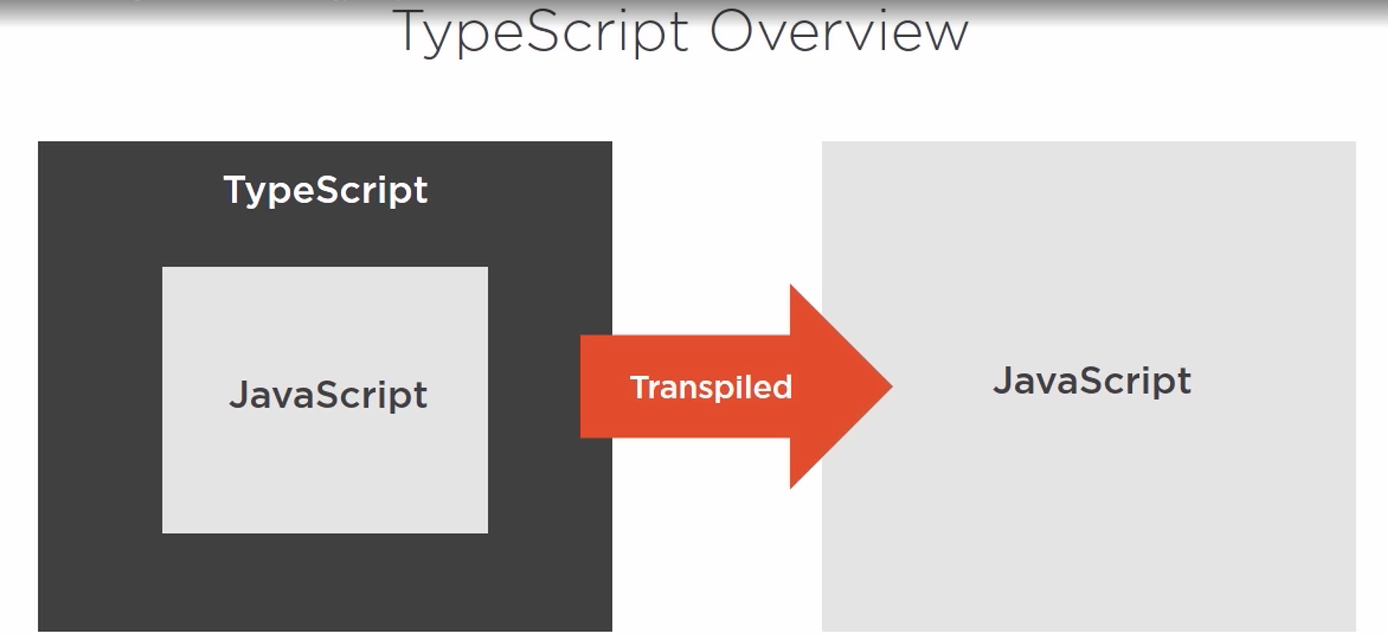
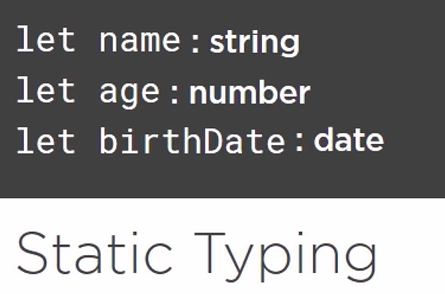
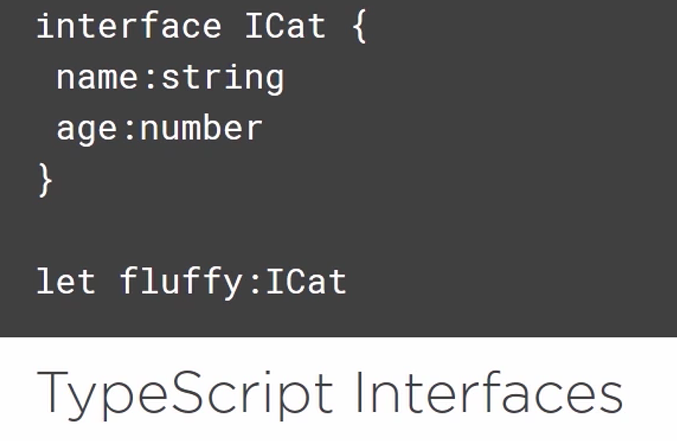
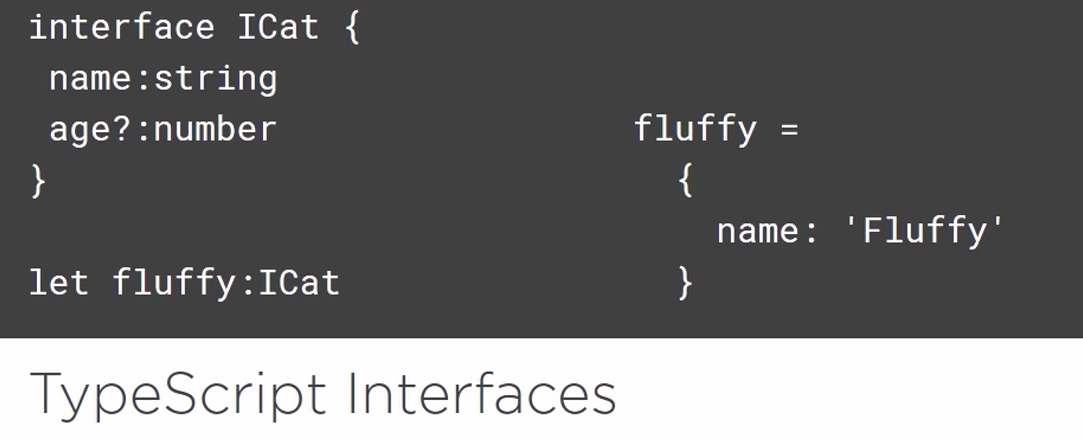
<https://jcoop.io/angular-2-practice-exercises/>



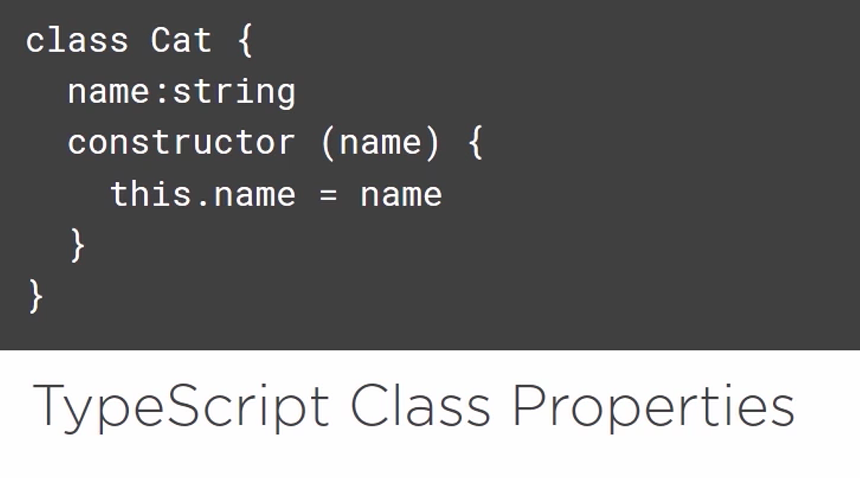
TypeScript is only development environment construct.

