ANGULAR 2.0

RUTHIRAKUMAR

91-9715531887 (RUTHRAN.KUMAR@GMAIL.COM)

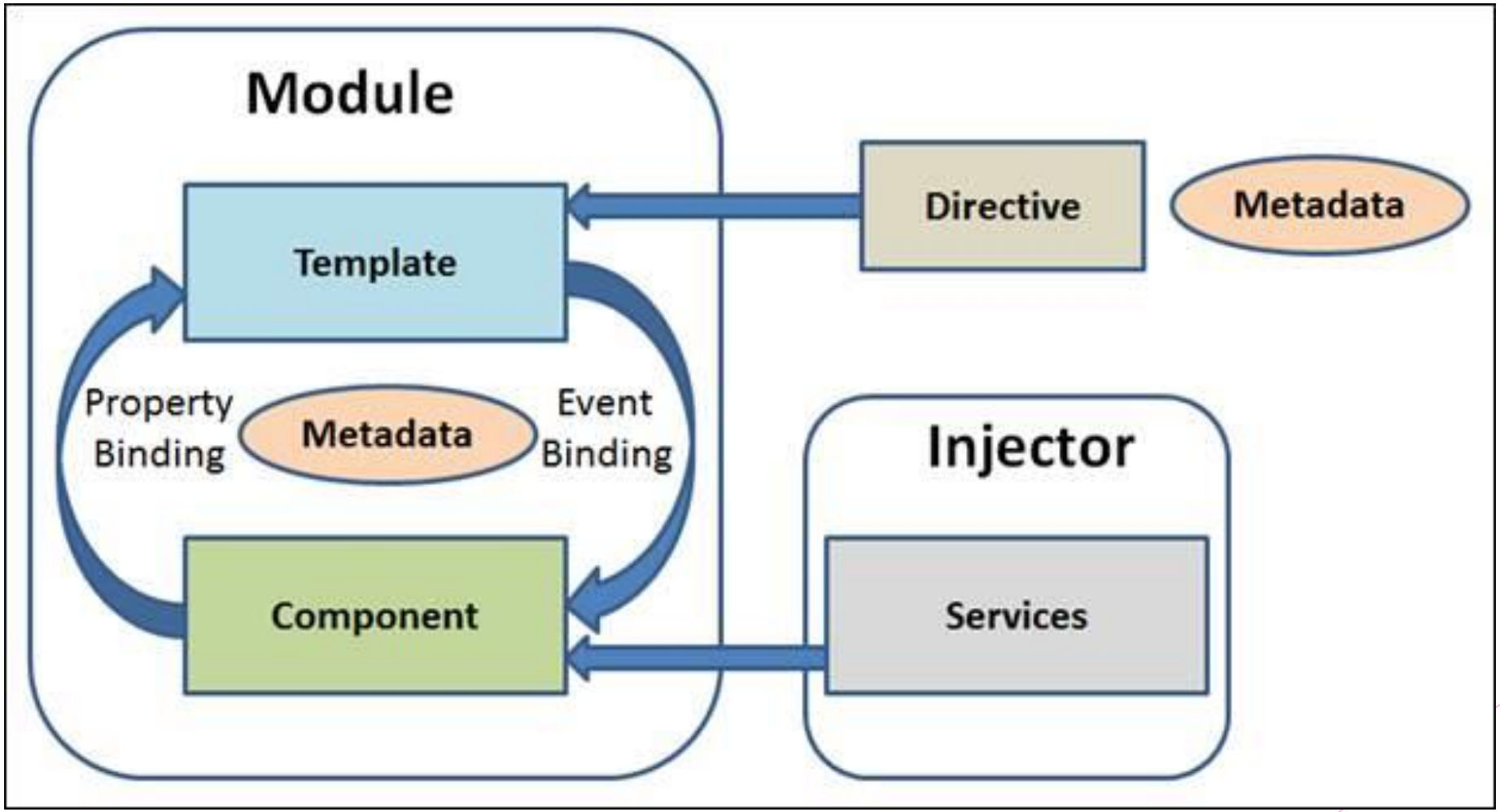
Agenda

- Overview
- Architecture
- Application set-up
- Typescript
- Component
- Template
- Module
- Communication b/w components
- Template Variable
- Expression & Statement
- Structural Directives
- Service
- Type Safety
- Forms &Validations
- ReactiveForms
- Custom Validator
- Content Projection
- Pipes(Inbuilt & Custom)
- Filter & Sorting
- Opaque Token
- Modal
- Directive
- ViewChild
- HTTP
- Unit Testing
- End to End Testing
- Production

Overview

- Open source UI framework launched beta version march 2014 (Google)
- Simpler, Easier and Faster than 1.x
- Web, mobile apps, Supports Responsive and heavily loaded app
- Programming in Typescript (Superset of Javascript)
- Release : 1.x → 2.x → 4 → 5
- Written by Typescript
- Not backward compatible with angular 1.x

Architecture



Application setup

- Node installation
- Quickstart (https://github.com/angular/quickstart)
  - **Npm install**
  - **Npm start (npm run build or npm run build:w)**
  - **Npm test**
  - **Npm run e2e**
- Angular-Cli
  - **Npm install –g @angular/cli**
  - **Ng new projectName**
  - **Cd Projectname**
  - **Ng serve - - open**
  - **Ng test**
  - **Ng e2e**

