UP AND RUNNING WITH ANGULAR 6

Justin James

Senior Software Engineer
Microsoft MVP – Developer Technologies
Founder of Speaker Coaching Specialist

Who Am I?

Web Developer for the past 21 years

Been using Angular 2+ for the past 3 years

Microsoft MVP in Developer Technologies

Speak at 10-12 conferences each year

Founder and CEO of Speaker Coaching Specialist

Morning Agenda

Release Plan

Setup

TypeScript Essentials

Angular Essentials

Reactive Forms

Afternoon Agenda

Route Security

Default Route

Reusable Components

Environment Settings

Deployment

Your Experience?

HTML, CSS, JavaScript

TypeScript

Angular 1

Angular 2

Your Expectations?

What do you want to get out of today?

Some "Rules"

Every question is important

Help each other

Have fun

Angular

Framework for building client side applications using HTML, CSS, and TypeScript

Why Angular

Expressive HTML

Powerful Data Binding

Modular

Streamlined Http Calls

Why Angular 2+?

Modern

Speed

Simplified API

Enhanced Productivity

ANGULAR 6 ALREADY???

Angular Release Plan

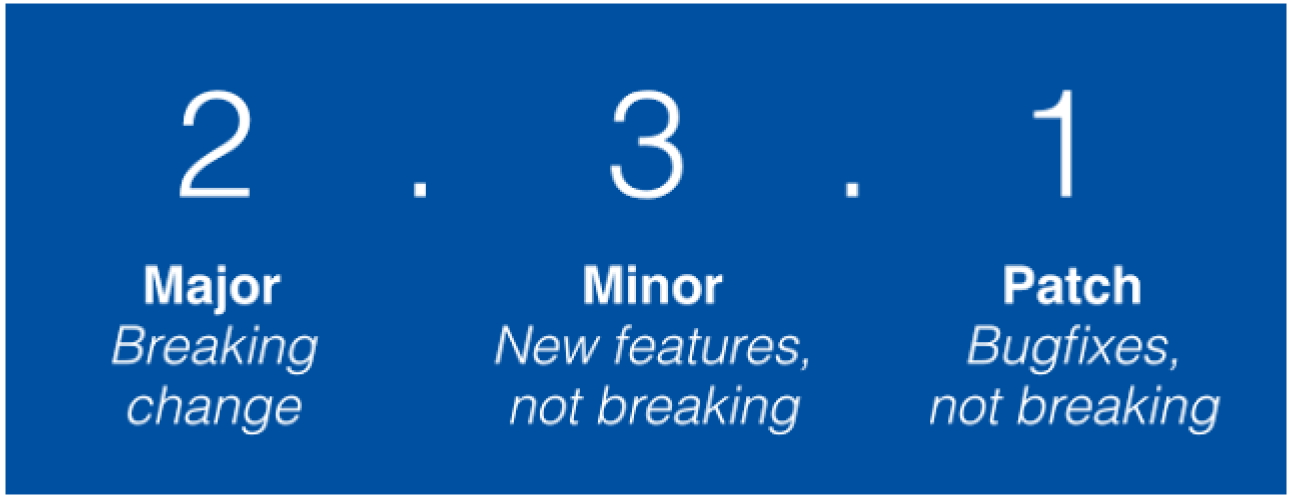
Weekly – Patches

Monthly – Minor Versions

Every 6 Months – Major Versions

Major Version Release	Timeframe
Angular 4	March 2017
Angular 5	Oct 2017
Angular 6	May 2018
Angular 7	Oct 2018

Semantic Versioning (SEMVER)



2 . 3 . 1

Major	Minor	Patch
Breaking change	*New features, not breaking*	*Bugfixes, not breaking*

Release Details

https://angular.io/guide/releases

SETUP

Development Environment

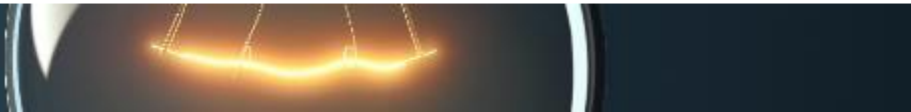
Angular Setup

Node JS (suggest 8+ LTS)

http://nodejs.org

NPM Global Packages

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



Creating Applications

Create New Project

```
ng  new AppName --style scss --routing
```

Generate

```
ng generate [TYPE] [NAME]
```

Start Application

```
ng serve
```

Editor

Visual Studio Code

Extensions:

- Angular Language Service
- TS Lint
- Prettier

TYPESCRIPT ESSENTIALS

Base Knowledge for Getting Started

https://digitaldrummerj.me

digitaldrummerj

TypeScript

Typed Superset of JavaScript

Features

Supports JavaScript

Static Typing

Encapsulation

Intellisense &
Syntax Checking

```
var items = getItems();

var goSportsTeam = true;

var super = 'mario';
```

Most JavaScript is already valid TypeScript

~~some-javascript.js~~

some-typescript.ts

Typing….

Dynamic

JavaScript, Python, Ruby, PHP

```
var number = 5;
number = "Hello!";
// work great!
```

Static

C, C++, C#, Java

```
int number = 10;
number = "Hello!";
// Compiler Error
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, 5);
```

JavaScript Typing Fun

```javascript
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, 5);

>> 10
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, true);
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, true);

>> 6
```

🙁

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, 'js4lyfe');
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, 'js4lyfe');

>> "5js4lyfe"
```

😁

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, '5');
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, '5');

>> "55"
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, { param: true });
```

JavaScript Typing Fun

```
function numberCruncher (numA, numB){
    return numA + numB;
}

var result = numberCruncher(5, { param: true });

>> "5[object Object]"
```

```
status: string;

havingFun: boolean;

daysTillVacation: number;
```

Types

boolean, number, string
array, enum
any, void

```
status = 'TS Rocks!';

havingFun = true;

daysTillVacation = 60;
```

Types

boolean, number, string
array, enum
any, void

```
myNumbers : number[] = [170, 2.6, 2245, 3032, 400];

// Or...

myNumbers : Array<number> = [170, 2.6, 2245, 3032, 400];
```

Arrays

List of values

Argument of type 'string' is not assignable to parameter of type 'number'

```
myNumbers: number[] = [];

myNumbers.push('700');
```

Arrays

Enforce types for array content

```
myTypedFunction(paramName : dataType) : returnType {
    // code here
}
```

Functions

Input Parameter and Return Types

```
myTypedFunction(paramName : dataType) : returnType {
    // code here
}
```

Functions

Input Parameter and Return Types

```
myTypedFunction(paramName : dataType) : returnType {
    // code here
}
```

Functions

Input Parameter and Return Types

```
myTypedFunction(paramName : dataType) : returnType {
    // code here
}
```

Functions

Input Parameter and Return Types

```
myTypedFunction(paramName : dataType) : returnType {
    // code here
}
```

Functions

Input Parameter and Return Types

```
trimLength(inputVal: string): number {
    return inputVal.trim().length;
}
```

Functions

Input Parameter and Return Types

```
trimLength(inputVal: string): number {
    return inputVal.trim().length;
}
```

Functions

Input Parameter and Return Types

```
trimLength(inputVal: string): number {
    return inputVal.trim().length;
}
```

Functions

Input Parameter and Return Types

```
trimLength(inputVal: string): number {
    return inputVal.trim().length;
}
```

Functions

Input Parameter and Return Types

```
trimLength(inputVal: string): number {
    return inputVal.trim().length;
}
```

Functions

Input Parameter and Return Types

```
initSomething() : void {
    // do something
}
```

Void

A function that returns nothing

```
initSomething() : void {
    // do something
}
```

Void

A function that returns nothing

```
initSomething() : void {
    // do something
}
let pointless = initSomething();
```

Void

A function that returns nothing

```
initSomething() : void {
    // do something
}
let pointless = initSomething(); // Compiler Error!
```

Void

A function that returns nothing

```
let foo: any;
```

Any

Restores basic JavaScript dynamic typing behavior

```
let foo: any;

foo = 'Hello';

foo = true;

foo = 42;
```

THINK TWICE BEFORE USING!