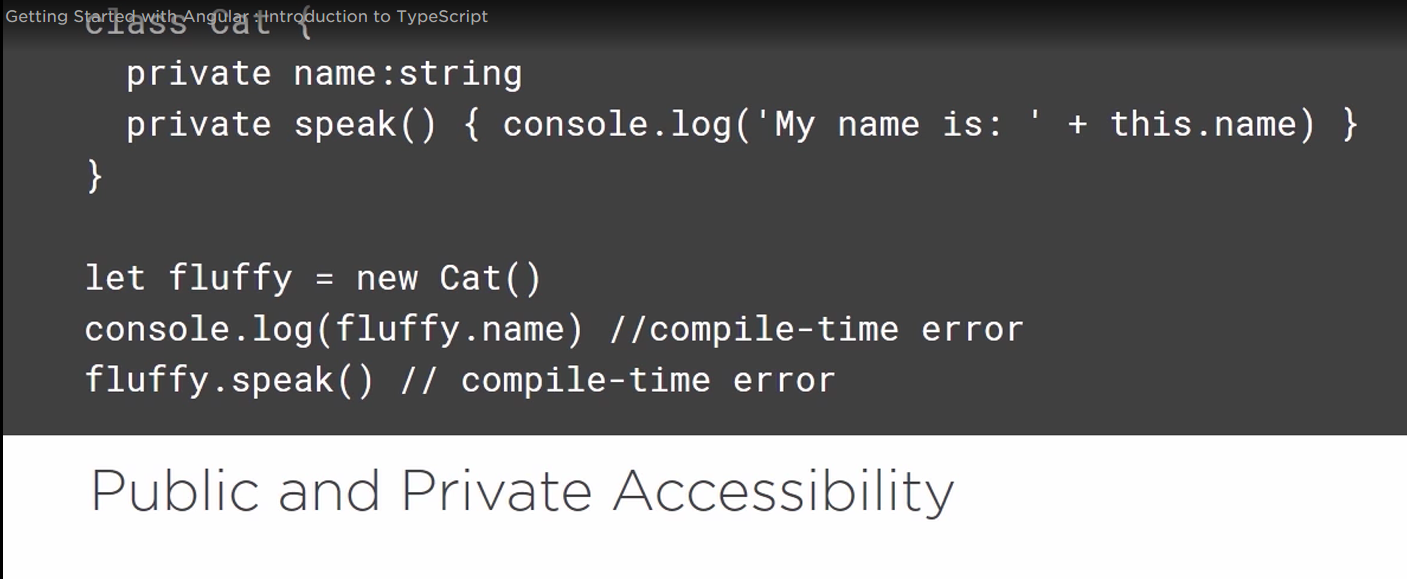


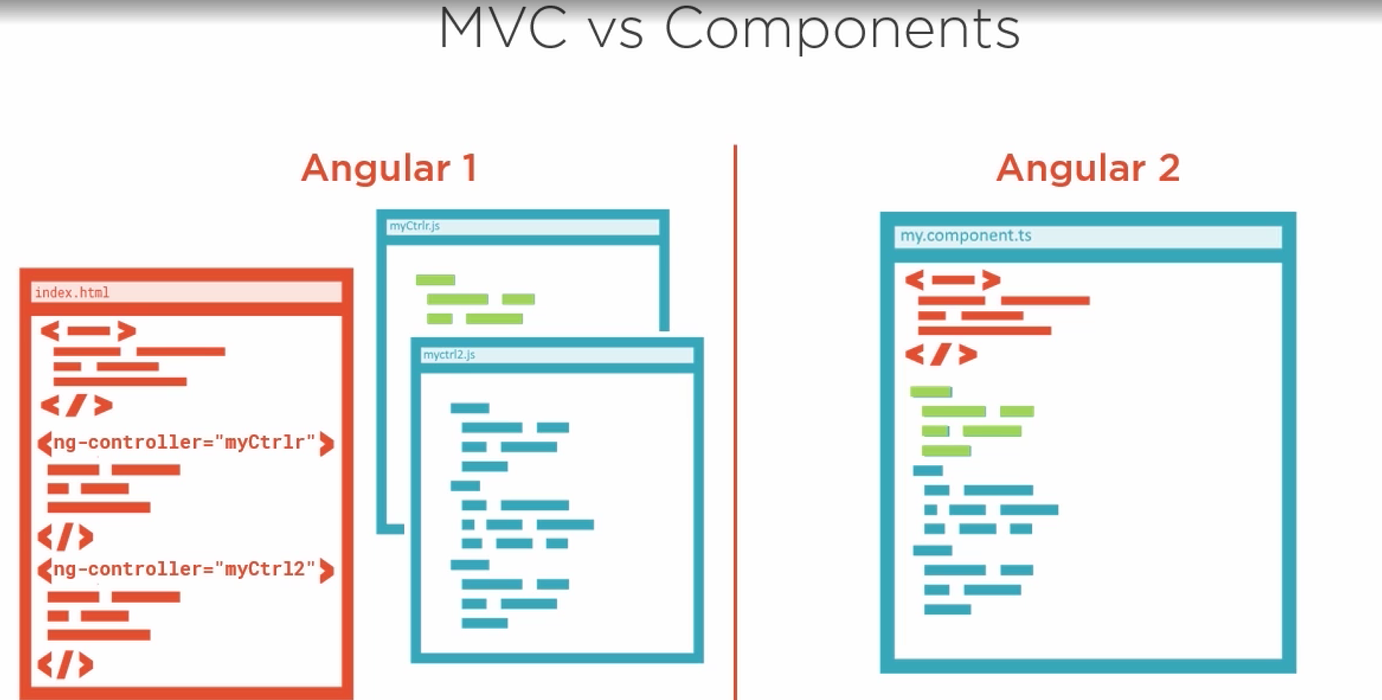


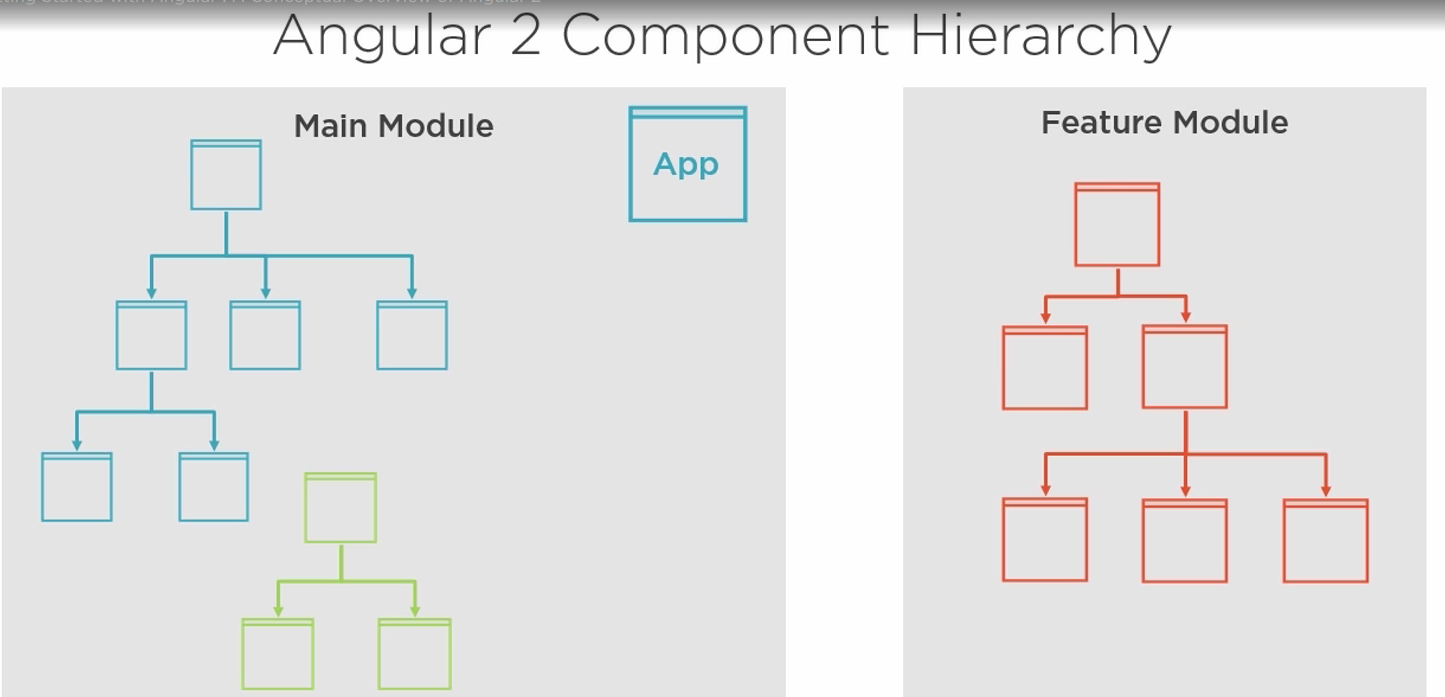


Age will be optional.