ng g component my-new-component

ng g directive my-new-directive

ng g pipe my-new-pipe

ng g service my-new-service

ng g interface my-new-interface

ng g enum my-new-enum

ng g module my-module

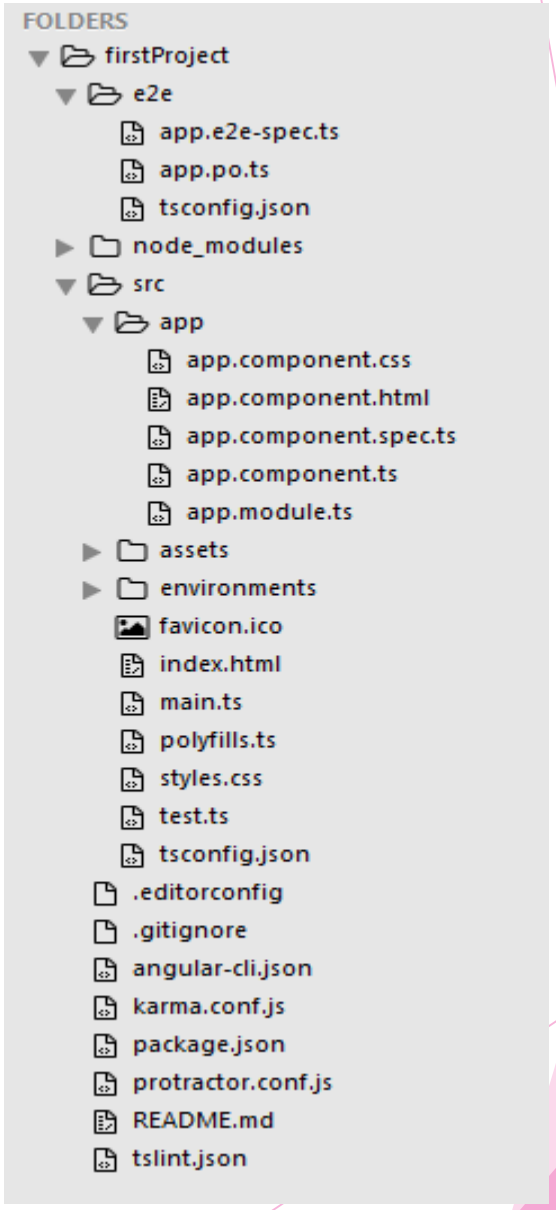
Typescript Overview

- Typed superset of JavaScript that compiles to plain JavaScript.
- Object oriented with classes, interfaces, enum like C# or Java.

```
class Student {
    fullName: string;
    constructor(public firstName, public middleInitial, public lastName) {
        this.fullName = firstName + " " + middleInitial + " " + lastName;
    }
}

interface Person {
    firstName: string;
    lastName: string;
}

function greeter(person : Person) {
    return "Hello, " + person.firstName + " " + person.lastName;
}

var user = new Student("Jane", "M.", "User");
```

Files & Structure

```
D:\Ruthran\OFFICE\Training\Angular JS2\April 15>ng new firstProject
installing ng2
  create .editorconfig
  create README.md
  create src\app\app.component.css
  create src\app\app.component.html
  create src\app\app.component.spec.ts
  create src\app\app.component.ts
  create src\app\app.module.ts
  create src\assets\.gitkeep
  create src\environments\environment.prod.ts
  create src\environments\environment.ts
  create src\favicon.ico
  create src\index.html
  create src\main.ts
  create src\polyfills.ts
  create src\styles.css
  create src\test.ts
  create src\tsconfig.json
  create angular-cli.json
  create e2e\app.e2e-spec.ts
  create e2e\app.po.ts
  create e2e\tsconfig.json
  create .gitignore
  create karma.conf.js
  create package.json
  create protractor.conf.js
  create tslint.json
Successfully initialized git.
Installing packages for tooling via npm.
```



Component & Template

Controller class with a template which deals with view of the application, logic on the page and dependency injection

app.component.ts

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

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

```html
<main class="container">
    <cm-navbar></cm-navbar>
    <router-outlet></router-outlet>
    <cm-growler position="top-right" timeout="2000">
    </cm-growler>
    <cm-modal></cm-modal>
</main>
{{ title }}
```

Module

- library for another module

```
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { FormsModule } from '@angular/forms';
4  import { HttpModule } from '@angular/http';
5
6  import { AppComponent } from './app.component';
7
8  @NgModule({
9    declarations: [
10     AppComponent
11   ],
12   imports: [
13     BrowserModule,
14     FormsModule,
15     HttpModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
20 export class AppModule { }
21
```

Communication B/W Components (Input)

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

@Component({
  selector: 'child',
  template: `<div>{{flight.no}}</div><div>{{flight.name}}</div>`
})
export class ChildComponent  {

  @Input() flight:any;

}
```

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

@Component({
  selector: 'parent',
  template: '<child [flight]="flightDetail"></child>',
  styleUrls: ['./parent.component.css']
})
export class ParentComponent {
public flightDetail:any = {
    no : 12345,
    name :'Air India',
    status : 'Departured'
  }

}
```

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

<parent></parent>
```

app works!

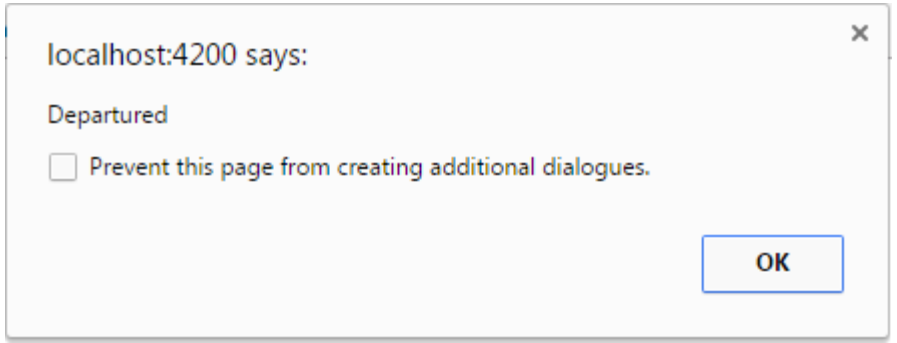
12345
Air India

Communication B/W Components (Output)

```
import { Component, Input, Output, EventEmitter } from '@angular/core';
@Component({
  selector: 'child',
  template: `<div>{{flight.no}}</div><div>{{flight.name}}</div>
<button (click)="displayStatus()">Display Status </button>`
})
export class ChildComponent  {
  @Input() flight:any;
  @Output() alertStatus = new EventEmitter();
  displayStatus() {
    this.alertStatus.emit({status : this.flight.status })
  }
}
```

Output Cont.

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

@Component({
  selector: 'parent',
  template: `<child [flight]="flightDetail"
  (alertStatus)="diplayCurrentStatus($event)"></child>`,
  styleUrls: ['./parent.component.css']
})
export class ParentComponent {
public flightDetail:any = {
    no : 12345,
    name :'Air India',
    status : 'Departured'
  }
  diplayCurrentStatus(params){
    alert(params.status);
  }
}
```

