </links>
</mdb-navbar>
```

Routing (5)

- Om de knoppen in de navigatiebalk te laten werken met de **RouterModule** moeten we gebruik maken van:
  - De **[routerLink]** directive
  - Let op de **syntax**:
    - De parameter is een **lijst** [ ]
    - En de route is een **string**
- Om de huidige route te laten weergeven gebruiken we:
  - De **[routerLinkActive]** directive

```
<mdb-navbar SideClass="navbar navbar-expand-lg navbar-dark green">
    <logo>
        <a href="" class="navbar-brand">Web Frameworks</a>
    </logo>
    <links>
        <ul class="navbar-nav">
            <li class="nav-item" [routerLinkActive]="['active']">
                <a [routerLink]="['/home']" class="nav-link waves-light" mdbRippleRadius>Home
                    <span class="sr-only">(current)</span>
                </a>
            </li>
            <li class="nav-item" [routerLinkActive]="['active']">
                <a [routerLink]="['/game']" class="nav-link waves-light" mdbRippleRadius>Guess The N
            </li>
            <li class="nav-item" [routerLinkActive]="['active']">
                <a [routerLink]="['/calc']" class="nav-link waves-light" mdbRippleRadius>Calculator<
            </li>
        </ul>
    </links>
</mdb-navbar>
```

Oefening: Routing

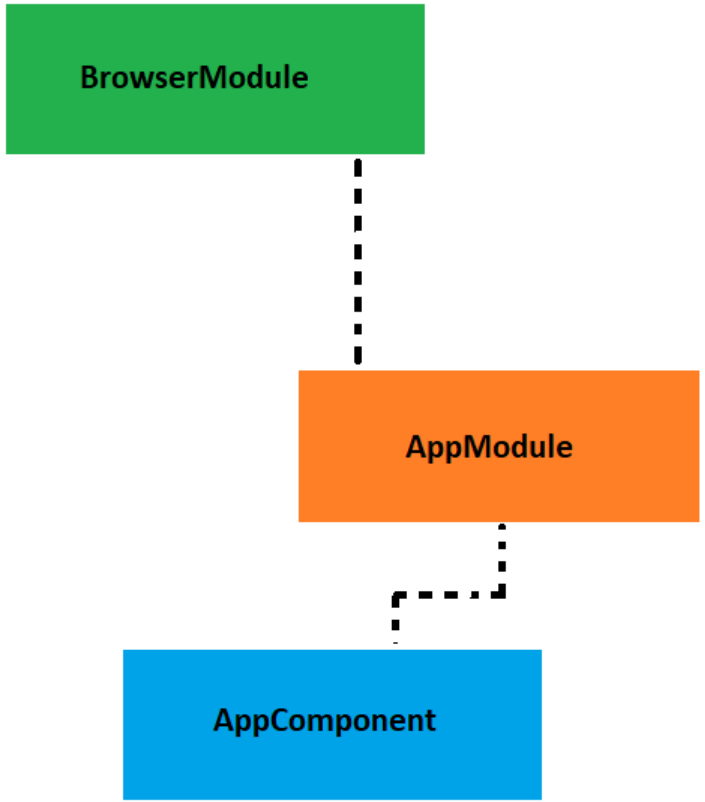
- Voeg Routing toe aan je applicatie
- Stel routes in:
  - /home : Toon de HomePageComponent
  - /page1: Toon de Page1Component
    - Met daarin een css grid
    - Met meerdere componenten
  - Alle andere routes verwijzen naar de /home route

Inhoud

1. Deel 1
    1. Waarom Angular ?
    2. Nieuw Angular project
        - Aanmaken
        - Bestandsstructuur
    3. Opbouw
        - Van een Applicatie
        - Van een Component
    4. Nieuwe Component Aanmaken
2. Deel 2
    1. Interpolation
    2. Pipes
3. Deel 3
    1. Directives
4. Deel 4
    1. Property binding
    2. Event binding
5. Deel 5
    1. Two-way binding
    2. Parent-child componenten
6. **Deel 6**
    1. Routing
    2. **Modules**

Angular Modules

- Een nieuw Angular project bevat:
  - Startup Module: **AppModule**
  - Startup Component: **AppComponent**
  - Externe modules:
    - **BrowserModule**: bevat de nodige functionaliteit zodat de toepassing in de browser kan draaien (DOM)



Angular Modules

- Een Angular Module:
  - Is eveneens een klasse met keyword ‘export’
  - De afspraak is dat de naam van de klasse eindigt met …Module
  - De klasse krijgt een decorator : @NgModule om aan te geven dat het een Module is
  - De decorator bevat parameters:
    - **declarations**: dit is een lijst van componenten die onder deze module worden geplaatst
    - **imports**: Lijst van gebruikte & externe Modules
      - BrowserModule = standaard (Angular in Browser)
    - **Exports:** Lijst van componenten, modules, pipes die je ter beschikking wil stellen van andere modules
    - **providers**: zie later
    - **bootstrap**: De ‘main’ (opstart) module van het project

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Angular Modules, nieuwe componenten aanmaken

- Om een nieuwe Component aan te maken zijn er dus 3 acties nodig:
  1. Aanmaken van de component zelf:
     - xxx.**component.ts bestan**d met klasse/decorator
     - xxx.component.html bestand met HTML template
  2. **Declaratie** van de component in **1 Angular module**
  3. Aangeven waar de component dient te worden weergegeven
     - via de **selector**

AP HOGESCHOOL
ANTWERPEN