Hide those \*.js and \*.js.map file:

{

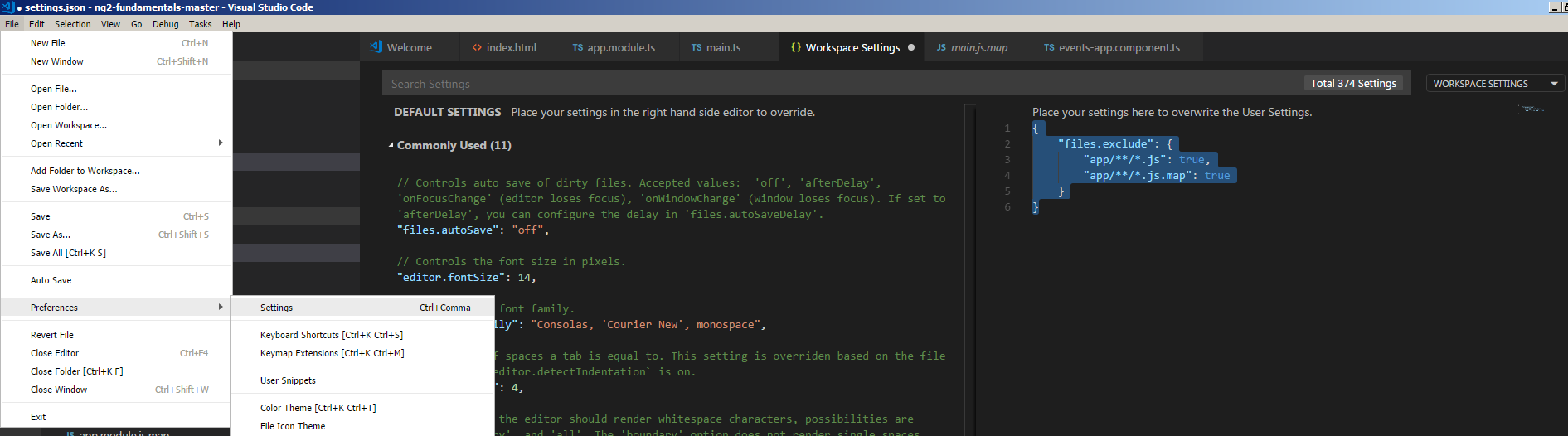
"files.exclude": {

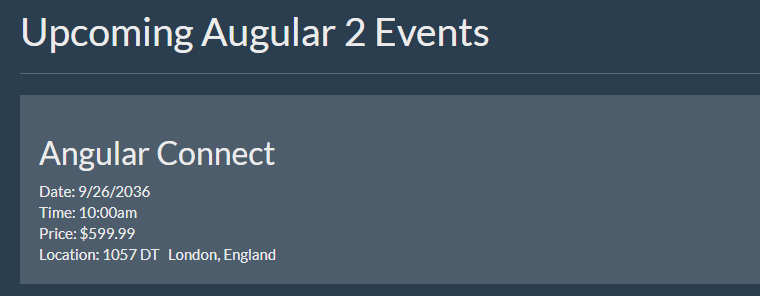
"app/\*\*/\*.js": true,

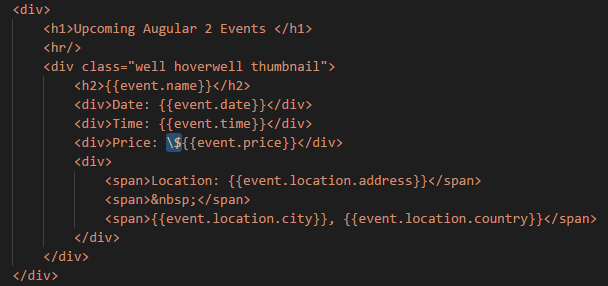
"app/\*\*/\*.js.map": true

}

}







For the price, you need \$ in order to show the $ in the price.



When you moved the html into the file, you don’t need to add \$ any more.







How to pass the data back from the children component to the parent component:

Child component:

import { Component, Input, Output, EventEmitter } from '@angular/core';

@Component({

selector: 'event-thumbnail',

template: `

<div class="well hoverwell thumbnail">

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

<div>Date: {{event.date}}</div>

<div>Time: {{event.time}}</div>

<div>Price: \${{event.price}}</div>

<div>

<span>Location: {{event.location.address}}</span>

<span>&nbsp;</span>

<span>{{event.location.city}}, {{event.location.country}}</span>

</div>

<button class="btn btn-primary" (click)="handleClickMe()">Click me!</button>

</div>

`

})

export class EventThumbnailComponent {

@Input() event: any;

@Output() eventClick = new EventEmitter();

handleClickMe() {

this.eventClick.emit(this.event.name);

}

}

Parent component:

import { Component } from "@angular/core";

@Component({

selector: 'events-list',

template: `

<div>

<h1>Uncomping Angular 2 Events</h1>

<hr />

<event-thumbnail (eventClick)="handleEventClicked($event)" [event]="event1"></event-thumbnail>

</div>

`

})

export class EventsListComponent {

event1 = {

id: 1,

name: 'Angular Connect',

date: '9/26/2036',

time: '10:00am',

price: 599.99,

imageUrl: 'app/assets/images/angularconnect-shield.png',

location: {

address: '1057 DT',

city: 'London',

country: 'England'

}

}

handleEventClicked(data) {

console.log('received:', data);

}

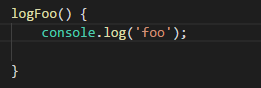
}

* Using Template Reference Variable to Interact with child component:

In the parent component add #variable name to reference the template variable:



In the child component, add public method:



We can call the public method in the parent component:



In the child component, define somProperty.