Any

Restores basic JavaScript dynamic typing behavior

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}
```

Classes

Encapsulate functionality and data for an object

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}
```

Classes

Encapsulate functionality and data for an object

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}
```

Classes

Encapsulate functionality and data for an object

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}
```

Classes

Encapsulate functionality and data for an object

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}
```

Classes

Encapsulate functionality and data for an object

```
export class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello " + this.greeting;
    }
}

let greeter = new Greeter("world").greet();
```

Classes

Encapsulate functionality and data for an object

```
interface Label {
  value: string;
}
```

Interfaces

Powerful way of defining contracts

```
interface Label {
  value: string;
}

class label implements Label {
  value: string = 'label value';
}
```

Interfaces

Powerful way of defining contracts

```
interface Label {
  value: string;
}

label: Label = {
  value: 'label value';
}
```

Interfaces

Powerful way of defining contracts

Official Website
www.typescriptlang.org

TYPESCRIPT Q&A

ANGULAR ESSENTIALS

Just enough to be really dangerous

Main Building Blocks

Components

Routing

Services

Modules

Component

Most basic building block of the UI

Contains a template, stylesheet, and data logic

Can include other components

Provides CSS Isolation

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Template

HTML to tell Angular how to render a component

Include data binding as well as other components and directives

Leverages native DOM events and properties

Template – App Component Html

app.component.html

```html
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
```

Template – App Component Html

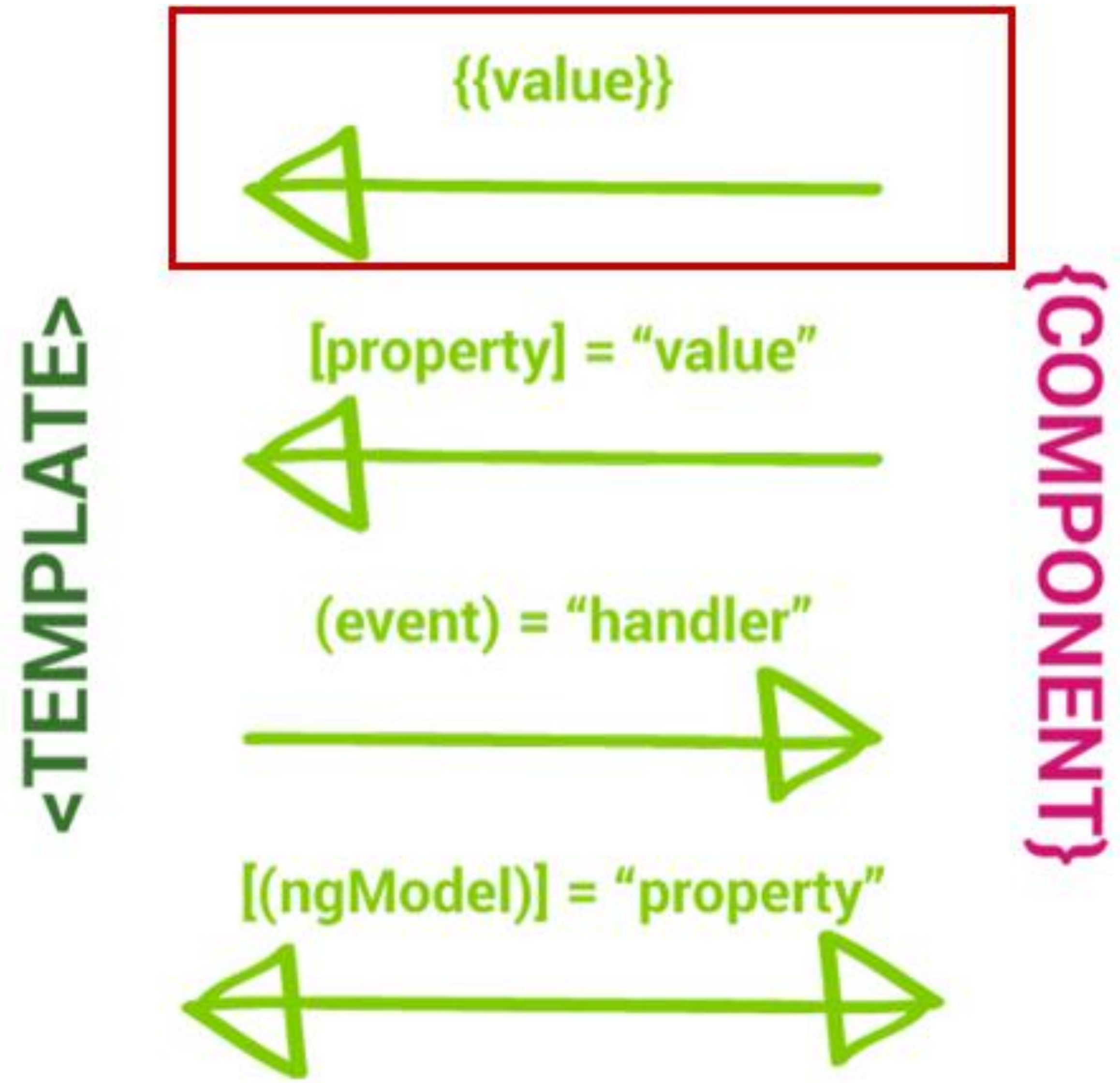
app.component.html

