Geavanceerde Technieken voor Webapplicaties

Angular

Elke Steegmans

Angular

- is a **complete JavaScript-based open-source front-end web application framework** mainly maintained by **Google** and by a community of individuals and corporations to address many of the challenges encountered in developing **single-page applications**

Versions

- AngularJS 1

- **Angular 2**

AngularJS 1

- is a **hip JavaScript framework**
- is made for building large, **single-page** web applications
- is **client-side**
- made in **2012**
- a framework for client-side model–view–controller (**MVC**) and model–view–view-model (**MVVM**) architectures

Angular 2

- is not a version upgrade, but a **complete rewrite**
- announced in **2014**
- is a **framework** for building client applications in **HTML** and either **JavaScript** or a language like **TypeScript** that compiles to JavaScript
- => called Angular from now on in the slides

TypeScript

- Angular is mostly used in combination with TypeScript
  - is a superset of JavaScript
    - a real OO JavaScript :-)
  - TS code is compiled to JS code
- Angular can also be used in combination of JavaScript

Current version …

- Angular 5.2 is out …
    - is backwards compatible with 2.x.x for most applications
- If you want to read more about it
    - https://blog.angular.io/angular-5-2-now-available-312d1099bd81
- Angular 7 => coming september 2018 :-)

Angular

- Angular applications are made up of **components**
- A component is the combination of an **HTML template** and a **component class** that controls a portion of the screen

Hello Example

app.component.ts

```
import { Component } from '@angular/core';
@Component({
  selector: 'my-app',
  template: `<h1>Hello {{name}}</h1>`
})
export class AppComponent { name = 'Angular'; }
```

index.html

```
<my-app>Loading …</my-app>
```

Setting up a local development environment

- Follow the instructions on
  - https://angular.io/docs/ts/latest/guide/setup.html

Steps

- Install node and nam
  - https://docs.npmjs.com/getting-started/installing-node
- Install npm packages (npm install)
- Run npm start to launch the sample application (npm start)

Node.js and npm

- Node.js and npm are essential to modern web development with Angular and other platforms.
- Node powers client development and build tools.
- The npm package manager, itself a node application, installs JavaScript libraries.

Checking version ...

- node -v
- npm -v

Heroes Example

- download the starting code
- npm install
- npm start

Heroes Example
Step 1

- Add a title property for the app name and a hero property for a hero named "Windstorm"

Interpolation Binding

- The **double curly braces** are Angular's interpolation binding syntax.
    - These **interpolation bindings** present the component's title and hero property values, as strings, inside the HTML header tags.

- Also known as **one-way binding**

Heroes Example
Step 2

- The hero needs more properties: a name and an id. Use a class for it and create an hero with name “WindStorm” and id 1.

Classes and objects

- In TypeScript you can make **classes** and **objects** …

Heroes Example
Step 3

- To show all of the hero's properties, add a for the hero's id property and another for the hero's name. To keep the template readable, place each on its own line.

Template literals

- The **backticks** around the component template let you put the <h1>, <h2>, and elements on their own lines, thanks to the **template literals** feature in ES2015 and TypeScript.

Heroes Example
Step 4

- Users should be able to edit the hero name in an <input> text box. The text box should both display the hero's name property and update that property as the user types.

- You need a two-way binding between the <input> form element and the hero.name property.

Two-way Binding

- **[(ngModel)]** is the Angular syntax to bind the hero.name property to the textbox. Data flows in both directions: from the property to the textbox, and from the textbox back to the property = **two-way binding**.

- Although NgModel is a valid Angular directive, it isn't available by default. It belongs to the optional **FormsModule**. You must opt-in to using that module.

Heroes Example
Step 5

- Create an array of 10 heroes and show all the details of these 10 heroes.

ngFor

- The built-in directive *ngFor
  - The (*) prefix to ngFor is a critical part of this syntax. It indicates that the <li> element and its children constitute a master template.
  - The ngFor directive iterates over the component's heroes array and renders an instance of this template for each hero in that array.

Building blocks of an Angular Application

- Module
- Component
- Directive
- Data binding

Module

- is a container for a group of related components, services, directives, and so on

- @NgModule

- all apps must have at least a root module that is bootstrapped during the app launch

AppModule

- Every component must be declared in one—and only one—Angular module.
- AppModule
  - import
  - declarations

Component

- is the main building block of an Angular application
- each component consists of 2 parts
  - a view the defines user interface
  - a class that implements the logic behind the view
- @Component
- each app must have at least one component called the root component

Component

- each @Component must have
  - selector: is similar to a CSS selector
  - template: contains HTML markup

Directives

- allows you to attach custom behaviour to an HTML element
    - ngFor for example

Data Binding

- allows you to keep a component’s properties in sync with the view

References

- https://angular.io