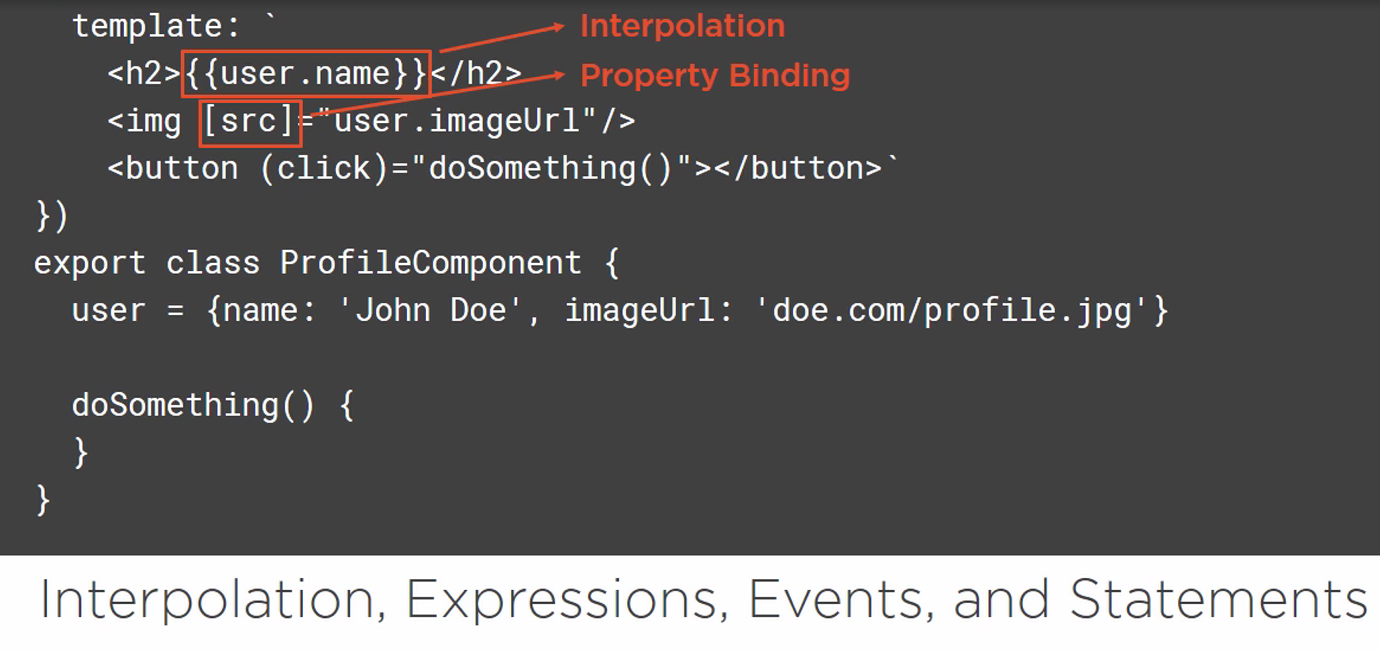


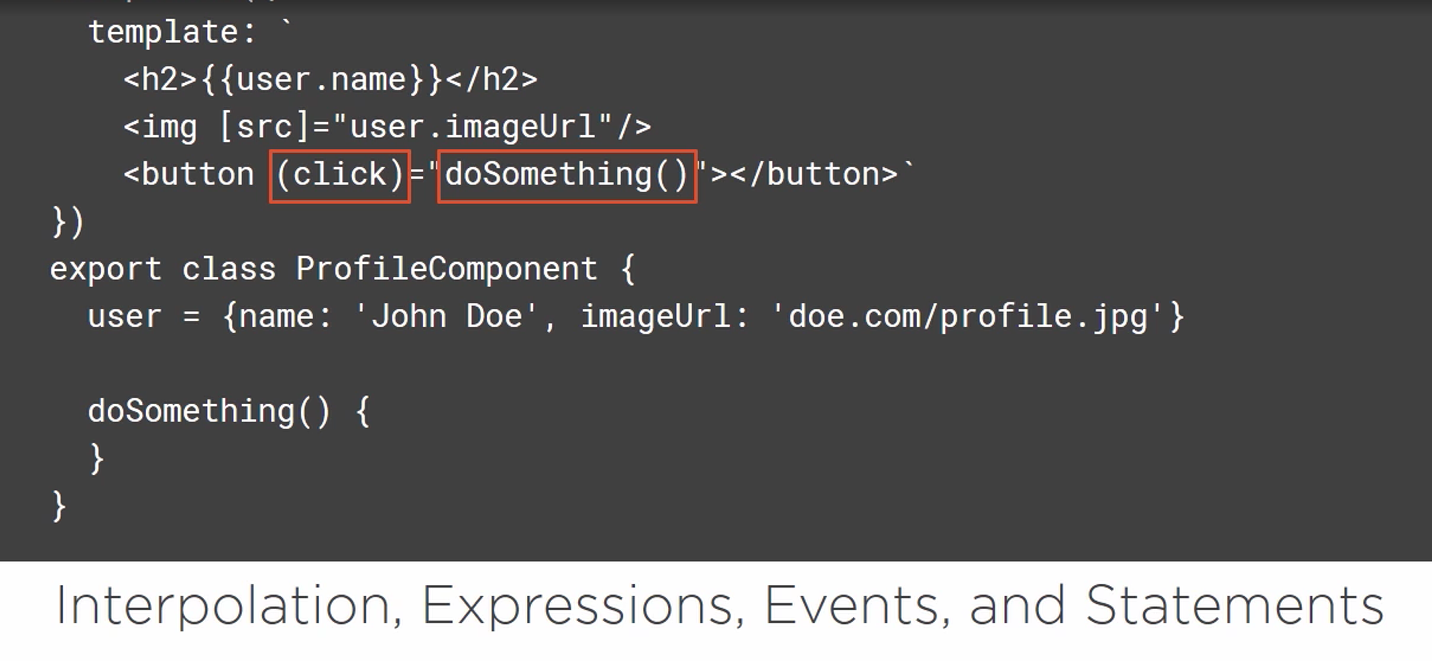
We can call this property in the parent component:



* Adding a site Header:
* <div class="navbar navbar-default">
* <div class="container-fluid">
* <div class="navbar-header">
* <a class="navbar-brand" >ngEvents</a>
* </div>
* <div class="collapse navbar-collapse">
* <ul class="nav navbar-nav">
* <li>
* <a >All Events</a>
* </li>
* <li><a href="">Create Event</a></li>
* <li class="dropdown">
* <a href="#" class="dropdown-toggle" data-toggle="dropdown" >
* Events
* <span class="caret"></span>
* </a>
* <ul class="dropdown-menu">
* <li >
* <a href="">Angular Connect</a>
* </li>
* </ul>
* </li>
* </ul>
* <div class="navbar-header navbar-right">
* <ul class="nav navbar-nav">
* <li>
* <a>Welcome John</a>
* </li>
* </ul>
* </div>
* <form id="searchForm" class="navbar-form navbar-right" >
* <div class="form-group">
* <input type="text" class="form-control" placeholder="Search Sessions" >
* </div>
* <button class="btn btn-default" >
* Search
* </button>
* </form>
* </div>
* </div>
* </div>

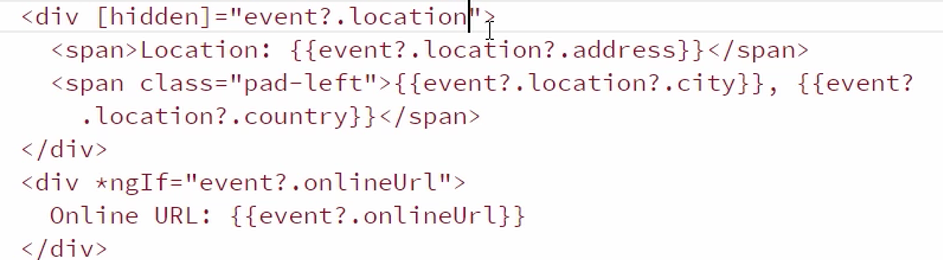




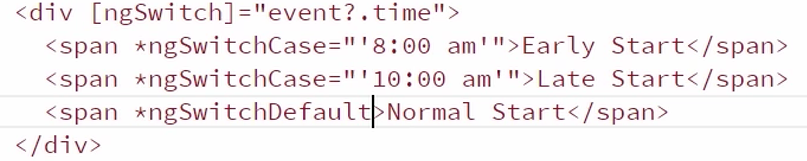


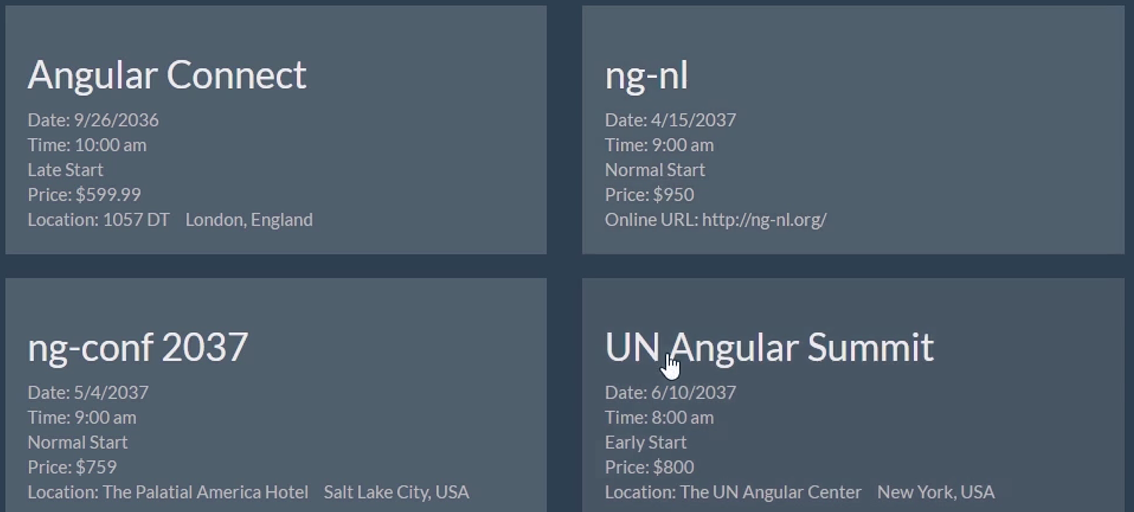
ngIf will complete remove the element from the DOM.

