**Fundamentals – Angular** (examples in TypeScript)

**by Todd Bagley**

**ANGULAR.JS** – First version of Angular. “Angular 1.0”

**ANGULAR** – Second and subsequent versions of Angular. “Angular 2, Angular 4, Angular 5”

**DIFFERENCES** – why Angular over Angular.JS?

**css scoping** – automatic scoping of css to a single component

**components** – instead of controllers and scope = simpler

**simplified directives** – creating custom directives is simpler

**intuitive data binding** – intuitive syntax for linking data and listening to events

**services** – are now just a class

**LIBRARY SCRIPT** – import app/main.ts

<head>

<script>

System.import(‘app’)

.catch(function(err){ console.error(err); });

</script>

</head>

**main.ts** – app/main.ts

import { NgModule, Component } from ‘@angular/core’;

import { BrowserModule } from ‘@angular/platform-browser’;

import { platformBrowserDynamic } from ‘@angular/platform-browser-dynamic’;

@Component({

selector: ‘my-app’,

template: `<h1>{{title}}</h1>

<p>{{body.begin}} {{body.end}}</p>`

})

class AppComponent {

title = ‘Hello World’;

body = { “begin”: ‘Hello’ , “end”: ‘world’ };

}

@NgModule({

declarations:[ AppComponent ],

imports: [ BrowserModule ],

bootstrap: [ AppComponent ]

})

class AppModule { }

platformBrowserDynamic()

.bootstrapModule(AppModule);

**export** – ES2015 feature used to export functions, objects or primatives

**import** – ES2015 feature used to import functions, objects or primatives

**Component** – function used to create components, basic building blocks of Angular

each controlling a portion of the screen

**STRUCTURAL DIRECTIVES** – directives:

**ngIf** – if statement functionality

**ngFor** – for loop functionality

…

@Component({

selector: ‘my-app’,

template: `<h1>{{title}}</h1>

<ul>

<li \*ngFor=”let row of rows”>

<h2>{{row.name}}</h2>

<p>{{row.description}}</p>

<p \*ngIf=”row.age < 41”>{{row.age}}</p>

<p \*ngIf=”row.age > 40”>over 40</p>

</li>

</ul>`

})

class AppComponent {

title = ‘hello world’;

rows = [{

“name”: “Jimmy Buffet”,

“age”: 70,

“description”: “Son of a son of a sailor.”

},{

“name”: “Todd Bagley”,

“age”: 46,

“description”: “Son of a son of a conductor.”

}];

}

…

**PIPES** – documentation available at https://angular.io/docs

**Docs** – goto the docs

**API** – click API in the left hand menu

**Filter** – search for “pipes”

**CLASS METHOD** – method / function

…

@Component({

selector: ‘my-app’,

template: `<p>{{totalRowsOver40()}}</p>`

})

class AppComponent {

rows = [{

“name”: “Jimmy Buffet”,

“age”: 70,

“description”: “Son of a son of a sailor.”

},{

“name”: “Todd Bagley”,

“age”: 46,

“description”: “Son of a son of a conductor.”

}];

totalRowsOver40() {

let sum = 0 ;

for (let row of this.rows) {

if(row.age>40){

sum++;

}

}

return sum;

}

}

…

**IMPORTING & EXPORTING** – main.ts, ./app.component.ts, ages.component.ts

***main.ts***

import { AppComponent } from ‘./app.component’;

import { AgesComponent } from ‘./ages.component’;

import { NgModule, Component } from ‘@angular/core’;

import { BrowserModule } from ‘@angular/platform-browser’;

import { platformBrowserDynamic } from ‘@angular/platform-browser-dynamic’;

@NgModule({

declarations:[ AppComponent, Ages ],

imports: [ BrowserModule ],

bootstrap: [ AppComponent ]

})

class AppModule { }

platformBrowserDynamic()

.bootstrapModule(AppModule);

***app.component.ts***

import { Component } from ‘@angular/core’;

@Component({

selector: ‘my-app’,

template: `<h1>{{title}}</h1>

<ages></ages>`

})

export class AppComponent {

title = ”hello world”;

}

***ages.component.ts***

import { Component } from ‘@angular/core’;

@Component({

selector: ‘ages’,

templateUrl: `app/ages.html`,

styleUrls: [`app/ages.css`]

})

export class AgesComponent {

rows = [{

“name”: “Jimmy Buffet”,

“age”: 70,

“description”: “Son of a son of a sailor.”

},{

“name”: “Todd Bagley”,

“age”: 46,

“description”: “Son of a son of a conductor.”

