Angular

Angular - Power Workshop for Developers


One framework. Mobile & desktop.
— Angular - angular.io

AngularJS versus Angular

	AngularJS	Angular
Website	https://angularjs.org/	https://angular.io/
Created in	2009	2016
Version	1.7.8	8.0.1
Language	JavaScript	TypeScript
Dependencies	jQuery (*optional*)	core-js (*optional*) / Zone.js / RxJS

Design goals

HTML	TypeScript / JavaScript	CSS
Structure	Event-Handling	Layout
Content	Business-Logic	Design
Templating	Data-Binding	

- Modular structure
- Strictly defined building blocks
- Strictly defined data flow
- Platform independence through abstraction

Platforms

- Browser
- Desktop
- Server
- Native Mobile
- WebWorker

Getting started

Demo-App

1. Backend: Blinkt! LEDs on Raspberry Pi
2. Middleware: REST API based on Node.js / TypeScript
3. Frontend: will be coded as Angular app during the workshop

Tools

1. CLI tool for Angular
2. webpack
3. Angular Augury
4. Angular Language Service
5. StackBlitz
6. Angular Console

Conventions: Style Guides

1. Style Guide
2. TSLint
3. Prettier
4. codelyzer

Create a first Angular application

The Angular CLI is a tool to initialize, develop, scaffold and maintain Angular applications

— Angular CLI

Angular Command Line Interface

```
npm install --global @angular/cli

ng new blinking-pi --prefix pi --style scss --routing

cd blinking-pi

npm start

open http://localhost:4200
```

Configuration options

1. `angular.json`
2. `tsconfig.json`
3. `src/tsconfig.app.json`
4. `tslint.json`
5. `karma.conf.js`
6. `protractor.conf.js`

1. ES5 has **no own** module system
2. ES2015 defines a new module system
3. Current VMs are not yet implementing the new module system *(completely)*
4. TypeScript can generate code for various (*legacy*) module systems
5. A Detailed Introduction To Webpack


Components

Purpose

— Everything you can see or use in the user interface (*UI*) is made up of components.

— Components are **always** composed of *HTML*, *TypeScript* and, if applicable, *CSS/SASS/LESS*.

— Components can communicate with each other and are always arranged in a *tree structure*

Component

Import

Decorator

Class

Bootstrapping

Konventionen


Templating

Purpose

1. The UI is composed of HTML templates
2. Angular Templates (*DSL*) support
   - Custom Components
   - DOM-Property-Binding
   - Data- / Event-Binding
   - Directives
3. Angular compiler transforms templates into

Data Binding

Directives

Component

Styles

Component Lifecycle


Pipes

Purpose

— Pipes are used to transform data.

— In the template of a component, one uses ' | ' to convert the data that is to the left of that symbol into other data.

Built-In Pipes

Custom Pipes


Services

Purpose

- Services do work such as:
  - Send or receive data via HTTP
  - Run business logic
  - Keep data in memory / local storage

Service erstellen

Dependency Injection

Service nutzen

Servicehierarchie

HTTP, RxJS

HTTP Client

RxJS

Observables

Router, Navigation

Routing

Router

Setup Navigation


Forms

Template-Driven Forms

Reactive Forms


Testing

Testing in JavaScript

Test-Frameworks

Test-Runner

Testing in the Cloud

Testing in Angular


Upgrade

AngularJS Styleguide

Module Loader

TypeScript

Component Directives