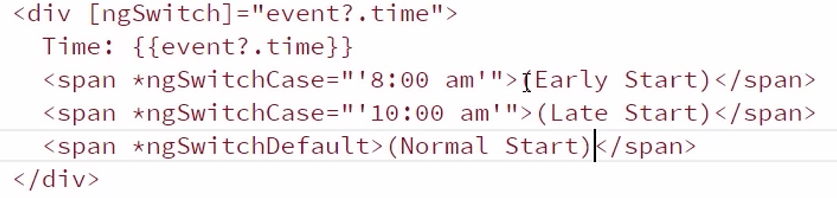
How to hide the element in the Angular?



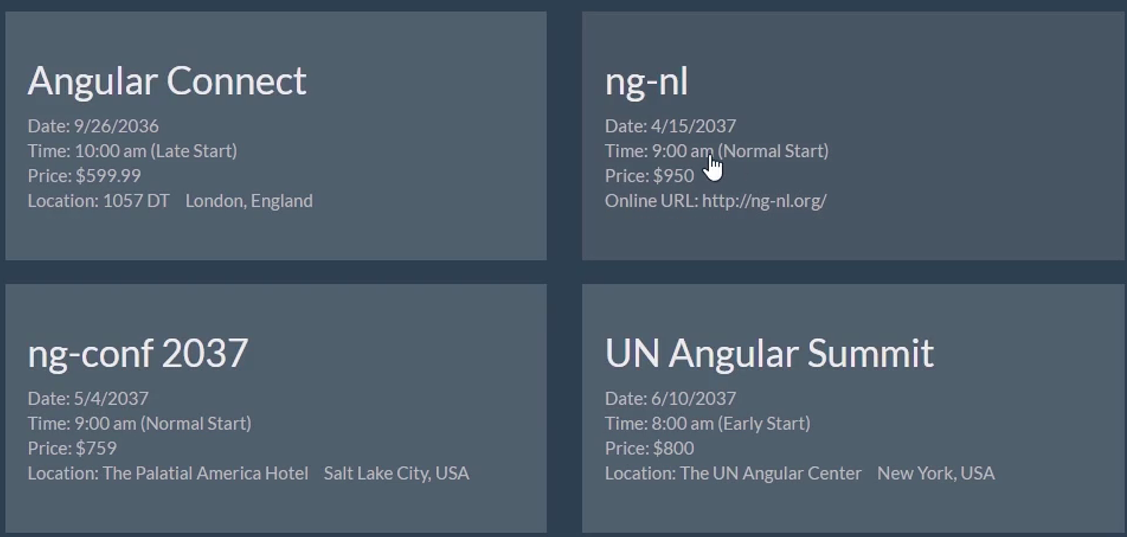
ngSwith:



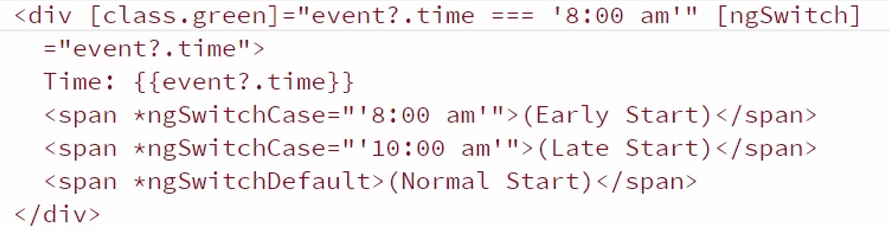


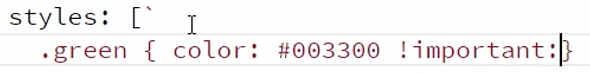


Show them in the same line:

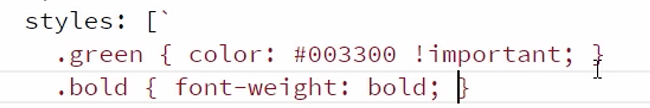


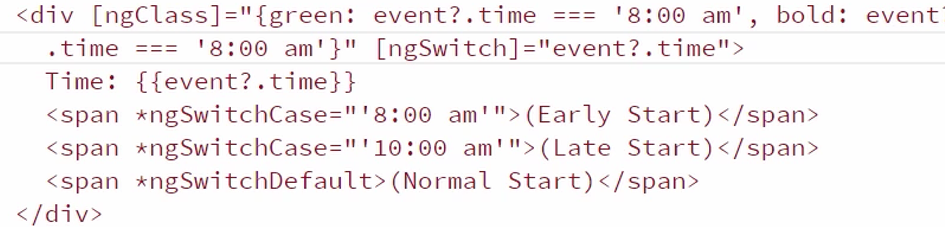
* Class binding:



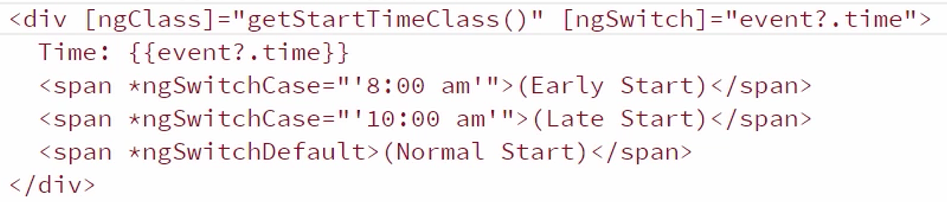


We want to bind two class, we will use ngClass:



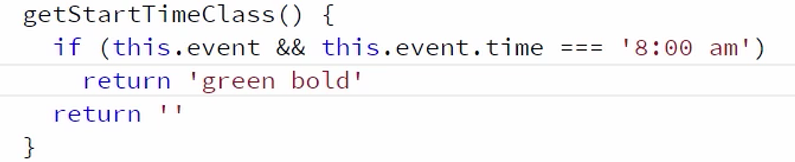


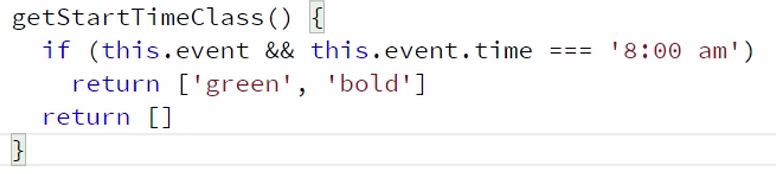
We will use the function instead:



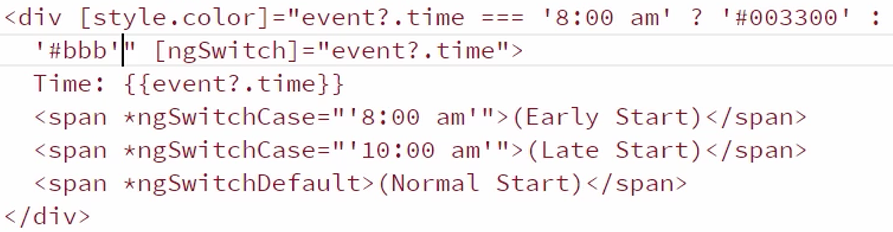


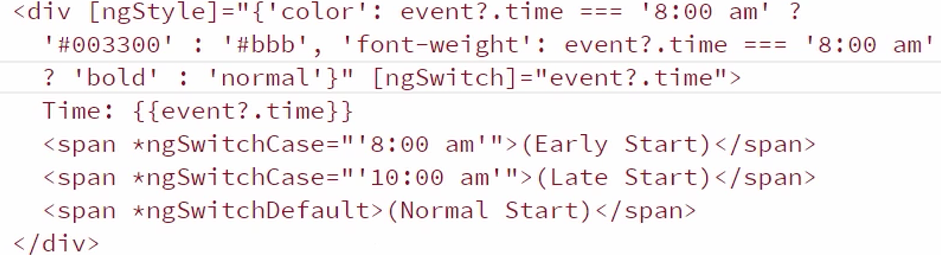
Or



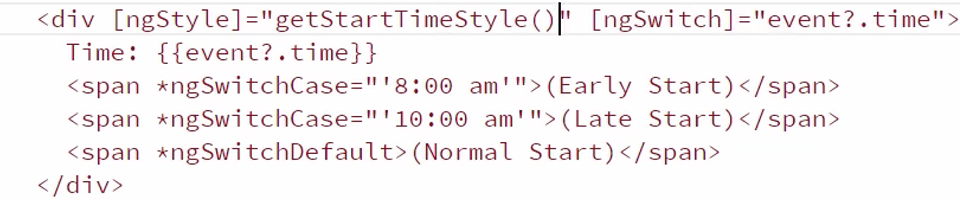
Or 

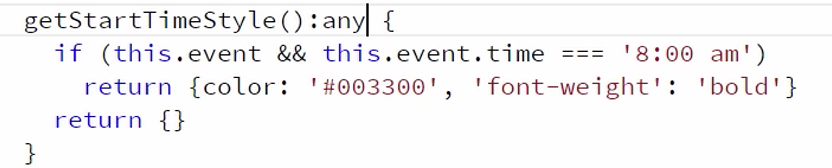
* ngStyle:





Or use the function:





* Angular Services:

Services allow you to define business logic in a separate file and then inject whatever service we need whenever we need it.



Event.service.ts:

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

@Injectable()

export class EventService {

getEvents() {

return EVENTS;

}

}

const EVENTS = [

{

id: 1,

name: 'Angular Connect',

date: '9/26/2036',

time: '10:00 am',

price: 599.99,

imageUrl: '/app/assets/images/angularconnect-shield.png',

location: {

address: '1057 DT',

city: 'London',

country: 'England'

},

sessions: [

{

id: 1,

name: "Using Angular 4 Pipes",

presenter: "Peter Bacon Darwin",

duration: 1,

level: "Intermediate",

abstract: `Learn all about the new pipes in Angular 4, both

how to write them, and how to get the new AI CLI to write

them for you. Given by the famous PBD, president of Angular

University (formerly Oxford University)`,

voters: ['bradgreen', 'igorminar', 'martinfowler']

},

{

id: 2,

name: "Getting the most out of your dev team",

presenter: "Jeff Cross",

duration: 1,

level: "Intermediate",

abstract: `We all know that our dev teams work hard, but with

the right management they can be even more productive, without

overworking them. In this session I'll show you how to get the

best results from the talent you already have on staff.`,

voters: ['johnpapa', 'bradgreen', 'igorminar', 'martinfowler']

},

{

id: 3,

name: "Angular 4 Performance Metrics",

presenter: "Rob Wormald",

duration: 2,

level: "Advanced",

abstract: `Angular 4 Performance is hot. In this session, we'll see

how Angular gets such great performance by preloading data on

your users devices before they even hit your site using the

new predictive algorithms and thought reading software

built into Angular 4.`,

voters: []

},

{

id: 4,

name: "Angular 5 Look Ahead",

presenter: "Brad Green",

duration: 2,

level: "Advanced",

abstract: `Even though Angular 5 is still 6 years away, we all want

to know all about it so that we can spend endless hours in meetings

debating if we should use Angular 4 or not. This talk will look at

Angular 6 even though no code has yet been written for it. We'll

look at what it might do, and how to convince your manager to

hold off on any new apps until it's released`,

voters: []

},

{

id: 5,

name: "Basics of Angular 4",

presenter: "John Papa",

duration: 2,

level: "Beginner",

abstract: `It's time to learn the basics of Angular 4. This talk

will give you everything you need to know about Angular 4 to

get started with it today and be building UI's for your self

driving cars and butler-bots in no time.`,

voters: ['bradgreen', 'igorminar']

}

]

},

{

id: 2,

name: 'ng-nl',

date: '4/15/2037',

time: '9:00 am',

price: 950.00,

imageUrl: '/app/assets/images/ng-nl.png',

location: {

address: 'The NG-NL Convention Center & Scuba Shop',

city: 'Amsterdam',

country: 'Netherlands'

},

sessions: [

{

id: 1,

name: "Testing Angular 4 Workshop",

presenter: "Pascal Precht & Christoph Bergdorf",

duration: 4,

level: "Beginner",

abstract: `In this 6 hour workshop you will learn not only how to test Angular 4,

you will also learn how to make the most of your team's efforts. Other topics

will be convincing your manager that testing is a good idea, and using the new

protractor tool for end to end testing.`,

voters: ['bradgreen', 'igorminar']

},

{

id: 2,

name: "Angular 4 and Firebase",

presenter: "David East",

duration: 3,

level: "Intermediate",

abstract: `In this workshop, David East will show you how to use Angular with the new

ultra-real-time 5D Firebase back end, hosting platform, and wine recommendation engine.`,

voters: ['bradgreen', 'igorminar', 'johnpapa']

},

{

id: 3,

name: "Reading the Angular 4 Source",

presenter: "Patrick Stapleton",

duration: 2,

level: "Intermediate",

abstract: `Angular 4's source code may be over 25 million lines of code, but it's really

a lot easier to read and understand then you may think. Patrick Stapleton will talk

about his secretes for keeping up with the changes, and navigating around the code.`,

voters: ['martinfowler']

},

{

id: 4,

name: "Hail to the Lukas",

presenter: "Lukas Ruebbelke",

duration: 1,

level: "Beginner",

abstract: `In this session, Lukas will present the

secret to being awesome, and how he became the President

of the United States through his amazing programming skills,

showing how you too can be success with just attitude.`,

voters: ['bradgreen']

},

]

},

{

id: 3,

name: 'ng-conf 2037',

date: '5/4/2037',

time: '9:00 am',

price: 759.00,

imageUrl: '/app/assets/images/ng-conf.png',

location: {

address: 'The Palatial America Hotel',

city: 'Salt Lake City',

country: 'USA'

},

sessions: [

{

id: 1,

name: "How Elm Powers Angular 4",

presenter: "Murphy Randle",

duration: 2,

level: "Intermediate",

abstract: `We all know that Angular is written in Elm, but did you

know how the source code is really written? In this exciting look

into the internals of Angular 4, we'll see exactly how Elm powers

the framework, and what you can do to take advantage of this knowledge.`,

voters: ['bradgreen', 'martinfowler', 'igorminar']

},

{

id: 2,

name: "Angular and React together",

presenter: "Jamison Dance",

duration: 2,

level: "Intermediate",

abstract: `React v449.6 has just been released. Let's see how to use

this new version with Angular to create even more impressive applications.`,

voters: ['bradgreen', 'martinfowler']

},

{

id: 3,

name: "Redux Woes",

presenter: "Rob Wormald",

duration: 1,

level: "Intermediate",

abstract: `Everyone is using Redux for everything from Angular to React to

Excel macros, but you're still having trouble grasping it? We'll take a look

at how farmers use Redux when harvesting grain as a great introduction to

this game changing technology.`,

voters: ['bradgreen', 'martinfowler', 'johnpapa']

},

{

id: 4,

name: "ng-wat again!!",

presenter: "Shai Reznik",

duration: 1,

level: "Beginner",

abstract: `Let's take a look at some of the stranger pieces of Angular 4,

including neural net nets, Android in Androids, and using pipes with actual pipes.`,

voters: ['bradgreen', 'martinfowler', 'igorminar', 'johnpapa']

},

{

id: 5,

name: "Dressed for Success",

presenter: "Ward Bell",

duration: 2,

level: "Beginner",

abstract: `Being a developer in 2037 is about more than just writing bug-free code.

