ng commands

Create new angular project

Creates a new Angular project.

> *ng new project <projectname>*

Start development server

Start Angular live devolopment server.

> *ng serve*

ng generate component

Create a full component in src folder

> *ng generate component <name>*

oder

> *ng g c <name>*

npm commands

Install bootstrap

Install newest version of bootstrap

> *npm install --save bootstrap*

Fix severity vulnerabilities

Fix severity vulnerabilities in project.

> *npm audit fix*

manual project changes

add bootstrap to new project

Add "./node_modules/bootstrap/dist/css/bootstrap.min.css" line in .angular-cli.json file in app.styles array.

> see example 1.1

angular

1 Getting started

Setup Development Environment

```
1. Get newest NodeJs from nodejs.org
2. run _npm install -g npm_
3. run _npm uninstall -g angular/cli_
4. run _npm cache clean_
5. run _npm install -g @angular/cli_
```

2 Basics

Component

Databinding: ngModel

> *<input type="text" [(ngModel)]="name">*

It accepts a domain model as an optional Input. If you have a one-way binding to ngModel with [] syntax, changing the value of the domain model in the component class sets the value in the view. If you have a two-way binding with [()] syntax (also known as 'banana-box syntax'), the value in the UI always syncs back to the domain model in your class.

Databinding: string interpolation

One way databinding from model to view

> *{{propertyName}}*

Directive: ngIf

> *<div *ngIf="condition">Content to render when condition is true. *

Directive: ngFor

Example 1

> *<app-server *ngFor="let server of servers"></app-server>*

Example 2

> <div
>
> > *ngFor="let logItem of log; let i = index" [ngStyle]="{backgroundColor: i >= 4 ? 'blue' : 'transparent'}" [ngClass]="{'white-text': i >= 4}" >{{ logItem }}

Directive: ngClass

> <p

```
[ngClass]="{online: serverStatus === 'online'}"> {{ 'Server' }} with ID {{ serverId }} is {{ getServerStatus() }} </p>
```

Directive: ngStyle

```
<p
    [ngStyle]="{backgroundColor: getColor()}" {{ 'Server' }} with ID {{ serverId }} is {{ getServerStatus() }} </p>
```

3 Course Project Basics

4 Debugging

Use Chrome Debugging Tools

Open Chrome debugging tools after by pressing F12.

Use SourceMaps

Angular CLI adds SourceMaps to Javascript files when it sets up bundles for the browser to get an reference between JavaScript files and TypeScript files. Only available in development mode. They are not provided in production mode.

Access TypeScript files:

```
Chrome-> F12 -> Sources -> top -> webpack -> . -> src -> app
```

Here you find your TypeScript file like in your dev environment.

Use Augury

Augury is a chrome extension to debbug your Angular app. You can see your Router, Components and Models. Helps you understand and analyse your Angular app at runtime.

5 Databinding: Components & Databinding Deep Dive

Component life cycle

Event	Description
ngOnChanges	Called after a bound input property changes
ngOnInit	Called once the component is initialized
ngDoCheck	Called during every change detection run
ngAfterContentInit	Called after content (ng-content) has been projected into view
ngAfterContentChecked	Called every time the projected content has been checked
ngAfterViewInit	Called after the component's view (and child views) has been initialized
ngAfterViewChecked	Called every time the view (and child views) have been checked

Event	Description
ngOnDestroy	Called once the component is about to be destroyed

6 Databinding: Course Project - Components & Databinding

7 Directives Deep Dive

8 Course Project - Directives

9 Using Services & Dependency Injection

10 Course Project - Services & Dependency Injection

11 Changing Pages with Routing

12 Course Project - Routing

13 Understanding Observables

14 Course Project Observables

15 Handling Forms in Angular Apps

16 Course Project - Forms

17 Using Pipes to Transform Output

18 Making Http Requests

19 Course Project - Http

20 Authentication & Route Protection in Angular

21 Dynamic Components

22 Angular Modules & Optimizing Angular Apps

23 Deploying an Angular App

24 Bonus: Working with ngRx in our Project

25 Bonus: Angular Universal

26 Angular Animations

27 Adding Offline Capabilities with Service Workers

28 A Basic Introduction to Unit Testing in Angular Apps

29 Angular Changes & New Features

30 Course Roundup

31 Custom Project & Workflow Setup

32 Bonus: TypeScript Introduction (for Angular 2 Usage)