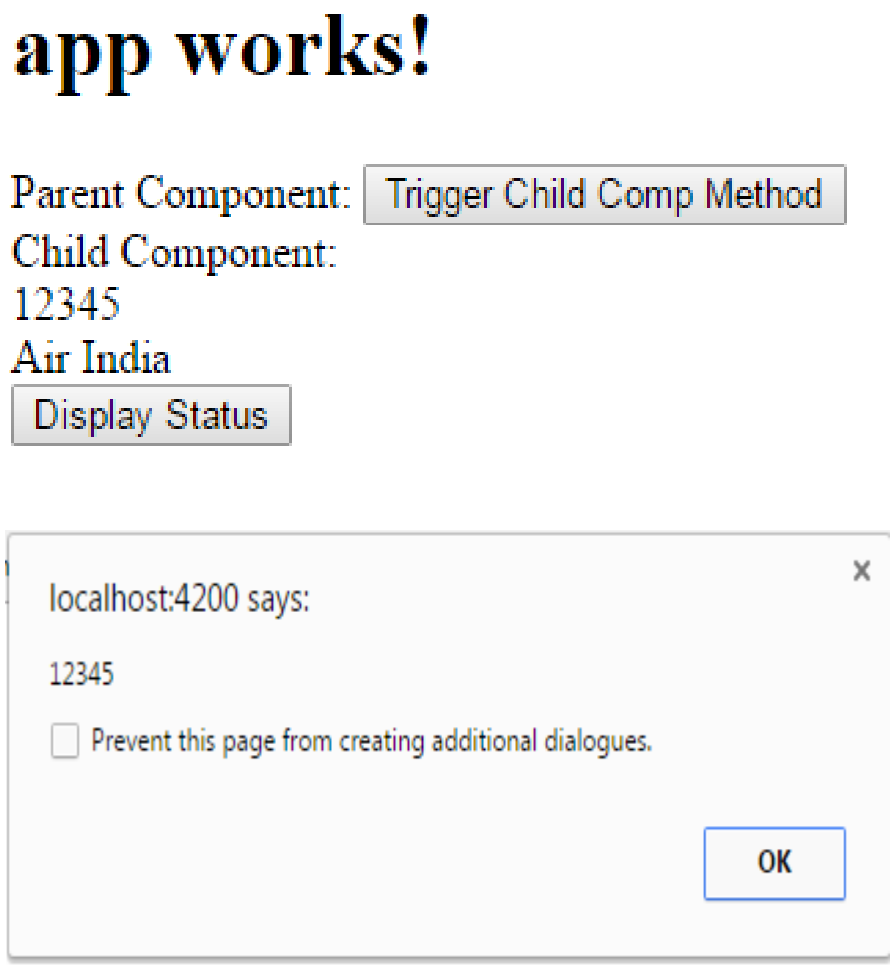


Template Variable

```
Parent Component:
<button (click)="childComp.displayNumber()">Trigger Child Comp Method
</button>

<child [flight]="flightDetail" #childComp
 (alertStatus)="diplayCurrentStatus($event)"></child>

 {{childComp.name}}
```

```
import { Component, Input, Output, EventEmitter } from '@angular/core';
@Component({
  selector: 'child',
  template: `<br> Child Component:<div>{{flight.no}}</div><div>{{flight.name}}
<button (click)="displayStatus()">Display Status </button>`
})
export class ChildComponent  {
  @Input() flight:any;
  @Output() alertStatus = new EventEmitter();
  displayStatus() {
    this.alertStatus.emit({status : this.flight.status })
  }
  displayNumber() {
    alert(this.flight.no);
  }
}
```

Template Variable Cont.

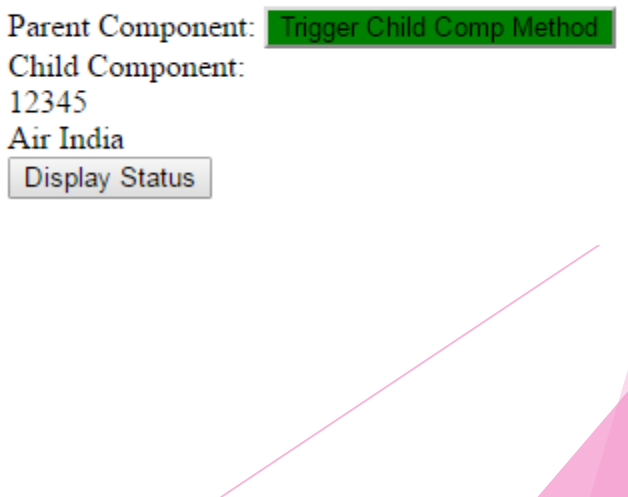


StyleUrls

```
Parent Component:
<button (click)="childComp.displayNumber()">Trigger Child Comp Method
</button>

<child [flight]="flightDetail" #childComp
 (alertStatus)="diplayCurrentStatus($event)"></child>

 {{childComp.name}}
```

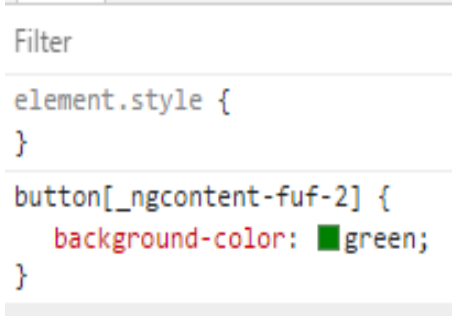
```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'parent',
  templateUrl: './parent.component.html',
  styleUrls: ['./parent.component.css']
})
export class ParentComponent {
public flightDetail:any = {
    no : 12345,
    name :'Air India',
    status : 'Departured'
  }
  diplayCurrentStatus(params){
    alert(params.status);
  }
}
```

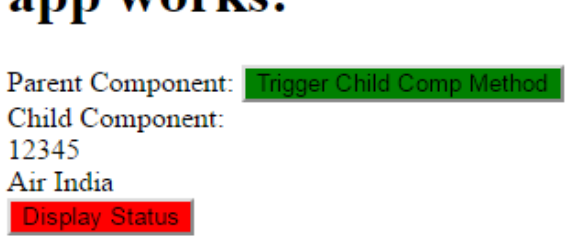
```
button {
    background-color: green;
}
```



Styles

```
import { Component, Input, Output, EventEmitter } from '@angular/core';
@Component({
  selector: 'child',
  template: `<br> Child Component:<div>{{flight.no}}</div><div>{{flight.name}}
<button (click)="displayStatus()">Display Status </button>`,
styles :['button { background-color: red }']

})
export class ChildComponent  {
  @Input() flight:any;
  @Output() alertStatus = new EventEmitter();
  displayStatus() {
    this.alertStatus.emit({status : this.flight.status })
  }
  displayNumber() {
    alert(this.flight.no);
  }
}
```





Interpolation

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
})
export class AppComponent {
  title = 'app works!';
  getTitle() {
    return "method";
  }