You also have to look the part. In this amazing expose, Ward will talk you through

how to pick out the right clothes to make your coworkers and boss not only

respect you, but also want to be your buddy.`,

voters: ['bradgreen', 'martinfowler']

},

{

id: 6,

name: "These aren't the directives you're looking for",

presenter: "John Papa",

duration: 2,

level: "Intermediate",

abstract: `Coinciding with the release of Star Wars Episode 18, this talk will show how

to use directives in your Angular 4 development while drawing lessons from the new movie,

featuring all your favorite characters like Han Solo's ghost and Darth Jar Jar.`,

voters: ['bradgreen', 'martinfowler']

},

]

},

{

id: 4,

name: 'UN Angular Summit',

date: '6/10/2037',

time: '8:00 am',

price: 800.00,

imageUrl: '/app/assets/images/basic-shield.png',

location: {

address: 'The UN Angular Center',

city: 'New York',

country: 'USA'

},

sessions: [

{

id: 1,

name: "Diversity in Tech",

presenter: "Sir Dave Smith",

duration: 2,

level: "Beginner",

abstract: `Yes, we all work with cyborgs and androids and Martians, but

we probably don't realize that sometimes our internal biases can make it difficult for

these well-designed coworkers to really feel at home coding alongside us. This talk will

look at things we can do to recognize our biases and counteract them.`,

voters: ['bradgreen', 'igorminar']

},

{

id: 2,

name: "World Peace and Angular",

presenter: "US Secretary of State Zach Galifianakis",

duration: 2,

level: "Beginner",

abstract: `Angular has been used in most of the major peace brokering that has

happened in the last decade, but there is still much we can do to remove all

war from the world, and Angular will be a key part of that effort.`,

voters: ['bradgreen', 'igorminar', 'johnpapa']

},

{

id: 3,

name: "Using Angular with Androids",

presenter: "Dan Wahlin",

duration: 3,

level: "Advanced",

abstract: `Androids may do everything for us now, allowing us to spend all day playing

the latest Destiny DLC, but we can still improve the massages they give and the handmade

brie they make using Angular 4. This session will show you how.`,

voters: ['igorminar', 'johnpapa']

},

]

},

{

id: 5,

name: 'ng-vegas',

date: '2/10/2037',

time: '9:00 am',

price: 400.00,

imageUrl: '/app/assets/images/ng-vegas.png',

location: {

address: 'The Excalibur',

city: 'Las Vegas',

country: 'USA'

},

sessions: [

{

id: 1,

name: "Gambling with Angular",

presenter: "John Papa",

duration: 1,

level: "Intermediate",

abstract: `No, this talk isn't about slot machines. We all know that

Angular is used in most waiter-bots and coke vending machines, but

did you know that was also used to write the core engine in the majority

of voting machines? This talk will look at how all presidential elections

are now determined by Angular code.`,

voters: ['bradgreen', 'igorminar']

},

{

id: 2,

name: "Angular 4 in 60ish Minutes",

presenter: "Dan Wahlin",

duration: 2,

level: "Beginner",

abstract: `Get the skinny on Angular 4 for anyone new to this great new technology.

Dan Wahlin will show you how you can get started with Angular in 60ish minutes,

guaranteed!`,

voters: ['bradgreen', 'igorminar', 'johnpapa']

}

]

}

];

App.module.ts: add Providers

import { NgModule } from '@angular/core';

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

import { EventsAppComponent } from './events-app.component';

import { EventsListComponent } from './events/events-list.component';

import { EventThumbnailComponent } from './events/event-thumbnail.component';

import { NavBarComponent } from './nav/navbar.component';

import { EventService } from './events/shared/event.service';

@NgModule({

imports: [BrowserModule],

declarations: [

EventsListComponent,

EventsAppComponent,

EventThumbnailComponent,

NavBarComponent],

providers: [EventService],

bootstrap: [EventsAppComponent]

})

export class AppModule {

}

Events-list.component.ts:

import { Component } from "@angular/core";

import { EventService } from "./shared/event.service";

@Component({

selector: 'events-list',

template: `

<div>

<h1>Uncomping Angular 2 Events</h1>

<hr />

<div class="row">

<div \*ngFor="let event of events" class="col-md-5">

<event-thumbnail [event]="event"></event-thumbnail>

</div>

</div>

</div>

`

})

export class EventsListComponent {

events:any[];

constructor(private eventService: EventService) {

this.events = this.eventService.getEvents();

}

}

It is a not a good idea to put getEvents in our constructor that are potentially long-running and eventually this will be an AJAX call, and it will take while to fetch those events. We need to have this happen when the component first loads. Components have lifecycle hooks that you can hook into and one of those is the ngOnInit method so that lifecycle event is called when the component is being loaded.

import { Component, OnInit } from "@angular/core";

import { EventService } from "./shared/event.service";

@Component({

selector: 'events-list',

template: `

<div>

<h1>Uncomping Angular 2 Events</h1>

<hr />

<div class="row">

<div \*ngFor="let event of events" class="col-md-5">

<event-thumbnail [event]="event"></event-thumbnail>

</div>

</div>

</div>

`

})

export class EventsListComponent implements OnInit{

events:any[];

constructor(private eventService: EventService) {

}

ngOnInit() {

this.events = this.eventService.getEvents();

}

}

* Wrapping third party services into an injectable service in Angular:

<http://codeseven.github.io/toastr/demo.html>

Toaster is a JavaScript library that allows you to create pop-up notification message like this.

1. Let’s install it first:

Npm install toastr –save

1. Import it into index.html:

<!DOCTYPE html>

<html>

<head>

<title>ng Events</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="node\_modules/toastr/build/toastr.min.css">

<link rel="stylesheet" href="node\_modules/ng2f-bootstrap/dist/bootstrap.min.css">

<link rel="stylesheet" href="styles.css">

<!-- Polyfill(s) for older browsers -->

<script src="node\_modules/core-js/client/shim.min.js"></script>

<script src="node\_modules/jquery/dist/jquery.min.js"></script>

<script src="node\_modules/toastr/build/toastr.min.js"></script>

<script src="node\_modules/bootstrap/dist/js/bootstrap.js"></script>

<script src="node\_modules/zone.js/dist/zone.js"></script>

<script src="node\_modules/systemjs/dist/system.src.js"></script>

<script src="systemjs.config.js"></script>

<script>

System.import('app').catch(function(err){console.error(err);});

</script>

</head>

<body class="container">

<events-app></events-app>

</body>

<script>toastr.success('This is a toast.')</script>

</html>

1. Create toastr.service.ts in the app/common folder:

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

declare let toastr: any;

@Injectable()

export class ToastrService {

success (message: string, title?: string) {

toastr.success(message, title);

}

info(message: string, title?: string) {

toastr.info(message, title);

}

warning(message: string, title?: string) {

toastr.warning(message, title);

}

error(message: string, title?: string) {

toastr.error(message, title);

}

}

1. Register in the app.module.ts file as Providers:

import { NgModule } from '@angular/core';

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

import { EventsAppComponent } from './events-app.component';

import { EventsListComponent } from './events/events-list.component';

import { EventThumbnailComponent } from './events/event-thumbnail.component';

import { NavBarComponent } from './nav/navbar.component';

import { EventService } from './events/shared/event.service';

import { ToastrService } from './common/toastr.service';

@NgModule({

imports: [BrowserModule],

declarations: [

EventsListComponent,

EventsAppComponent,

EventThumbnailComponent,

NavBarComponent],

providers: [EventService, ToastrService],

bootstrap: [EventsAppComponent]

})

export class AppModule {

}

1. Call it in the events-list.component.ts:

import { Component, OnInit } from "@angular/core";

import { EventService } from "./shared/event.service";

import { ToastrService } from "../common/toastr.service";

@Component({

selector: 'events-list',

template: `

<div>

<h1>Uncomping Angular 2 Events</h1>

<hr />

<div class="row">

<div \*ngFor="let event of events" class="col-md-5">

<event-thumbnail (click)="handleThumnailClick(event.name)"

[event]="event"></event-thumbnail>

</div>

</div>

</div>

`

})

export class EventsListComponent implements OnInit{

events:any[];

constructor(private eventService: EventService, private toastr: ToastrService) {

}

ngOnInit() {

this.events = this.eventService.getEvents();

}

handleThumnailClick(eventName) {

this.toastr.success(eventName);

}

}

* Routing and Navigating pages:

<https://github.com/jmcooper/angular2-fundamentals-files>

* Add <router-outlet> to Events-app.component.ts:

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

@Component({

selector: 'events-app',

template: ` <nav-bar></nav-bar>

<router-outlet></router-outlet>

`

})

export class EventsAppComponent {

}

* Define each routes for the components in the routes.ts file:

import { Routes } from '@angular/router';

import { EventsListComponent } from "./events/events-list.component";

import { EventDetailsComponent } from "./events/event-details/event-details.component";

export const appRoutes: Routes = [

{path: 'events', component: EventsListComponent},

{path: 'events/:id', component: EventDetailsComponent},

{path: '', redirectTo: '/events', pathMatch: 'full'}

]

:id parameter placeholder /events/1 or /events/foo

Path: ‘’: default router pathMatch: prefix or full. Prefix means redirect if the URL starts with the specified path string and full means redirect if it fully matches the specified path string.