```
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
```

Data Binding

< >

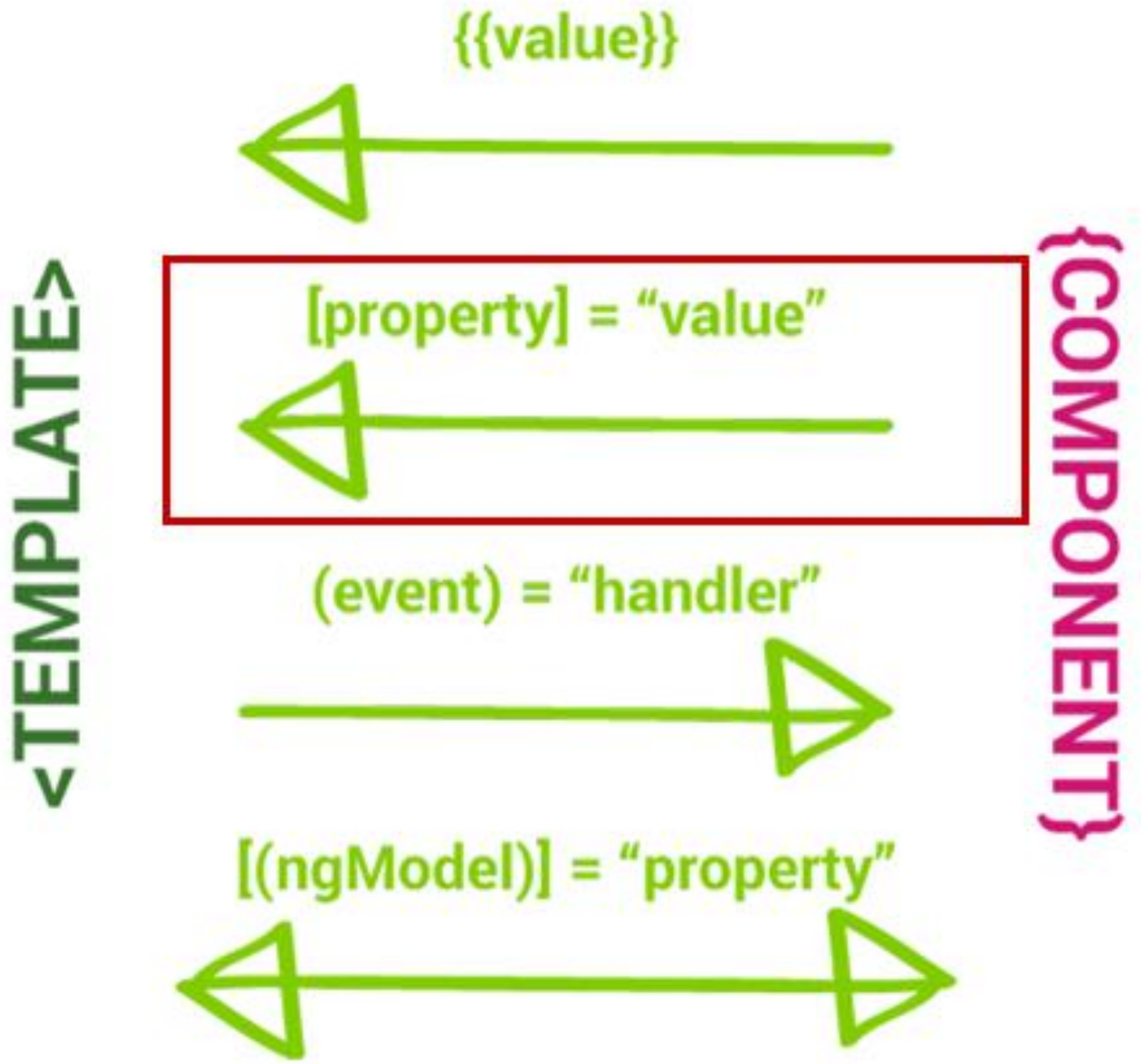
Data Binding



Template – data binding

```
<h3>{{ title }}</h3>
```

Data Binding



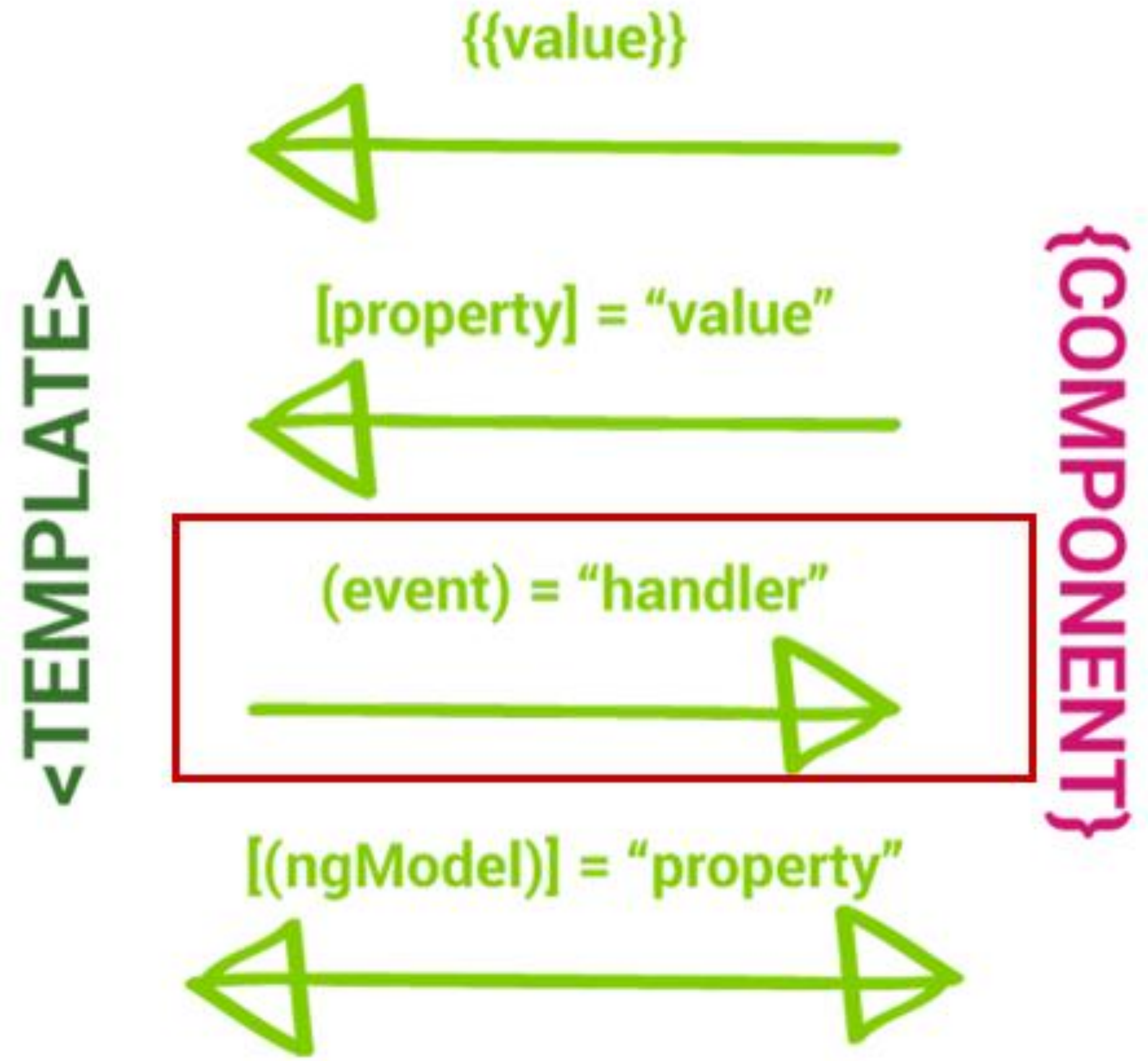
Template – data binding

