



Curso de Angular 14

Enrique de la Calle Santa Ana

22 Febrero 2,024

Índice

1. Introducción a Angular
2. Angular Cli
3. Componentes
4. Directivas
5. Formularios Reactivos
6. Servicios
7. Peticiones HTTP

Índice

- 8. Routing
- 9. Pipes
- 10. Estado de la Aplicación
- 11. Expecificidad CSS , BEM , Animaciones y preprocesador SASS
- 12. Testing
- 13. Builds y Despliegue
- 14. Proyecto Final

01

Introducción a Angular

Introduccion Angular

- **Introducción:**
- <https://docs.angular.lat/>
- Ventajas y desventajas
- Novedades sobre A14
- Creación de entorno trabajo
- <https://angular.io/guide/setup-local>
- Creamos nuestra primera App
- Analizamos los ficheros

Revisión de puntos

- ☐ ¿Por qué Angular?
- ☐ Ventajas y desventajas principales
- ☐ Características de Angular 14.2
- ☐ Diferencias más destacables frente a las versiones actuales de Angular
- ☐ ¿Qué es el renderizador de Angular?
- ☐ Instalación de Angular CLI a través de NPM
- ☐ Diferenciando entre instalaciones locales y globales de NPM
- ☐ ¿Qué es NPX y por qué usarlo?
- ☐ Creación de tu primer proyecto Angular ☐

Revisión de puntos

- ☐ Análisis de la estructura del proyecto y sus archivos de configuración
- ☐ Desplegando nuestra aplicación localmente con Angular CLI

02

Comandos Ng-cli

Revisión de puntos

- ☐ Introducción Comandos CLI esenciales
- ☐ ng new
- ☐ ng serve
- ☐ ng generate
- ☐ ng add
- ☐ ng build
- ☐ ng update
- ☐ Otros comandos existentes

03

Componentes / Material

Links Interés

- <https://codingpotions.com/angular-componentes>
- <https://ngchallenges.gitbook.io/project/componentes>
- <https://codingpotions.com/angular-comunicacion-componentes/>
- <https://platzi.com/clases/2486-angular-componentes/41180-ciclo-de-vida-de-componentes/#:~:text=Ciclo%20de%20vida%20en%20Angular,los%20inputs%20en%20todo%20momento>
- <https://material.angular.io/guide/getting-started>

Componentes

- Componente es un decorador Typescript (personaliza visual)
- Componente como unidad visual mínima en Angular
- Respeta MVC (template = vista = html, css) , (Modelo = prop clase) , (Controlador = Typescript o clase)
- Metadatos en un componente (selector , templateUrl , styleUrls)
- Recomendación:
- las Clases CamelCase y terminadas en Component (Si es componente)
- El selector en Minuscula y separado por guiones (html no dife may)

Componentes

- Creación Componente
- Se puede generar a mano creando todos los ficheros o
- Ng generate component components/contador
- Revisamos los ficheros generados
- Enlace a datos
- {{cliente.nombre}} interpolación solo en un sentido
- [property] = “value” Propiedad o atributo
- (event) = “handler” Evento o controlador
- [(ng-model)] = “property” propiedad (two binding)

Practica Componentes

- Creamos un proyecto nuevo Componentes / No Router / CSS
- C:\Ang14\npx ng new 03_Componentes
- C:\Ang14\npx generate component components/contador
- Revisamos la estructura y vemos como app.module.ts se ajusta
- Vemos la jerarquía de estilos css en cada parte de la app
- Incorporo en la pagina app.component.html el contador (varios fijos)
- Le paso el atributo mediante [] en varios contadores
- `@Input() titulo:string = 0;`

Practica Componentes

- En app.component.ts incorporamos un array de contadores

- `interface ContadorInterface {`
- `id:number,`
- `title:string,`
- `inicio:number,`
- `valor:number`
- `}`
- `aContadores:ContadorInterface[] = [`
- `{id:1, title:"Contador 1", inicio: 0 },`
- `{id:2, title:"Contador 2", inicio: 5 },`
- `{id:3, title:"Contador 3", inicio: 4 },`
- `]`

Practica Componentes

- Quiero ver el array de contadores en la app (introduzco el concepto pipe | json)
- `<p>Prueba {{ aContadores | json }}</p>`
- Adapto la estructura con un bucle json
- `<table border="1">`
- `<thead>`
- `<tr>`
- `<th *ngFor="let contador of aContadores" >{{contador.title}}</th>`
- `</tr>`
- `</thead>`
- `<tbody>`
- `<tr>`
- `<td *ngFor="let contador of aContadores" ><app-contador></app-contador></td>`
- `</tr>`
- `</tbody>`
- `</table>`