* Include the routes to the app.module.ts:

import { NgModule } from '@angular/core';

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

import { EventsAppComponent } from './events-app.component';

import { EventsListComponent } from './events/events-list.component';

import { EventThumbnailComponent } from './events/event-thumbnail.component';

import { NavBarComponent } from './nav/navbar.component';

import { EventService } from './events/shared/event.service';

import { ToastrService } from './common/toastr.service';

import { EventDetailsComponent } from './events/event-details/event-details.component';

import { RouterModule } from '@angular/router';

import { appRoutes } from './routes';

@NgModule({

imports: [BrowserModule,

RouterModule.forRoot(appRoutes)

],

declarations: [

EventsListComponent,

EventsAppComponent,

EventThumbnailComponent,

EventDetailsComponent,

NavBarComponent],

providers: [EventService, ToastrService],

bootstrap: [EventsAppComponent]

})

export class AppModule {

}

* Add base tag to the index.html:

<!DOCTYPE html>

<html>

<head>

<base href="/">

<title>ng Events</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="node\_modules/toastr/build/toastr.min.css">

<link rel="stylesheet" href="node\_modules/ng2f-bootstrap/dist/bootstrap.min.css">

<link rel="stylesheet" href="styles.css">

<!-- Polyfill(s) for older browsers -->

<script src="node\_modules/core-js/client/shim.min.js"></script>

<script src="node\_modules/jquery/dist/jquery.min.js"></script>

<script src="node\_modules/toastr/build/toastr.min.js"></script>

<script src="node\_modules/bootstrap/dist/js/bootstrap.js"></script>

<script src="node\_modules/zone.js/dist/zone.js"></script>

<script src="node\_modules/systemjs/dist/system.src.js"></script>

<script src="systemjs.config.js"></script>

<script>

System.import('app').catch(function(err){console.error(err);});

</script>

</head>

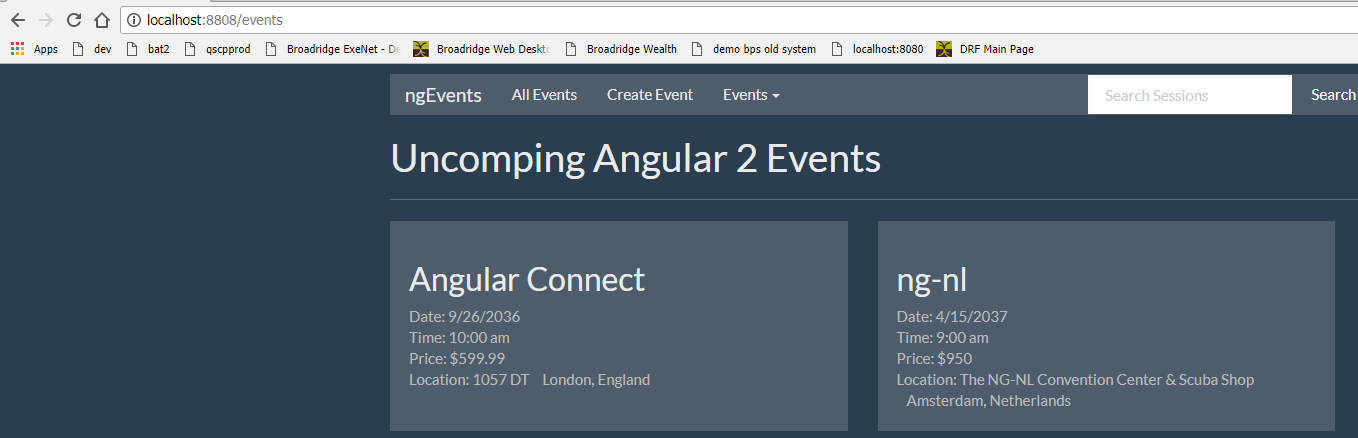
<body class="container">

<events-app></events-app>

</body>

</html>

* We can remove the selector ‘events-list’ from the events-list.component.ts, we don’t need it any more.



Localhost:8808 redirect route to /events.

* Accessing Route Parameters:

Event-details.component.ts:

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

import { EventService } from '../shared/event.service';

import { ActivatedRoute } from '@angular/router';

@Component({

templateUrl: 'app/events/event-details/event-details.component.html',

styles: [`

.container {padding-left: 20px; padding-right: 20px}

.event-image {height: 100px;}

`]

})

export class EventDetailsComponent {

event: any;

constructor(private eventService: EventService, private route:ActivatedRoute) {

}

ngOnInit () {

this.event = this.eventService.getEvent(+this.route.snapshot.params['id']);

}

}

+ getEvent take a number, we will use +

* Linking to Routes:

Event-thumbnail.component.ts:

import { Component, Input, Output, EventEmitter } from '@angular/core';

@Component({

selector: 'event-thumbnail',

template: `

<div [routerLink]="['/events', event.id]" class="well hoverwell thumbnail">

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

<div>Date: {{event.date}}</div>

<div>Time: {{event.time}}</div>

<div>Price: \${{event.price}}</div>

<div>

<span>Location: {{event.location.address}}</span>

<span class="pad-left" >{{event.location.city}}, {{event.location.country}}</span>

</div>

</div>

`,

styles: [`

.thumbnail {min-height: 210px;}

.pad-left {margin-left: 10px;}

.well div {color: #bbb;}

`]

})

export class EventThumbnailComponent {

@Input() event: any;

}

Navbar.component.html:

<div class="navbar navbar-default">

<div class="container-fluid">

<div class="navbar-header">

<a class="navbar-brand" >ngEvents</a>

</div>

<div class="collapse navbar-collapse">

<ul class="nav navbar-nav">

<li>

<a [routerLink]="['events']">All Events</a>

</li>

<li><a href="">Create Event</a></li>

<li class="dropdown">

<a href="#" class="dropdown-toggle" data-toggle="dropdown" >

Events

<span class="caret"></span>

</a>

<ul class="dropdown-menu">

<li >

<a href="">Angular Connect</a>

</li>

</ul>

</li>

</ul>

<div class="navbar-header navbar-right">

<ul class="nav navbar-nav">

<li>

<a>Welcome John</a>

</li>

</ul>

</div>

<form id="searchForm" class="navbar-form navbar-right" >

<div class="form-group">

<input type="text" class="form-control" placeholder="Search Sessions" >

</div>

<button class="btn btn-default" >

Search

</button>

</form>

</div>

</div>

</div>

* Navigating from code:

Create-event.component.ts:

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

import { Router } from '@angular/router';

@Component({

template: `

<h1>New Event </h1>

<hr>

<div class="col-mod-6">

<h3>[Create Event From will go here]</h3>

<br/>

<br/>

<button type="submit" class="btn btn-primary">Save</button>

<button type="button" class="btn btn-default" (click)="cancel()">Cancel</button>

</div>

`

})

export class CreateEventComponent {

constructor(private router: Router) {

}

cancel() {

this.router.navigate(['events']);

}

}