}
```

```
{{ 5+ 6}} <br>
Interpolation By Property:   {{title}}
<br>
InterPolation By Method : {{getTitle()}}
<br>
{{title + title}}
```

11
Interpolation By Property: app works!
InterPolation By Method : method

Restriction:

- Assignment Operator (=, =+, etc..)
- Multiple expression separated by ;
- New keyword
- Global namespace (Console.log())

```
{{ console.log("test")}} <br>
{{num = num + 5}}
<br>
{{title ; title}}
{{new Class()}}
```

Property Binding

```
<input type="checkbox" value="male" [checked]="trueValue">
<input type="checkbox" value="male" [checked]="getStatus()">
<input type="checkbox" value="male" [checked]="trueValue && !trueValue">
```

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
})
export class AppComponent {
    trueValue =true;
    getStatus() {
        return true;
    }
}
```

Restriction:

- Assignment Operator (=, =+, etc..)
- Multiple expression separated by ;
- New keyword
- Global namespace (Console.log())

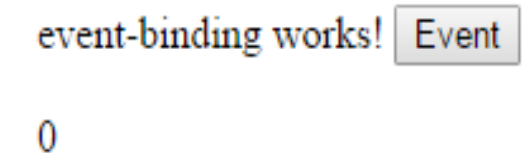
Event Binding : (event) =“statement”

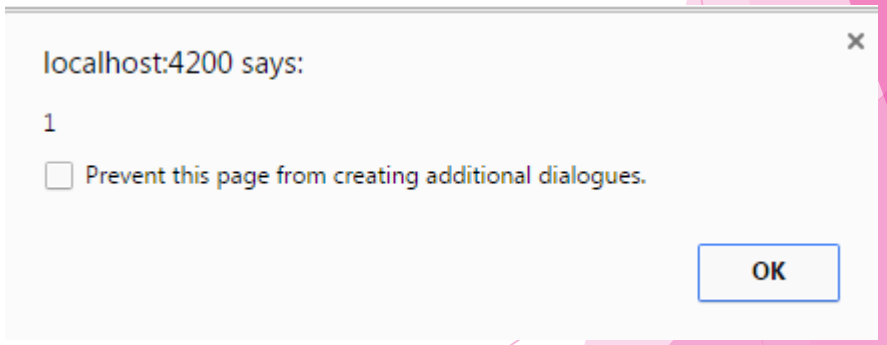
```
import { Component } from '@angular/core';

@Component({
  selector: 'app-event-binding',
  templateUrl: './event-binding.component.html',

})
export class EventBindingComponent {

num = 0;
increment() {
    this.num += 1;
}
  displayone() {
    alert(this.num);
  }

}
```

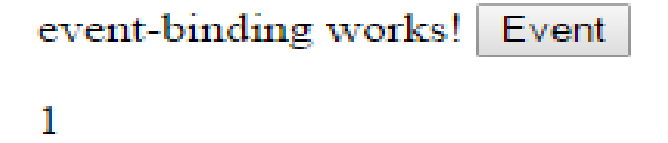
```
<p>
  event-binding works!
  <input type="button" (click)="increment();displayone()" value="Event">
</p>
{{num}}
```

Restriction:

- Assignment Operator (=, =+, etc..)
- Multiple expression separated by ;
- New keyword
- Global namespace (Console.log())







Null Value handling (Safe Navigating)

```
<p>
  safe-navigating works!
  {{flight?.name}}
  {{flight?.details?.departureTime}}
</p>
```

safe-navigating works! Air india 10 AM

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-safe-navigating',
  templateUrl: './safe-navigating.component.html',
})
export class SafeNavigatingComponent  {
flight:any = {
    name : "Air india",
    details : {
        departureTime : "10 AM",

    }
}

}
```

```
TypeError: Cannot read property 'departureTime' of undefined
    at CompiledTemplate.proxyViewClass.View_SafeNavigatingComponent0.detectChangesInter
    at CompiledTemplate.proxyViewClass.AppView.detectChanges (http://localhost:4200/ver
    at CompiledTemplate.proxyViewClass.DebugAppView.detectChanges (http://localhost:420
    at CompiledTemplate.proxyViewClass.AppView.internalDetectChanges (http://localhost:
    at CompiledTemplate.proxyViewClass.View_AppComponent0.detectChangesInternal (/AppMo
    at CompiledTemplate.proxyViewClass.AppView.detectChanges (http://localhost:4200/ver
    at CompiledTemplate.proxyViewClass.DebugAppView.detectChanges (http://localhost:420
    at CompiledTemplate.proxyViewClass.AppView.internalDetectChanges (http://localhost:
    at CompiledTemplate.proxyViewClass.View_AppComponent_Host0.detectChangesInternal (/
    at CompiledTemplate.proxyViewClass.AppView.detectChanges (http://localhost:4200/ver
    at CompiledTemplate.proxyViewClass.DebugAppView.detectChanges (http://localhost:420
    at ViewRef_.detectChanges (http://localhost:4200/vendor.bundle.js:54700:20)
    at http://localhost:4200/vendor.bundle.js:36615:67
    at Array.forEach (native)
    at ApplicationRef_.tick (http://localhost:4200/vendor.bundle.js:36615:25)
```

```
EXCEPTION: Error in ./SafeNavigatingComponent class SafeNavigatingComponent - inline template:0:3 caused by: Cannot read property 'departureTime' of undefined
ORIGINAL EXCEPTION: Cannot read property 'departureTime' of undefined
ORIGINAL STACKTRACE:
```

Directives *ngFor, *ngIf

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

@Component({
  selector: 'app-directives',
  templateUrl: './directives.component.html',
})
export class DirectivesComponent {
flightList = [{name : "Air India", status : "Arrived",
 classType:"Business"},
{name : "Jet Airways", status : "Departured"},
{name : "Indigo", classType:"Economic"}]

}
```

directives works!