```
<h3>{{ title }}</h3>

<h3 [innerHTML]="title"></h3>

<img [src]="logo">
```

Data Binding



Template – data binding

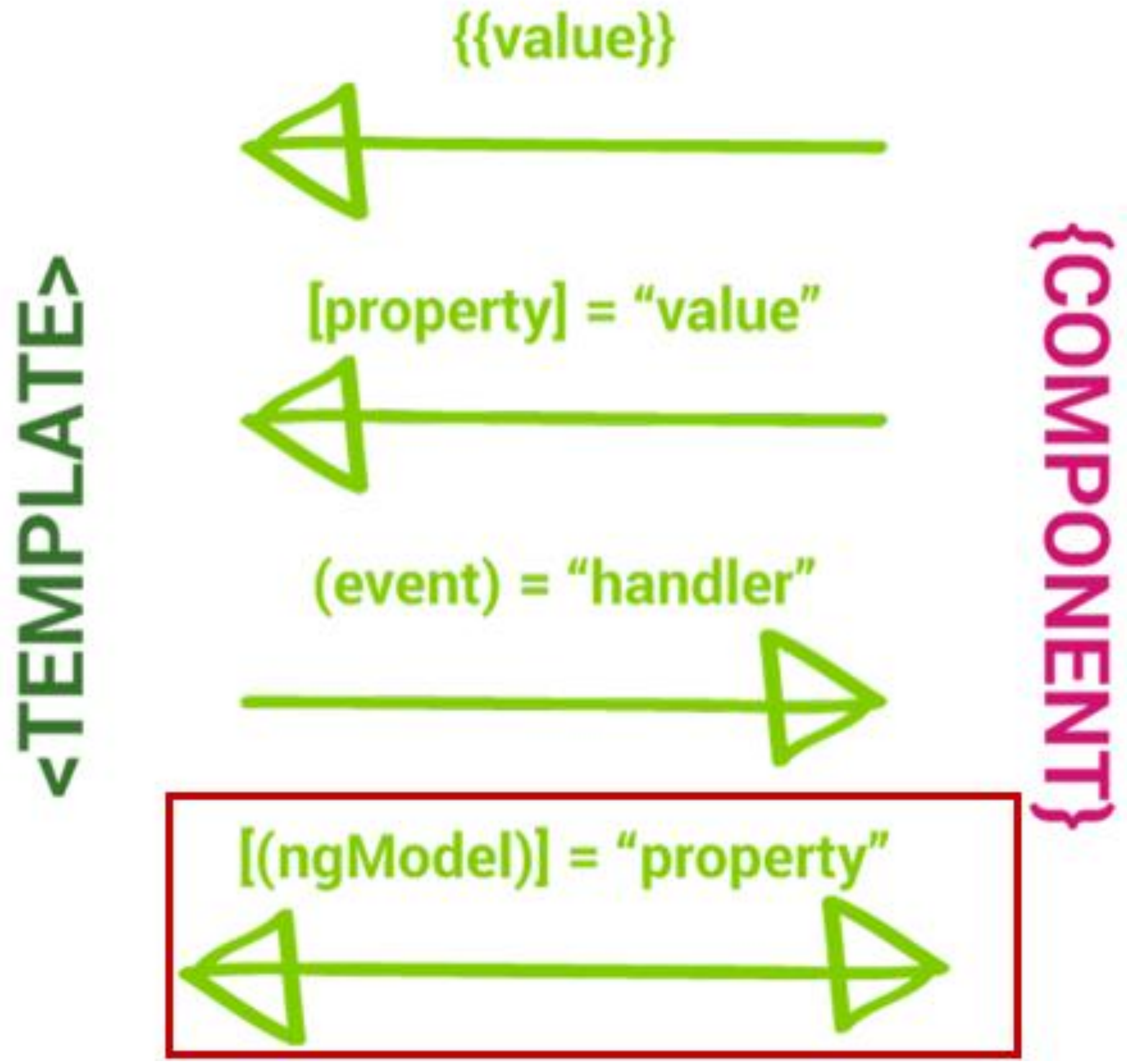
```
<h3>{{ title }}</h3>

<h3 [innerHTML]="title"></h3>

<img [src]="logo">

<div (click)="doSomething($event)"></div>
```

Data Binding



Template – data binding

```
<h3>{{ title }}</h3>

<h3 [innerHTML]="title"></h3>

<img [src]="logo">

<div (click)="doSomething($event)"></div>

<input [(ngModel)]="title" />
```

Lifecycle Events

OnInit

OnChanges

OnDestroy

Routing

Renders component based on the URL state

Drives navigation

Lazy Loading

Parent/Child Routes

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent }
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent }
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent },
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent }
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent }
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent
    },
    { path: 'login', children: [], component: LoginComponent }
];
```

Template – App Component Html

app.component.html

```
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
```

Template – App Component Html

app.component.html

```html
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
```

Services

Data layer

Logic is not component related

Invariably asynchronous

Observables instead of Promises

Service Code – TodoService

shared\services\todo.service.ts

```
import { Injectable } from '@angular/core';


@Injectable({
    providedIn: 'root'
})
export class TodoService {
    constructor() {}
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Injectable({
    providedIn: 'root'
})
export class TodoService {
    constructor(private http: HttpClient) {}
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Service Code – TodoService

shared\services\todo.service.ts

```
import { Observable } from 'rxjs';
import { Todo } from '../classes/todo';

export class TodoService {
    constructor(private http: HttpClient) {}

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url);
    }
}
```

Component – Getting Data – TodoComponent

todo.component.ts

```
todoList: Todo[] = [];

this.todoService
    .getAll()
    .subscribe(
        (data: Todo[]) => {
            this.todoList = data;
        },
        (error: HttpErrorResponse) => {
            this.errorMessage =
                `${error.status} ${error.statusText}. ${error.message}`;
        }
    );
```

Component – Getting Data – TodoComponent

todo.component.ts

```
todoList: Todo[] = [];

this.todoService
    .getAll()
    .subscribe(
        (data: Todo[]) => {
            this.todoList = data;
        },
        (error: HttpErrorResponse) => {
            this.errorMessage =
                `${error.status} ${error.statusText}. ${error.message}`;
        }
    );
```

Component – Getting Data – TodoComponent

todo.component.ts

```
todoList: Todo[] = [];

this.todoService
    .getAll()
    .subscribe(
        (data: Todo[]) => {
            this.todoList = data;
        },
        (error: HttpErrorResponse) => {
            this.errorMessage =
                `${error.status} ${error.statusText}. ${error.message}`;
        }
    );
```

Component – Getting Data – TodoComponent

todo.component.ts

```
todoList: Todo[] = [];

this.todoService
    .getAll()
    .subscribe(
        (data: Todo[]) => {
            this.todoList = data;
        },
        (error: HttpErrorResponse) => {
            this.errorMessage =
                `${error.status} ${error.statusText}. ${error.message}`;
        }
    );
```

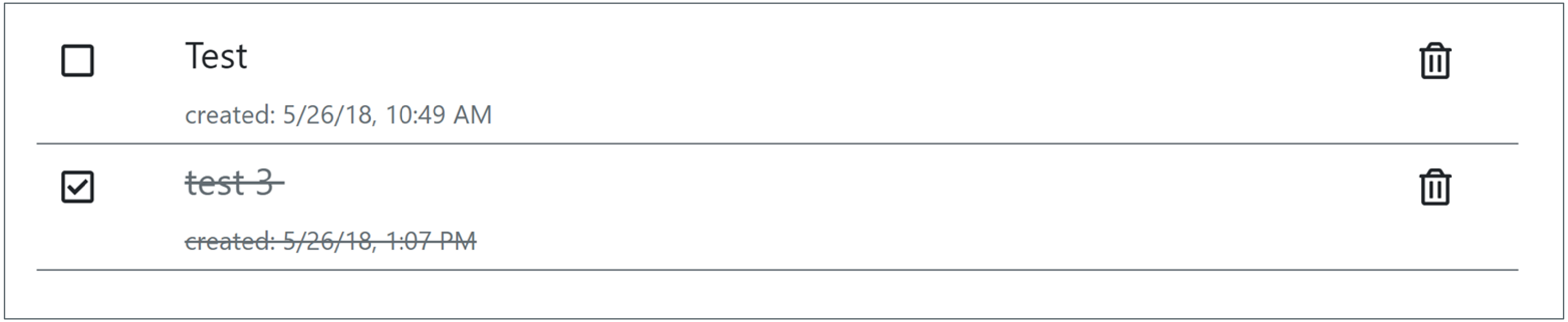
Component – Getting Data – TodoComponent

todo.component.ts