Practica Componentes

- He introducido el concepto directiva de estructura (*ngFor="let contador of aContadores ")
- Directiva de Atributo (modifican las características de un componente como title ,etc [])
- Directivas de estructura (modifican el dom *ngFor ,etc)
- Directivas de plantilla que son los componentes en si
- Le pasamos como directivas de atributo , ID , title , contador y presentamos en el contador su valor
- Incorporamos un timer: (podríamos hacer un timerinterval de JS pero aprovechamos y ponemos)
- `import { interval, Observable, Subscription } from 'rxjs';`
- `const unseg:Observable<number> = interval(1000);`
- Adelantamos la comunicación Bidireccional con ngModel
- `app.module: import { FormsModule } from '@angular/forms'`
- `App.component: <input [ngModel]="currentItem.name" (ngModelChange)="currentItem.name=$event" id="example-change">`

Codigo contador.component.ts

```
○ import { Component, Input, Output, OnChanges, OnInit, OnDestroy, SimpleChanges, EventEmitter } from '@angular/core';
○ import { interval, Observable, Subscription } from 'rxjs';
○ const unseg:Observable<number> = interval(1000);
○ @Component({ selector: 'app-contador', templateUrl: './contador.component.html', styleUrls: ['./contador.component.css']})
○ export class ContadorComponent implements OnInit, OnDestroy, OnChanges {
○   @Input() id:number = 0; @Input() valor:number = 0; @Input() parentMessage:string = "";
○   @Output() messageEvent = new EventEmitter<string>();
○   obs?: Subscription;
○   constructor() {
○     this.obs = unseg.subscribe(x=>{
○       console.log("temporizador un seg", x)
○       this.valor++;
○       this.messageEvent.emit(`Papa: soy ${this.id} con el valor ${this.valor}`);
○     })
○   }
○ }
```

Codigo contador.component.ts

```
○   ngOnInit(): void {  
○       console.log("init elemento")  
○   }  
○   ngOnDestroy() {  
○       console.log("destruyo elemento")  
○       this.obs?.unsubscribe();  
○   }  
○   ngOnChanges(changes: SimpleChanges) {  
○       console.log("cammbios ", changes)  
○   }  
○   ngDoCheck(){  
○       //console.log("do check")  
○   }  
○   }
```

Ejemplo Compente

- Explicamos el método de matar un hijo

Ciclo de Vida del componente

<https://platzi.com/clases/2486-angular-componentes/41180-ciclo-de-vida-de-componentes/#:~:text=Ciclo%20de%20vida%20en%20Angular,los%20inputs%20en%20todo%20momento>

- constructor
- ngOnChanges
- ngOnInit
- ngDoCheck
- ngAfterContentInit
- ngAfterContentChecked
- ngAfterViewInit
-
- ngOnDestroy

Revisión de puntos

- **3. COMPONENTES**

- ☐ Metadatos de componentes
- ☐ Creación de un componente
- ☐ Instanciando componentes en archivos HTML
- ☐ Introducción al ngModel
- ☐ Data binding
- ☐ Operador de coalescencia nula
- ☐ Anidado de componentes
- ☐ Pasando datos al componente a través de @Inputs
- ☐ Respondiendo a eventos con @Outputs
- ☐ Ciclo de Vida de los componentes
- ☐ ¿Cuándo usar el ciclo de vida de los componentes en aplicaciones reales?
- ☐ Aplicando estilos a los componentes desde una hoja CSS o SCSS
- ☐ ¿Qué son los módulos?
- ☐ La organización de un proyecto mediante módulos
- ☐ Creación de módulos en un proyecto Angular
- ☐ Inyección de dependencias en Angular
- ☐ Creación de una plantilla HTML inicial

03 (Bis)

Material

Angular Material

Librería Grafica (Google) para incorporar a nuestros Proyectos

Alternativas: BootStrap , PrimeNg , etc

- Mkdir c:\Ang14\demo_material
- Cd , copio proto0 y npm i
- Npx ng add @angular/material (Trampa Aviso)
- Revisamos app.module

```
• import { MatSlideToggleModule } from '@angular/material/slide-toggle';  
•  
•     @NgModule ({  
•         imports: [  
•             MatSlideToggleModule,  
•         ]  
•     })  
•  
•     class AppModule {}
```


Angular Material (Cont. Y Schematics)

- `App.component.html`
- `<mat-slide-toggle>Toggle me!</mat-slide-toggle>`
- Angular Material tiene distintos templates y macros
- Un ejemplo seria un marco de navegación
- `ng generate @angular/material:navigation navega`
- (Si error versión)
- `"npx ng update @angular/material@14"`

Revisión de puntos

- **3. COMPONENTES**

- ☐ Introducción a Angular Material como framework de componentes
- ☐ Breve introducción a los componentes más destacables de Angular Material
- ☐ Instalación y configuración de Angular Material en un proyecto Angular
- ☐ ¿Qué son los schematics?
- ☐ Comentarios acerca de los schematics de Angular Material
- ☐ Empleando componentes de Angular Material en un proyecto Angular

04

Directivas/Modulos