Name: Air India Type: Business Status: Arrived
Name: Jet Airways Status: Departured
Name: Indigo Type: Economic

```
<p>
  directives works!
</p>
<div *ngFor="let flight of flightList">
    Name: {{flight.name}}
    <span [hidden]="!flight.classType">Type:{{flight.classType}}</span>
    <span *ngIf="flight.status"> Status: {{flight.status}}</span>
</div>
```

Directives ngSwitch

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

@Component({
  selector: 'app-switch',
  templateUrl: './switch.component.html'
})
export class SwitchComponent  {
flightName = "JetAirways";
}
```

switch works!

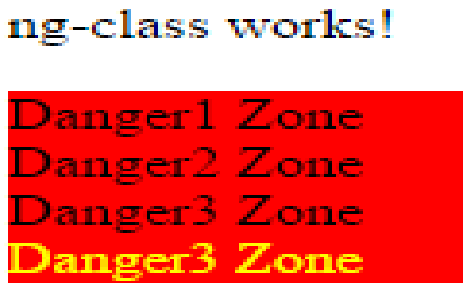
- Second largest airline in India

```
<p>
  switch works!
</p>
<ul [ngSwitch]="flightName">
  <li *ngSwitchCase="'AirIndia'">Third largest airline in India</li>
  <li *ngSwitchCase="'JetAirways'">Second largest airline in India</li>
  <li *ngSwitchDefault>Invalid</li>
</ul>
```

Directives ngClass

```
<p>
  ng-class works!
</p>
<div [class.danger]="isDanger">Danger1 Zone </div>
<div [ngClass]="{danger : isDanger == true }">Danger2 Zone </div>
<div [ngClass]="getClass()">Danger3 Zone </div>
<div [ngClass]="getClass1()">Danger3 Zone </div>
```

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

@Component({
  selector: 'app-ng-class',
  templateUrl: './ng-class.component.html',
  styles: ['.danger {background-color:red}',
  '.color { color :yellow}']
})
export class NgClassComponent  {
isDanger = true;
getClass() { return {danger : true}}
getClass1() { return 'danger color' }
}
```



Directives ngStyle

```
<p>
  ng-style works!
</p>
<div [style.color]="isDanger ? 'red' : 'green' ">Danger1 Zone </div>
<div [ngStyle]="{color : isDanger ? 'red' : 'color' }">Danger2 Zone </div>
<div [ngStyle]="getStyle()">Danger3 Zone </div>
```

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

@Component({
  selector: 'app-ng-style',
  templateUrl: './ng-style.component.html'
})
export class NgStyleComponent  {
isDanger = true;
getStyle() { return { color : 'red'}}
}
```

ng-style works!

Danger1 Zone
Danger2 Zone
Danger3 Zone

Service

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

@Injectable()
export class FlightService {
getFlight() {
    return {
        name : "Air India",
        status : "Arrived",
        no: 12345
    }
 }
}
```

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { HttpModule } from '@angular/http';
import {FlightService} from './flight.service';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    FormsModule,
    HttpModule
  ],
  providers: [FlightService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Service Cont.

```
import { Component } from '@angular/core';
import {FlightService} from './flight.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'app works!';
  flight:any;
  constructor(private flightService : FlightService){ }
  ngOnInit(){
    this.flight = this.flightService.getFlight();
  }
}
```

```
<h1>
  {{title}}
</h1>
Name:{{flight.name}}<br>
Number:{{flight.no}}<br>
Status :{{flight.status}}
```

app works!

Name:Air India
Number:12345
Status :Arrived

Type Safety & Forms

```
<form #loginForm="ngForm" (ngSubmit)="login(loginForm.value)" noValidate>
<label>User Name:</label><input (ngModel)="userName" name="userName">
<label>Password</label>
<input type="password" (ngModel)="password" name="password" required>
<button [disabled]="loginForm.invalid" type="submit">Login</button>
```

```
export class LoginService {
public isAuthenticated : boolean;
public user :IUser;
  constructor() { }
  authenticate(user:IUser) : boolean  {
    this.user = user;
    this.isAuthenticated = user.name  == 'ruthra' && user.password == '123' ;
    return this.isAuthenticated;  }
}
```

```
export class LoginComponent implements OnInit {
private user:IUser;
 constructor(private loginService:LoginService) { }
 ngOnInit() { }
 login(loginForm) { this.user = { id : 1,
    name : loginForm.userName,  password : loginForm.password};
    this.loginService.authenticate(this.user);
 }
```

```
export interface IUser {
    id:number,
    name:string,
    password:string
}
```

Validations

```
<form #loginForm="ngForm" (ngSubmit)="login(loginForm.value)" noValidate>
<label>User Name:</label>

<em *ngIf="loginForm.controls.userName?.invalid &&
 loginForm.controls.userName?.touched">Required</em>
<input (ngModel)="userName" name="userName" required>

<br><label>Password</label><em *ngIf="loginForm.controls.password?.invalid
&& (loginForm.controls.password?.touched || mouseOverLogin)">Required</em>

<input type="password" (ngModel)="password" name="password" required>

<div (mouseenter)="mouseOverLogin=true" (mouseleave)="mouseOverLogin=false">
<button [disabled]="loginForm.invalid" type="submit">Login</button></div>
```

loginForm.valid
loginForm.invalid
loginForm.valid
loginForm. dirty
loginForm.pristine
loginForm.submitted
loginForm.touched
loginForm.untouched

loginForm.controls.userName?.valid
loginForm.controls.userName?.invalid
loginForm.controls.userName?.dirty
loginForm.controls.userName?.pristine
loginForm.controls.userName?.touched
loginForm.controls.userName?.untouched

Reactive Forms & Validations

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

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

```
<form [formGroup]="profileForm" (ngSubmit)="saveProfile(profileForm.value)">
<label>First Name:</label>
<input formControlName="firstName"><br>
<label>Last Name</label>
<input formControlName="lastName1"><em *ngIf="!validateLastName()">
<span *ngIf="profileForm.controls.lastName1.errors.required">Required</span>
<span *ngIf="profileForm.controls.lastName1.errors.pattern">Value is invalid</span></em>
<button type="submit">Update</button>
</form>
```

Reactive Forms & Validations Cont.

```
export class ProfileComponent implements OnInit {

private profileForm : FormGroup;
private firstName : FormControl;
private lastName : FormControl;
  constructor(private loginService:LoginService) { }
  ngOnInit() {
    this.firstName = new FormControl(this.loginService.user.name, Validators.required);
    this.lastName = new FormControl(this.loginService.user.surName, [Validators.required,
      Validators.pattern('[a-zA-Z].*')]);
    this.profileForm = new FormGroup({firstName : this.firstName,lastName : this.lastName});
  }
  saveProfile(formValues){
    if(this.profileForm.valid){
          this.loginService.updateProfile({ id : 1,name : formValues.firstName,
           surName: formValues.lastName, password:"123"
        });
      }
    }
  validateLastName(){ return this.lastName.valid || this.lastName.untouched;  }
```

Custom Validator

```
 ngOnInit() {
    this.firstName = new FormControl(this.loginService.user.name, Validators.required);
    this.lastName = new FormControl(this.loginService.user.surName, [Validators.required,
      Validators.pattern('[a-zA-Z].*'), this.restrictedWords]);
    this.profileForm = new FormGroup({firstName : this.firstName,lastName : this.lastName});
 }
 private restrictedWords(control:FormControl) : {[key :string] : any} {
 return control.value && control.value.includes('foo') ? {'restrictedWords' : 'foo'} : null;
}
```

```
<label>Last Name</label>
<input formControlName="lastName">
<em *ngIf="lastName.errors?.restrictedWords">{{lastName.errors?.restrictedWords}}
</em>
```

Content Projection

```
<label>Title : </label> {{title}}
<hr>
<ng-content select="[details]"></ng-content>
<hr>
<ng-content select="[author]"></ng-content>
```

```
import { Component, Input } from '@angular/core';
@Component({
  selector: 'section',
  templateUrl: './section.component.html',
})
export class SectionComponent{
@Input() title:any;
}
```

```
<h1>
  {{title}}
</h1>
<section [title]="'sectionHeader'">
<div details>Body of the section </div>
<div author>Written By: Ruthra</div>
</section>
```

app works!

section works!