```
todoList: Todo[] = [];

this.todoService
    .getAll()
    .subscribe(
        (data: Todo[]) => {
            this.todoList = data;
        },
        (error: HttpErrorResponse) => {
            this.errorMessage =
                `${error.status} ${error.statusText}. ${error.message}`;
        }
    );
```

Component – Display – TodoComponent


☐ Test

created: 5/26/18, 10:49 AM

☑ ~~test 3~~

~~created: 5/26/18, 1:07 PM~~

Component – Display Data – TodoComponent

todo.component.html

```html
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

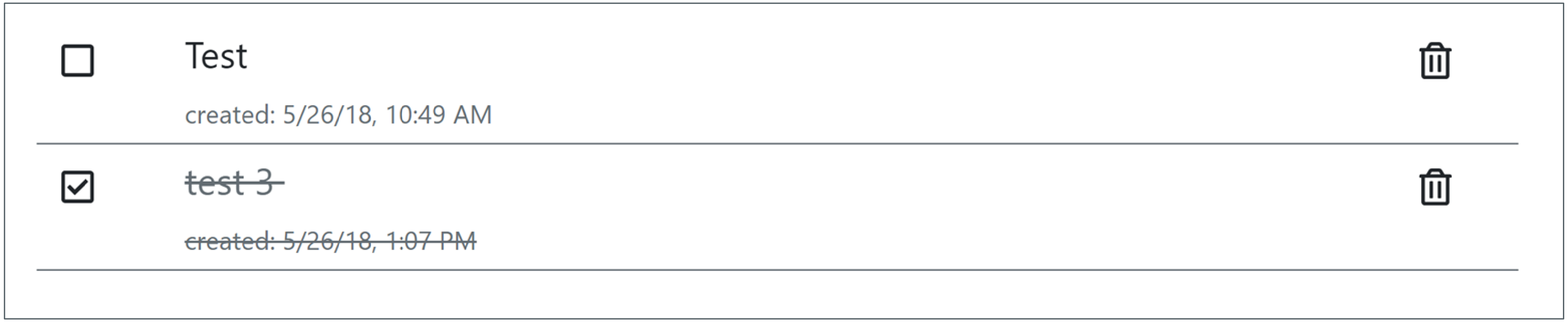
```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display Data – TodoComponent

todo.component.html

```
<div class="row todo" *ngFor="let todoItem of todoList">
    <div class="col-1" (click)="completeTodo(todoItem)">
        <fa-icon [icon]="['far', todoItem.completed ? 'check-square' :
'square']"></fa-icon>
    </div>
    <div class="col-10 done-{{todoItem.completed}}">
        {{todoItem.item}}
        <br />
        <small>created: {{ todoItem.createdAt | date:'short'}}</small>
    </div>
    <div class="col-1" (click)="deleteTodo(todoItem)">
        <fa-icon [icon]="['far', 'trash-alt']"></fa-icon>
    </div>
</div>
```

Component – Display – TodoComponent



☐ Test

created: 5/26/18, 10:49 AM

☑ ~~test 3~~

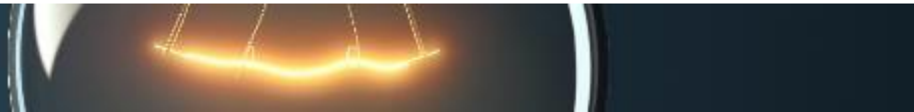
~~created: 5/26/18, 1:07 PM~~

Modules

Organize application into blocks of functionality.

Contain routes, components, services, and more.

Every app has one module minimum, the root module.



Module - AppModule

app.module.ts

```
@NgModule({
    declarations: [
        AppComponent, LoginComponent, SignupComponent
    ],
    imports: [
        BrowserModule, FormsModule, ReactiveFormsModule,
        HttpClientModule, AppRoutingModule
    ],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

Module - AppModule

app.module.ts

```
@NgModule({
    declarations: [
        AppComponent, LoginComponent, SignupComponent
    ],
    imports: [
        BrowserModule, FormsModule, ReactiveFormsModule,
        HttpClientModule, AppRoutingModule
    ],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

Module - AppModule

app.module.ts

```
@NgModule({
    declarations: [
        AppComponent, LoginComponent, SignupComponent
    ],
    imports: [
        BrowserModule, FormsModule, ReactiveFormsModule,
        HttpClientModule, AppRoutingModule
    ],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

Module - AppModule

app.module.ts

```
@NgModule({
    declarations: [
        AppComponent, LoginComponent, SignupComponent
    ],
    imports: [
        BrowserModule, FormsModule, ReactiveFormsModule,
        HttpClientModule, AppRoutingModule
    ],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

Module - AppModule

app.module.ts

```
@NgModule({
    declarations: [
        AppComponent, LoginComponent, SignupComponent
    ],
    imports: [
        BrowserModule, FormsModule, ReactiveFormsModule,
        HttpClientModule, AppRoutingModule
    ],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