}];

totalRowsOver40() {

let sum = 0 ;

for (let row of this.rows) {

if(row.age>40){

sum++;

}

}

return sum;

}

}

***ages.html***

<ul>

<li \*ngFor=”let age of ages” >

<img [alt]=”age.description” [src]=”age.img”>

<h2>{{age.name | uppercase}}</h2>

<p>Age: {{age.age}}</p>

<p>Birthdate: {{age.birthdate}}</p>

<div [hidden]=”!age.adult”>Child</div>

<button (click)=”invite(age)” [disabled]=”!age.adult” >Invite</button>

</li>

</ul>

<p class=”age”>{{totalRowsOver40()}}</p>

***ages.css***

.age {

color: #808080;

font-size: small;

}

**MOCKS & MODELS** – ages.component.ts, age.ts, ages.mocks.ts

***ages.component.ts***

import { Component } from ‘@angular/core’;

import { Age } from ‘./age’;

import { AGES } from ‘./ages.mocks’;

@Component({

selector: ‘ages’,

templateUrl: `app/ages.html`,

styleUrls: [`app/ages.css`]

})

export class AgesComponent {

rows: Age[];

ngOnInit() {

this.rows = AGES;

}

invite(params) {

if(params.adult) {

// $http service call goes here

}

}

}

***age.ts***

export class Age {

id: number;

name: string;

age: number;

birthdate: Date;

description: string;

adult: boolean;

img: string

}

***ages.mocks.ts***

export const AGES: Age[] =[{

“id”: 1,

“name”: “Jimmy Buffet”,

“age”: 70,

“description”: “Son of a son of a sailor.”,

“adult”: true,

“img”: ‘/res/img/ship.gif’

},{

“id”: 2,

“name”: “Todd Bagley”,

“age”: 46,

“description”: “Son of a son of a conductor.”,

“adult”: true,

“img”: ‘/res/img/train.gif’

}];

**ngMODEL** – [()] = “banana in a box” = two-way data binding

import { FormsModule } from ‘@angular/forms’;

@NgModule({

declarations:[ AppComponent, Ages ],

imports: [ BrowserModule , **FormsModule** ],

bootstrap: [ AppComponent ]

})

…

<input [(ngModel)]=”<must be a data property>” >

**ARCHITECTURE FLOW** – how the app is being processed by the engine

1. index.html
2. res
   1. img
      1. img.gif
   2. css
      1. style.css
3. app
   1. main.ts
   2. app.component.ts
   3. ages.component.ts
      1. age.ts
      2. ages.mocks.ts
      3. ages.html
      4. ages.css

**SERVICE CLASSES** – ages.data.ts, ages.component.ts

***ages.data.ts***

import { AGES } from ‘./ages.mocks’;

export class AgesData {

getAges() {

return AGES;

}

}

***ages.component.ts***

import { AgesData } from ‘./ages.data’;

…

export class AgesComponent {

rows: Age[];

ngOnInit() {

let agesData = new AgesData();

this.ages = agesData.getAges();

}

}

**DEPENDENCY INJECTION** – in charge of knowing how/when to create and send things

1. add injectable decorator to the “service” class
2. infom injector about “service” by naming it as a “provider”
3. inject dependency into component

step one

***ages.data.ts***

import { AGES } from ‘./ages.mocks’;

**import { Injectable } from ‘@angular/core’;**

**@Injectable()**

export class AgesData {

getAges() {

return AGES;

}

}

step two

***main.ts***

…

**import { AgesData } from ‘.ages.data’;**

@NgModule({

declarations:[ AppComponent, Ages ],

imports: [ BrowserModule ],

bootstrap: [ AppComponent ],

**providers: [ AgesData ]**

})

class AppModule { }

…

step three

***ages.component.ts***

…

**import { AgesData } from ‘./ages.data’;**

@Component([…])

export class AgesComponent {

rows: Age[];

**constructor(private agesData: AgesData) {}**

ngOnInit() {

**this.ages = this.agesData.getAges();**

}

}

**HTTP** – steps to use HTTP library

1. create a JSON file with data
2. ensure app include the libraries it needs to make HTTP calls
3. inform injector about HTTP provider
4. inject dependency into component
5. listen for data returned by the request

step one - ***ages.json***

{

“data”: [

{

“id”: 1,

“name”: “Jimmy Buffet”,

“age”: 70,

“description”: “Son of a son of a sailor.”,

“adult”: true,

“img”: ‘/res/img/ship.gif’

},{

“id”: 2,

“name”: “Todd Bagley”,

“age”: 46,

“description”: “Son of a son of a conductor.”,

“adult”: true,

“img”: ‘/res/img/train.gif’

},

{…},

{…}

]

}

step two - ***libraries***

HTTP provides get, post, put, delete

RxJS (Reactive Extensions) provides adv tooling for http calls

SystemJS includes these libraries by default (see Angular 5-min quickstart).

step three – ***inform injector***

***main.ts***

…

**import { HttpModule } from ‘@angular/http’;**

@NgModule({

declarations:[ AppComponent, Ages ],

imports: [ BrowserModule, FormsModule, **HttpModule** ],

bootstrap: [ AppComponent ],

providers: [ AgesData ]

})

class AppModule { }

…

step four – ***injecting HTTP and using it***

***ages.data.ts***

import { AGES } from ‘./ages.mocks’;

import { Injectable } from ‘@angular/core’;

**import { Http } from ‘@angular/http’;**

**import ‘rxjs/add/operator/map’;**

@Injectable()

export class AgesData {

**constructor(private http: Http) {}**

getAges() {

return this.http.get(‘app/ages.json’)

.map(response => <Age[]>response.json().data);

}

}

step five – ***subscribing to the stream***

***ages.component.ts***

…

export class AgesComponent {

constructor(private agesData: AgesData) {}

ngOnInit() {

**this.agesData.getAges()**

**.subscribe(ages => this.ages = ages)**

}

…

totalRowsOver40() {

let sum = 0 ;

**if(Array.isArray(this.ages)) {**

for (let row of this.ages) {

if(row.age>40){

sum++;

}

}

**}**

return sum;

}

…