Title : sectionHeader

Body of the section

Written By: Ruthra

select="div"
select=".well"
select="#userName"

Pipes

```
<div *ngFor="let course of visibleCourses">
<label>Technology : </label>  {{course.technology | uppercase}} <br>
<label>Fees : </label> {{course.fees | currency : 'USD' :true}} <br>
<label>Start Date : </label>{{course.startDate | date : 'y-M-d'}} <br>
<label>Mode : </label> {{course.mode | timing}} <hr>
</div>
```

```
import {Pipe, PipeTransform} from '@angular/core';

@Pipe({name :'timing'})
export class timingPipe implements PipeTransform {
    transform(value:string){
        if(value == 'Online') { return value + " 7 AM to 9 AM on WeekDays"; }
        else if(value == 'ClassRoom'){ return value + " 9 AM to 5 PM on WeekEnd"}
        else {  return value;}
    } }
```

Technology : JAVA
Fees : $700.00
Start Date : 2017-5-3
Mode : Online 7 AM to 9 AM on WeekDays

Technology : ANGULAR
Fees : $500.00
Start Date : 2017-5-3
Mode : ClassRoom 9 AM to 5 PM on WeekEnd

Filter & Sorting

```
export class AppComponent {
 startDate = new Date();
 courses =[{technology : 'Java',fees :'700', mode:"Online",startDate:this.startDate },
 {technology :'Angular', fees:'500', mode:'ClassRoom', startDate:this.startDate}]
}
```

```
<button (click)="filterByValue=''">Clear Filter</button>
<button (click)="filterByValue='Online'">Online Only</button>
<button (click)="sortByValue='feesAsc'">Sort By Fees(Asc)</button>
<button (click)="sortByValue='feesDesc'">Sort By Fees(Desc)</button>
<training [courses]="courses" [filterBy]="filterByValue" [sortBy]="sortByValue"></training>
```

```
export class TrainingComponent implements OnChanges  {
@Input() courses:any;
@Input() filterBy:string;
@Input() sortBy:string;
visibleCourses = [];
ngOnChanges()
{   if(this.courses)
    {   if(this.filterBy === "Online"){this.visibleCourses = this.courses.filter(item =>
         {return item.mode === this.filterBy;}) }
        else {this.visibleCourses = this.courses;}
        if(this.sortBy === "feesAsc"){ this.visibleCourses.sort(sortByFeesAsc); }
        else if(this.sortBy === "feesDesc"){this.visibleCourses.sort(sortByFeesDesc);}
    }
    function sortByFeesAsc(c1, c2){ return c1.fees-c2.fees; }
    function sortByFeesDesc(c1, c2){ return c2.fees-c1.fees; }
}}
```

Filter & Sorting Cont.

```
<div *ngFor="let course of visibleCourses">
<label>Technology : </label>  {{course.technology | uppercase}} <br>
<label>Fees : </label> {{course.fees | currency : 'USD' :true}} <br>
<label>Start Date : </label>{{course.startDate | date : 'y-M-d'}} <br>
<label>Mode : </label> {{course.mode | timing}} <hr>
</div>
```

Clear Filter | Online Only | Sort By Fees(Asc) | Sort By Fees(Desc)

Technology : JAVA
Fees : $700.00
Start Date : 2017-5-3
Mode : Online 7 AM to 9 AM on WeekDays

Technology : ANGULAR
Fees : $500.00
Start Date : 2017-5-3
Mode : ClassRoom 9 AM to 5 PM on WeekEnd

Opaque Token

Jquery Service

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

export let JQ_TOKEN = new OpaqueToken('jQuery');
```

App Module

```
import { JQ_TOKEN } from './jquery.service';

declare let jQuery : Object;
```

```
providers: [{provide : JQ_TOKEN,useValue : jQuery}],
```

Component

```
export class AppComponent {
    constructor(@Inject(JQ_TOKEN) private $:any){}
}
```

Modal Component

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

@Component({
  selector: 'sample-modal',
  templateUrl: './sample-modal.component.html',
  styleUrls: ['./sample-modal.component.css']
})
export class SampleModalComponent { }
```

```
<div id="modalId" class="modal">
  <div class="modal-content">
    <span  class="close">&times;</span>
    <p>Some text in the Modal..</p>
  </div>
</div>
```

App Component

```
<sample-modal ></sample-modal>
<button (click)="openModal()"> Open Modal</button>
```

```
export class AppComponent {
 constructor(@Inject(JQ_TOKEN) private $:any){}
  title = 'app works!';
  openModal(){
    this.$("#myModal").modal({});
  } }
```

Modal Trigger Directive

```
import { Directive, Inject, ElementRef, OnInit, Input } from '@angular/core';
import {JQ_TOKEN} from './jquery.service';
@Directive({
  selector: '[modal-trigger]'
})
export class ModalTriggerDirective implements OnInit {
private el:HTMLElement;
@Input('modal-trigger') modalId : string;
  constructor(ref:ElementRef, @Inject(JQ_TOKEN) private $:any) {
    this.el = ref.nativeElement;}
ngOnInit() {
    this.el.addEventListener('click', e => {
        this.$(`#${this.modalId}`).modal({});});
    }
}
```

APP Component

```
<sample-modal modalId="resultModal" [title]="'search Result'"></sample-modal>

<button modal-trigger="resultModal"> Open Modal</button>
```

Modal Trigger Directive Cont.