ANGULAR ESSENTIALS Q&A

Labs

Labs:
https://speakercoachingspecialist.com/ngws

- Lab 2: 30 minutes
- Skip Lab 3-5
- Lab 6.3: 15 minutes
- Stop After Lab 6.3

REACTIVE FORMS

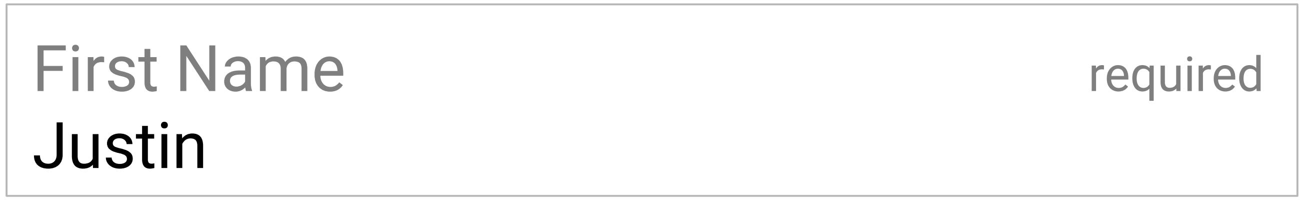
the recommended approach

Building Blocks of Forms

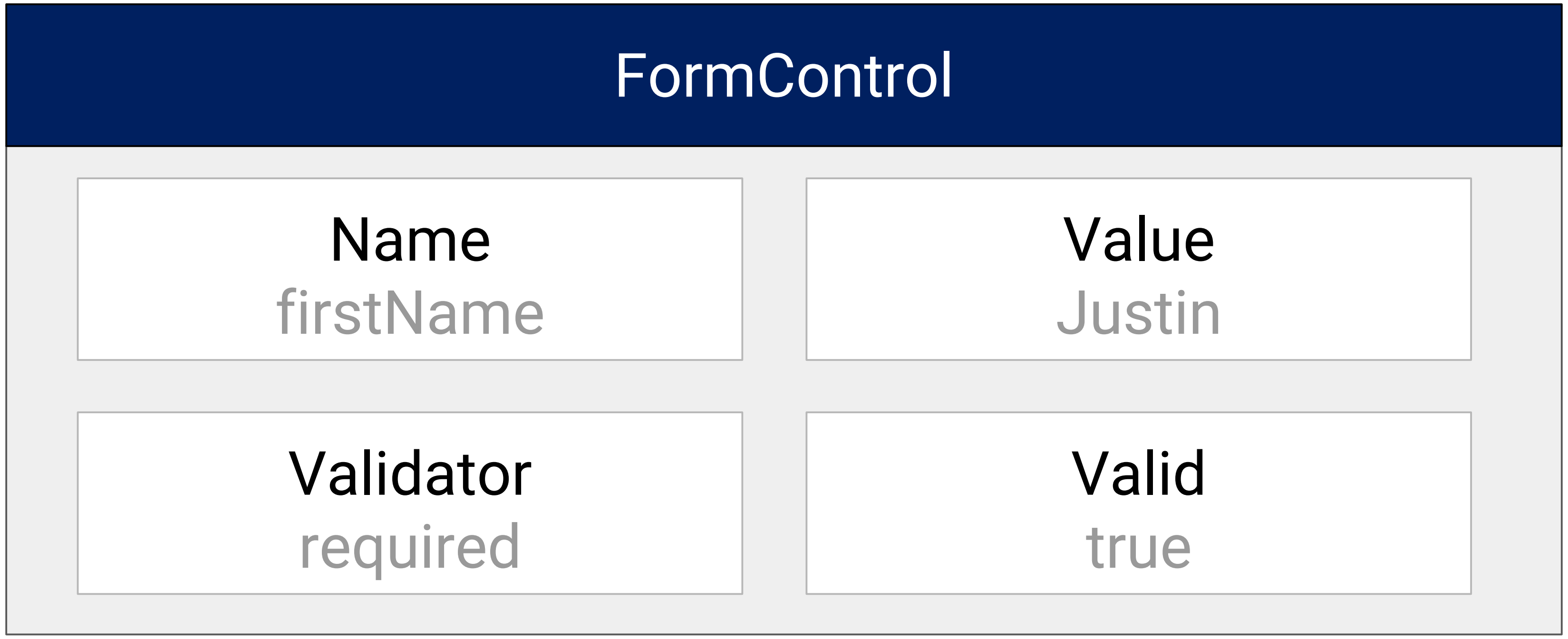
FormControl

FormGroup

FormArray

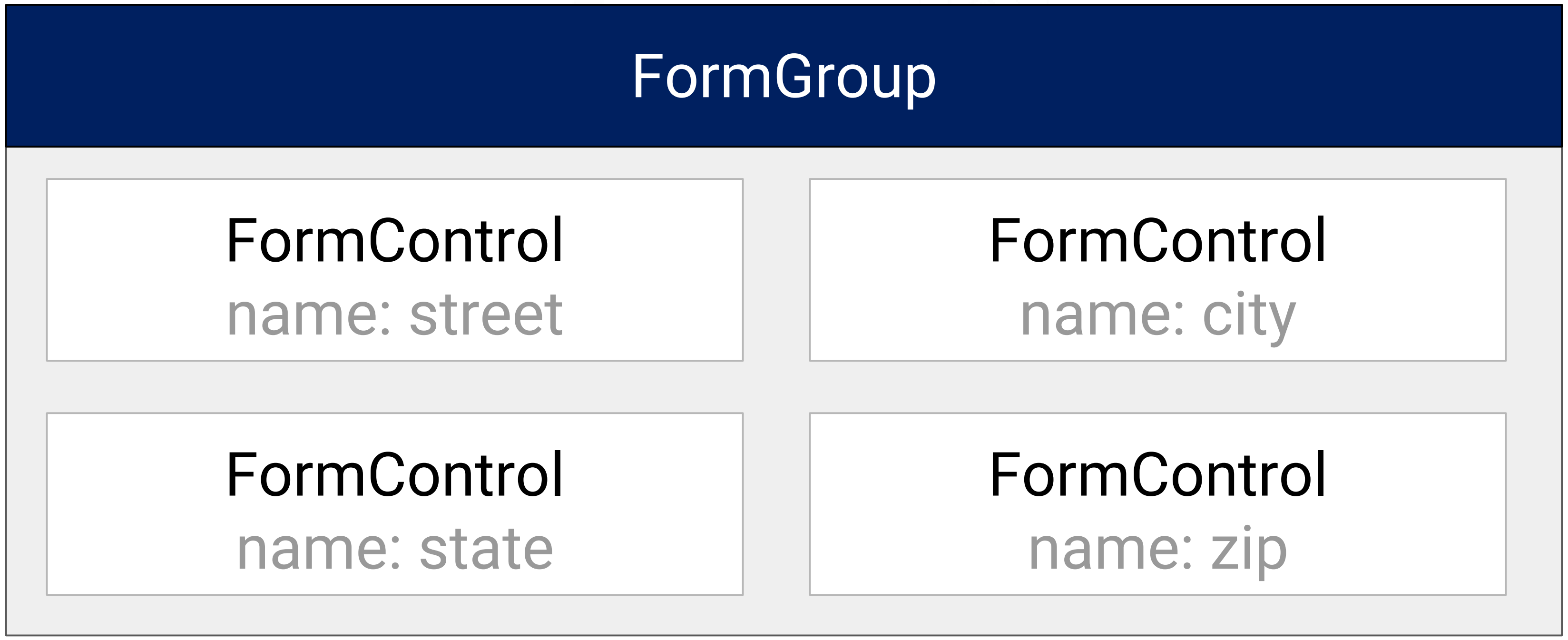


First Name required
Justin



Street ______________________________

City ____________ State __________ Zip __________



Street ______________________________

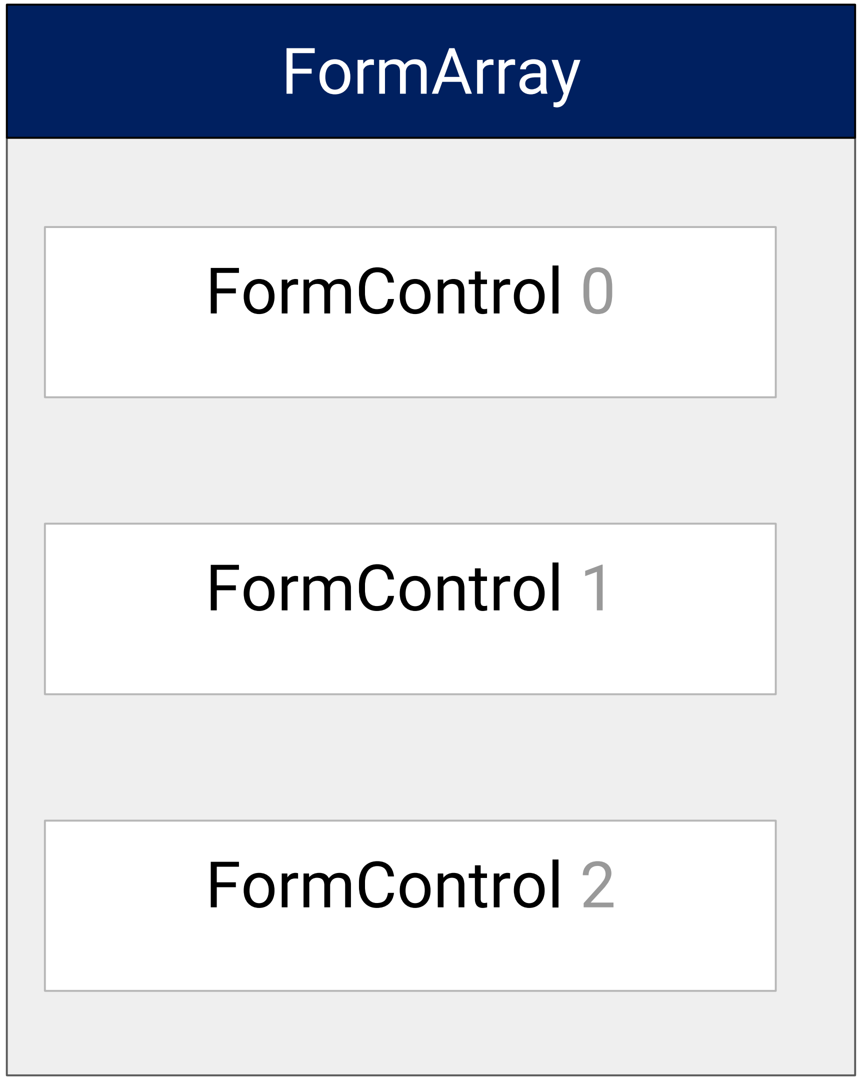
City ________State ____ Zip __________

Street ______________________________

City ________State ____ Zip __________

Street ______________________________

City ________State ____ Zip __________



Built-in Validators

Required	Email	Null	Pattern
Max	MaxLength	Min	MinLength

https://angular.io/api/forms/Validators

Form Control States

Touched

Untouched

Valid

Invalid

Pristine

Dirty

https://angular.io/guide/forms#track-control-state-and-validity-with-ngmodel

Reactive Approach

Form Setup
In Component

Validation Done
In Component

Data Binding
Implicitly Created

Benefits of Reactive Approach

Testable

Extendable Validation Rules

More Maintainable

Typings Prevent Errors

Can Monitor Form Changes

Drawbacks of Model Driven Approach

None

ENABLE REACTIVE FORMS MODULE

Enable Model Drive Approach - AppModule

app.module.ts

```
import { ReactiveFormsModule } from '@angular/forms';

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

Enable Model Drive Approach - AppModule

app.module.ts

```
import { ReactiveFormsModule } from '@angular/forms';

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

Enable Model Drive Approach - AppModule

app.module.ts

```
import { ReactiveFormsModule } from '@angular/forms';

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

Enable Model Drive Approach - AppModule

app.module.ts

```
import { ReactiveFormsModule } from '@angular/forms';

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

SETUP FORM DEFINITION

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

Model Driven Form - TodoComponent

todo.component.ts

```
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

export class TodoComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {}

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            item: ['', [Validators.required, Validators.minLength(3)]],
        });
    }
}
```

FORM HTML

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

Model Driven Form - TodoComponent

todo.component.html

```
<form (ngSubmit)="save()" [formGroup]="addForm">

    <input
        type="text"
        formControlName="item">

    <button
        type="submit"
        [disabled]="addForm.invalid">
        Add
    </button>

</form>
```

SAVE FORM DATA

Model Driven Form Submit

```
save(): void {
    console.log(this.addForm.value.item);
}
```

FORM VALIDATION

Model Driven Form Watch For Changes

todo.component.ts

```
ngOnInit() {
    this.addForm = this.formBuilder.group({
        item: ['', [Validators.required, Validators.minLength(3)]],
    });

    this.addForm
        .statusChanges
        .subscribe(data => this.onStatusChange(data));

    this.onStatusChange();
}
```

Model Driven Form Watch For Changes

todo.component.ts

```
ngOnInit() {
    this.addForm = this.formBuilder.group({
        item: ['', [Validators.required, Validators.minLength(3)]],
    });

    this.addForm