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

@Component({
  selector: 'sample-modal',
  templateUrl: './sample-modal.component.html',
  styleUrls: ['./sample-modal.component.css']
})
export class SampleModalComponent  {
 @Input() title:string;
 @Input() modalId:string; }
```

```
<div id="{{modalId}}" class="modal">
 <div class="modal-content">
    <span class="close">&times;</span>
    <p>Some text in the Modal..</p>
    {{title}}
 </div>
</div>
```

ViewChild (ContentChild / Children)

```
<div id="{{modalId}}" #modalContainer class="modal">
 <div class="modal-content">
    <span (click)="closeModal()" class="close">&times;</span>
    <p>Some text in the Modal..</p>    {{title}}
 </div></div>
```

```
import { Component, OnInit,Input, Inject, ViewChild, ElementRef } from '@angular/core';
import {JQ_TOKEN} from '../jquery.service';
@Component({
  selector: 'sample-modal',
  templateUrl: './sample-modal.component.html',
  styleUrls: ['./sample-modal.component.css']
})
export class SampleModalComponent {
 @Input() title:string;
 @Input() modalId:string;
 @ViewChild('modalContainer') el : ElementRef
 constructor(@Inject(JQ_TOKEN) private $:any) { }
 closeModal(){
    this.$(this.el.nativeElement).modal('hide');
  }
}
```

HTTP - Service

```
import { Injectable } from '@angular/core';
import {Http, RequestOptions, Response, Headers} from '@angular/http';
import {Observable} from 'rxjs/Rx';

@Injectable()
export class CustomerService {
  constructor(private http:Http) { }
  getCustomers(){
    return this.http.get("https://www.w3schools.com/angular/customers1.php")
    .map((res:Response) => {return res.json()}).catch(this.handleError);
  }
  postCustomer(customerList)
  {
    let url = "https://www.w3schools.com/angular/customers.php";
    let header = new Headers({'content-Type':'application/text'});
    let options = new RequestOptions({headers : header});
    return this.http.post(url,JSON.stringify(customerList));
  }
  handleError(error:Response)
  {
    console.log("HTTP Error : " + JSON.stringify(error));
    return Observable.throw(false);
  }     }
```

import { HttpModule } from '@angular/http';
imports: [HttpModule]

HTTP - Component

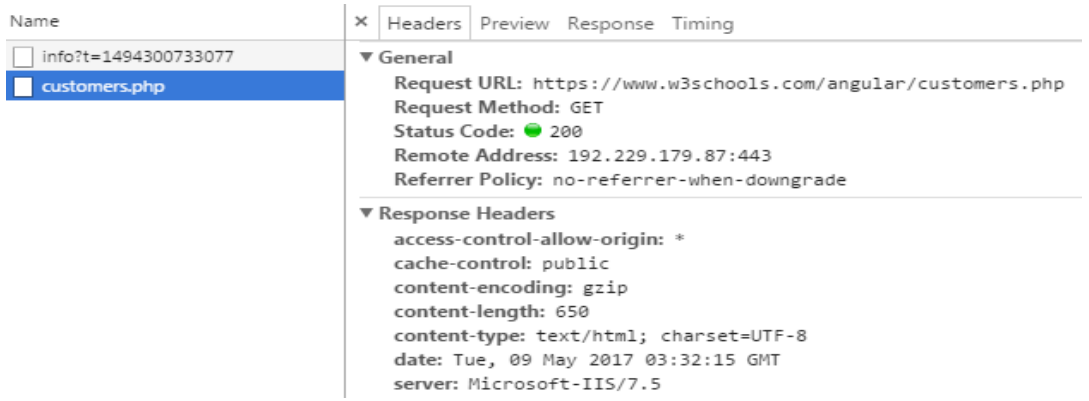
```
import { Component } from '@angular/core';
import {CustomerService} from './customer.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
})
export class AppComponent {
constructor(private customerService:CustomerService){}
title = 'app works!';
customerList:any;
getCall(){
    this.customerService.getCustomers().subscribe((customers) => {
        this.customerList = customers } );
}
postCall(){
    this.customerService.postCustomer(this.customerList).subscribe();}
}
```

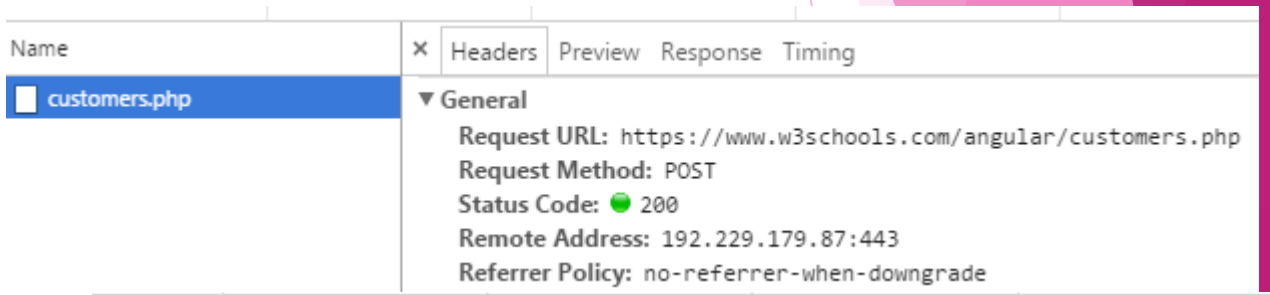
```
<button (click)="getCall()">Get Call</button>
<button (click)="postCall()">Post Call</button>
<div *ngFor="let customer of  customerList?.records">
    <label>Name : </label> <b>{{customer.Name}} </b>
    <label>City : </label> <b>{{customer.City}}</b>
    <label>Country : </label> <b> {{customer.Country}}</b><br>
</div>
```

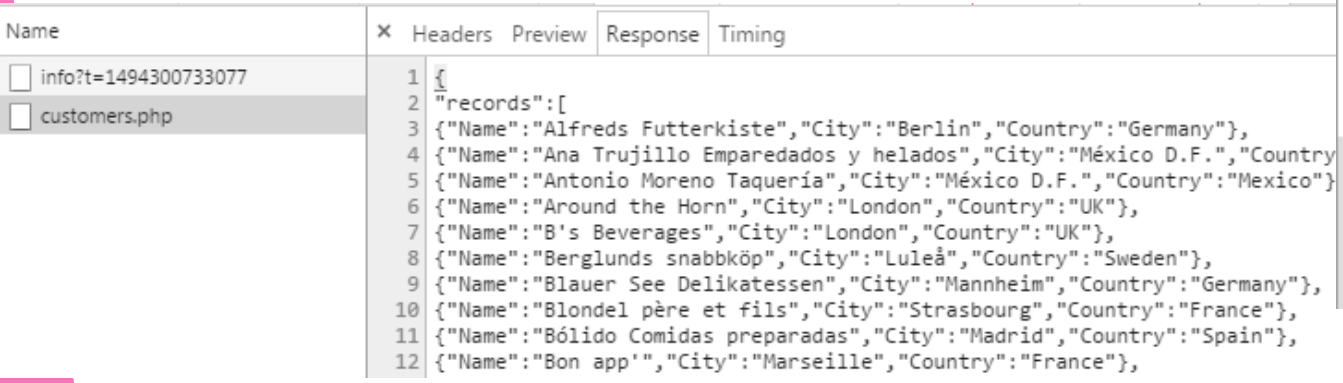
HTTP - Output

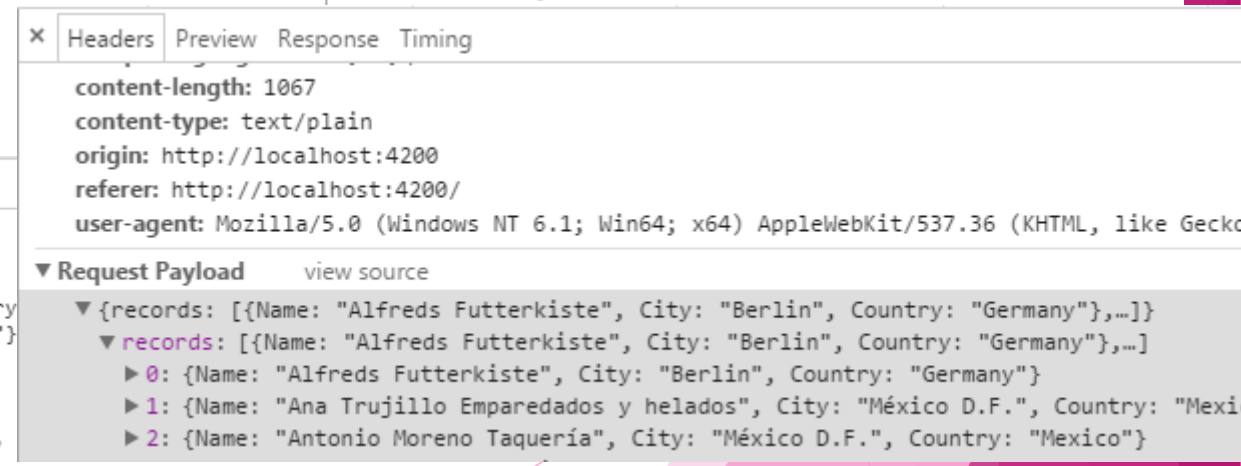
Get Call | Post Call

Name : **Alfreds Futterkiste** City : **Berlin** Country : **Germany**
Name : **Ana Trujillo Emparedados y helados** City : **México D.F.** Country : **Mexico**
Name : **Antonio Moreno Taquería** City : **México D.F.** Country : **Mexico**
Name : **Around the Horn** City : **London** Country : **UK**









Isolated Unit Test - Service