        .statusChanges
        .subscribe(data => this.onStatusChange(data));

    this.onStatusChange();
}
```

Model Driven Form Watch For Changes

todo.component.ts

```
ngOnInit() {
    this.addForm = this.formBuilder.group({
        item: ['', [Validators.required, Validators.minLength(3)]],
    });

    this.addForm
        .statusChanges
        .subscribe(data => this.onStatusChange(data));

    this.onStatusChange();
}
```

Model Driven Form Watch For Changes

todo.component.ts

```
ngOnInit() {
    this.addForm = this.formBuilder.group({
        item: ['', [Validators.required, Validators.minLength(3)]],
    });

    this.addForm
        .statusChanges
        .subscribe(data => this.onStatusChange(data));

    this.onStatusChange();
}
```

Model Driven Form Generic Validation

todo.component.t.s

```
onStatusChange(data?: any) {
    const form = this.addForm;
    for (const field in this.formErrors) {
        this.formErrors[field] = '';
        const control = form.get(field);
        if (control && control.dirty && !control.valid) {
            const messages = this.validationMessages[field];
            for (const key in messages) {
                if (control.hasError(key)) {
                    this.formErrors[field] += `${messages[key]} `;
                }
            }
        }
    }
}
```

Model Driven Form Validation - TodoComponent

todo.component.ts

```
formErrors = {
    item: '',
};

validationMessages = {
    item: {
        required: 'Item is required.',
        minlength: 'Item must be at least 3 characters',
    },
};
```

Model Driven Form Validation - TodoComponent

todo.component.ts

```
formErrors = {
    item: '',
};

validationMessages = {
    item: {
        required: 'Item is required.',
        minlength: 'Item must be at least 3 characters',
    },
};
```

Model Driven Form Validation - TodoComponent

todo.component.ts

```
formErrors = {
    item: '',
};

validationMessages = {
    item: {
        required: 'Item is required.',
        minlength: 'Item must be at least 3 characters',
    },
};
```

Model Driven Form Validation - TodoComponent

todo.component.html

```
<input type="text" formControlName="item">

<div *ngIf="formErrors.item" >
    {{ formErrors.item }}
</div>
```

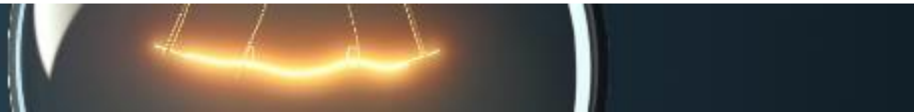
Model Driven Form Validation - TodoComponent

todo.component.html

```
<input type="text" formControlName="item">

<div *ngIf="formErrors.item" >
    {{ formErrors.item }}
</div>
```

REACTIVE FORMS Q&A



Labs

Labs:
https://speakercoachingspecialist.com/ngws

Skip Lab 3-5

Lab 6: 1 hour (starting at 6.4)

LOCKING DOWN ROUTES

Locking Down Routes

Check if you are allowed to view a route

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Guard

is-logged-in.guard.ts

```
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot,
Router } from '@angular/router';

@Injectable()
export class IsLoggedInGuard implements CanActivate {
    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
    }
}
```

Restricting Route

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent,
        canActivate: [IsLoggedInGuard]
    },
];
```

Restricting Route

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent,
        canActivate: [IsLoggedInGuard]
    },
];
```

Where should check if user actually has access or not?

Client or Server

ROUTE SECURITY Q&A

Labs

Labs:
https://speakercoachingspecialist.com/ngws

Lab 7: 30 minutes

DEFAULT ROUTES

What to do when Angular doesn't where to go to....