```
import { CustomerService } from './customer.service';
import {Observable} from 'rxjs/Rx';

describe('CustomerService', () => {
 let customerService : CustomerService, mockHttp;
  beforeEach(() => {
    mockHttp = jasmine.createSpyObj('mockHttp', ['get','post']);
    customerService = new CustomerService(mockHttp);
  });

  it('getCustomers Test', ()=>{
    mockHttp.get.and.returnValue(Observable.of({}));
    customerService.getCustomers();
    expect(mockHttp.get).toHaveBeenCalledWith('https://www.w3schools.com/angular/customers.php'
    });
  it('postCustomers Test', ()=>{
    mockHttp.post.and.returnValue(Observable.of(false));
    customerService.postCustomer({});
    expect(mockHttp.post).toHaveBeenCalledWith('https://www.w3schools.com/angular/customers.php
     "{}");
    });
});
```

Isolated Unit Test - Component

```
import { AppComponent } from './app.component';
import {Observable} from 'rxjs/Rx';

describe('AppComponent', () => {
 let appComponent: AppComponent, customerService : any;
 let customerList = {
      'records' :[ {'Name' :'Ruthra', 'Country' :'India', 'City' : 'Chennai'},
      {'Name' :'Gowthami', 'Country' :'India', 'City' : 'Bangalore'},
      {'Name' :'Kumar', 'Country' :'India', 'City' : 'Delhi'}   ]
    };
  beforeEach(() => {
    customerService = {
      getCustomers() {
        return Observable.of(customerList)
      }
    };
    appComponent = new AppComponent(customerService);

  });

  it('Customer List', ()=>{
    appComponent.getCall();
    expect(appComponent.customerList).toEqual(customerList);
    });
});
```

Integrated Unit Test

```
import { TestBed, async } from '@angular/core/testing';
import { AppComponent } from './app.component';

describe('AppComponent', () => {
  beforeEach(() => {
    TestBed.configureTestingModule({
      declarations: [AppComponent]  });
    TestBed.compileComponents();      });

  it('should create the app', async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.debugElement.componentInstance;
    expect(app).toBeTruthy();     }));

  it(`should have as title 'app works!'`, async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.debugElement.componentInstance;
    expect(app.title).toEqual('app works!');    }));

  it('should render title in a h1 tag', async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.debugElement.nativeElement;
    expect(compiled.querySelector('h1').textContent).toContain('app works!');  }));
});
```

End to End Test

```
import { browser, element, by } from 'protractor';

export class ProjectPage {
  navigateTo() {    return browser.get('/');  }
  getParagraphText() {   return element(by.css('app-root h1')).getText();  }
}
```

```
import { ProjectPage } from './app.po';

describe('project App', function() {
  let page: ProjectPage;
  beforeEach(() => {
    page = new ProjectPage();
  });
  it('should display message saying app works', () => {
    page.navigateTo();
    expect(page.getParagraphText()).toEqual('app works!');
  });
});
```

Production


```
import {Observable, Subject} from 'rxjs/Rx';
```

```
import {Observable} from 'rxjs/Observable';
import {Subject} from 'rxjs/Subject';
```

rxjx-extension.ts

```
import 'rxjs/add/oservable/of
import 'rxjs/add/oservable/throw

import 'rxjs/add/oservable/catch
import 'rxjs/add/oservable/do
import 'rxjs/add/oservable/map
import 'rxjs/add/oservable/filter
```

app.module.ts

```
import './rxjs-extension'
```

Thank You

91-9715531887 (RUTHRAN.KUMAR@GMAIL.COM)