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent,
        canActivate: [IsLoggedInGuard]
    },
    { path: 'login', children: [], component: LoginComponent },
    { path: '**', component: NotFoundComponent }
];
```

Routing Code - AppRoutingModule

app-routing.module.ts

```
const routes: Routes = [
    {
        path: '',
        component: TodoComponent,
        canActivate: [IsLoggedInGuard]
    },
    { path: 'login', children: [], component: LoginComponent },
    { path: '**', component: NotFoundComponent }
];
```

needs to be last route!

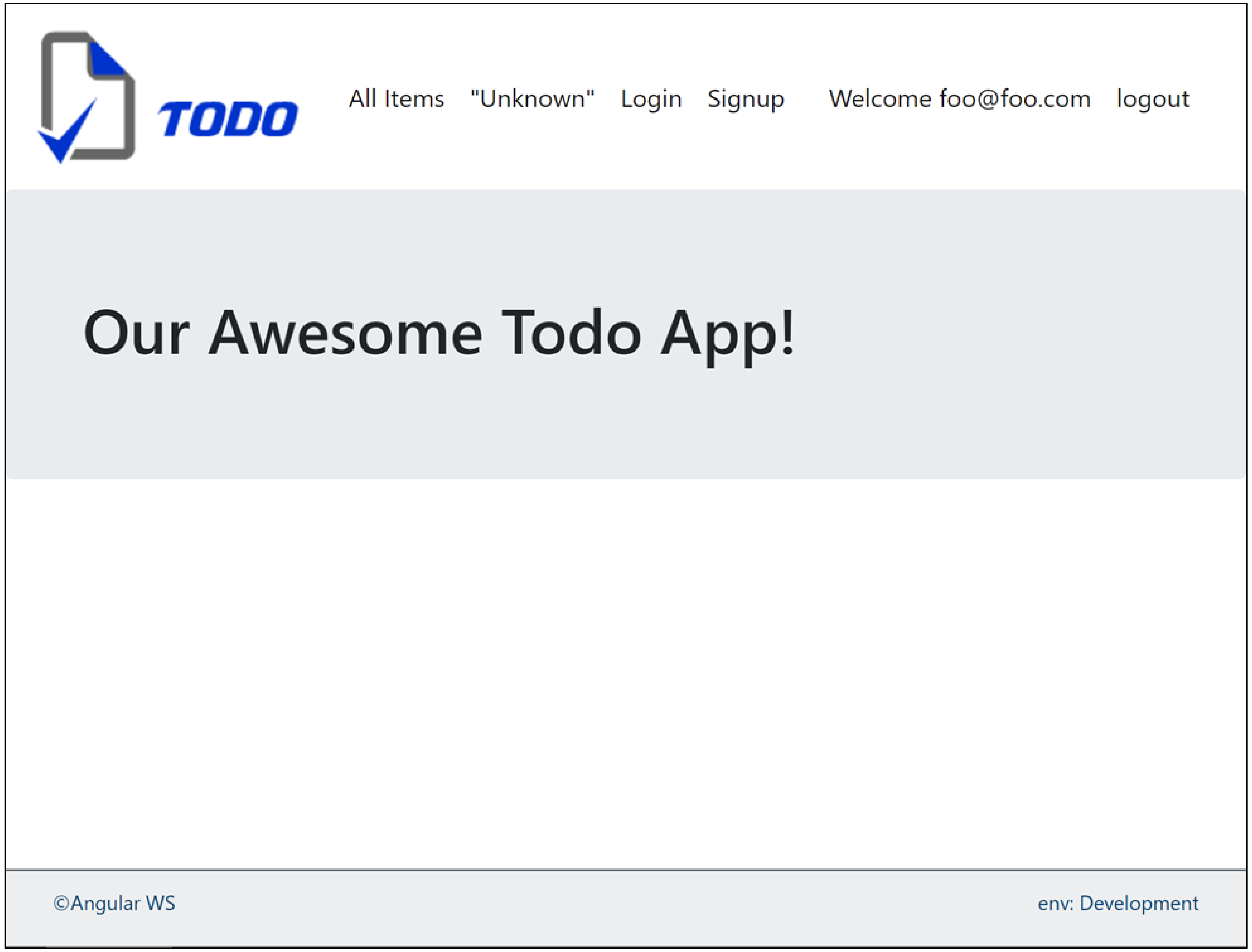
Labs

Labs:
https://speakercoachingspecialist.com/ngws

Lab 8: 15 minutes

REUSABLE COMPONENTS

Simple Header and Footer



TODO

All Items "Unknown" Login Signup Welcome foo@foo.com logout

Our Awesome Todo App!

©Angular WS

env: Development

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';


@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Component – AppComponent

app.component.ts

```
import { Component } from '@angular/core';
import { HeaderComponent } from './shared/header/header.component';
import { FooterComponent } from './shared/footer/footer.component';

@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.scss']
})
export class AppComponent {
    title = 'app works!';
}
```

Template – App Component Html

app.component.html

```html
<app-header></app-header>
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
<app-footer></app-footer>
```

Template – App Component Html

app.component.html

```
<app-header></app-header>
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
<app-footer></app-footer>
```

Component – HeaderComponent

header.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-header',
    templateUrl: './header.component.html',
    styleUrls: ['./header.component.scss']
})
export class HeaderComponent implements OnInit, OnDestroy {
}
```

Component – HeaderComponent

header.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-header',
    templateUrl: './header.component.html',
    styleUrls: ['./header.component.scss']
})
export class HeaderComponent implements OnInit, OnDestroy {
}
```

Template – App Component Html

app.component.html

```html
<app-header></app-header>
<div class="jumbotron">
    <div class="container">
        <h1>{{title}}</h1>
    </div>
</div>
<div class="container">
    <router-outlet></router-outlet>
</div>
<app-footer></app-footer>
```

Component –FooterComponent

footer.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-footer',
    templateUrl: './footer.component.html',
    styleUrls: ['./footer.component.scss']
})
export class FooterComponent implements OnInit {
}
```

Component –FooterComponent

footer.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'app-footer',
    templateUrl: './footer.component.html',
    styleUrls: ['./footer.component.scss']
})
export class FooterComponent implements OnInit {
}
```

REUSABLE COMPONENTS Q&A

Labs

Labs:
https://speakercoachingspecialist.com/ngws

Lab 9: 30 minutes

ENVIRONMENT SPECIFIC SETTINGS

Application Settings

Shouldn't hard code values

Generate build using different values

Use Environment Files

Environment File – environment.ts

```
export const environment = {
    production: false,
    environmentName: 'Development',
    apiBaseUrl: 'https://sails-ws.herokuapp.com'
};
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

Using Environment File

todo.service.ts

```
import { environment } from '../../../environments/environment';

@Injectable()
export class TodoService {
    private url = `${environment.apiBaseUrl}/todo`;

    getAll(): Observable<Todo[]> {
        return this.http.get<Todo[]>(this.url, requestOptions);
    }
}
```

DEPLOYING

ng lint

Lint

verify coding style

```
ng test --watch=false --code-coverage=true --browsers ChromeHeadless
```

Unit Testing

run with code coverage using Chrome headless

```
ng e2e -c protractor.conf.ci.js
```

E2E Test

run the end to end test using Protractor

```
capabilities: {
    browserName: 'chrome',
    chromeOptions: {
        args: [
            '--headless', '--disable-gpu',
            '--window-size=1280,768'
        ],
    },
}
```

E2E Test Configuration

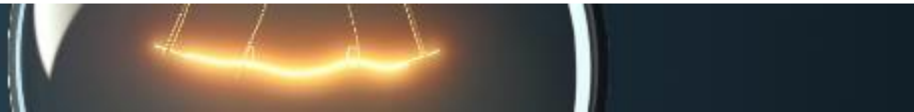
run with Chrome headless

```
ng build --prod
```

Production Build

ready to deploy build

ADDITIONAL RESOURCES



Angular Unit Testing

youtube.com/watch?v=uKqs7cLWSO4

Docs

Angular Style Guide
https://angular.io\docs\ts\latest\guide\style-guide.html

Angular Docs
https://angular.io/

Angular CLI Docs
https://github.com/angular/angular-cli/wiki

RETROSPECTIVE

Contact Info

digitaldrummerj.me

/digitaldrummerj

speakercoachingspecialist.com

please submit your evaluations

speakercoachingspecialist.com/ng-feedback