









Angular 17

Tema 8. Aplicaciones tradicionales.
Módulos

Objetivos

-  Conocer el funcionamiento de los módulos de Angular
-  Entender la sintaxis tradicional de plantillas de Angular

Contenidos

-  Sintaxis de los módulos de Angular
-  Introducción a las directivas estructurales
-  Ejemplo con rutas
-  Ejemplo con HttpClient

Módulos en Angular. @NgModule

Elementos que permiten organizar el código en unidades lógicas. Gestionan la importación/exportación de cualquier elemento de Angular

Un módulo de Angular se decora con **@NgModule**:

- declarations** Elementos que pertenecen al módulo.
- exports** Componentes, directivas y pipes del módulo que permitimos que se usen en otros módulos.
- imports** Módulos de los que podremos usar sus elementos exportados.
- providers** Servicios que estarán disponibles para ser inyectados en los elementos que pertenecen al módulo.
- bootstrap** Si es el módulo de arranque, componentes disponibles para ser dibujados en el arranque.

```
@NgModule({  
  declarations: [  
    AppComponent  
  ],  
  imports: [  
    BrowserModule,  
    AnimalesModule,  
    PlantasModule  
  ],  
  providers: [],  
  bootstrap: [AppComponent]  
})
```

Directivas estructurales (introducción)

Permiten añadir o eliminar elementos al DOM. Se recomienda reemplazarlas por la nueva sintaxis de @bloques de Angular 17.

Se basa en **<ng-template>** y sus directivas de atributo **[ngIf]**, **[ngForOf]**, etc. Ya no suele utilizarse directamente, en su lugar se emplean las abreviaturas sintácticas ***ngIf**, ***ngFor**, ***ngSwitchCase**:

```
<p *ngFor="let p of productos">
  {{p.id}} {{p.nombre}}
</p>
```

```
<p *ngIf="producto.precio>10">
  El coste es superior al esperado
</p>

<p *ngIf="producto.precio>10;else menor">
  El coste es superior al esperado
</p>

<ng-template #menor>
  <p>El coste es menor</p>
</ng-template>
```

```
<div [ngSwitch]="producto.estado">
  <p *ngSwitchCase="'A'">Nuevo</p>
  <p *ngSwitchCase="'B'">Reacondicionado</p>
  <p *ngSwitchCase="'C'">Segunda mano</p>
</div>
```

En ocasiones también se emplea el elemento **<ng-container>**:

```
<ng-container [ngSwitch]="producto.estado">
  <p *ngSwitchCase="'A'">Nuevo</p>
  <p *ngSwitchCase="'B'">Reacondicionado</p>
  <p *ngSwitchCase="'C'">Segunda mano</p>
</ng-container>
```

¿Qué hemos aprendido?



- Uso de módulos en Angular
- Sintaxis de las directivas estructurales
- Importación de rutas y HttpClient

Resumen de comandos

npm

npm install [-g] paquete
npm install
npm uninstall paquete
npm -version

ng (1)

ng version
ng help
ng serve [-o]
ng build
ng new nombre_proyecto
 --routing false
 --skip-tests
 --skip-git
 --no-standalone

ng (2)

ng generate component nombre_componente / ng g c
ng generate service nombre_servicio / ng g s
ng generate module nombre_módulo / ng g m

Ejercicio 08

Adapta el ejercicio 07 B del tema anterior para su uso con módulos. Si no lo has completado, lo tendrás en la sección de ficheros del tema siete bajo el nombre «ejercicio 07 B.zip»

Crea un módulo «nasa» que contenga los componentes y servicios, y por supuesto utiliza rutas y HttpClient como módulos.

```

v nasa
  > detalle
  > lista
  v models
    TS foto.ts
    TS resumen.ts
  > ver
  TS datos.service.ts
  TS nasa.module.ts
  
```

```

v rutas
  > no-encontrado
  TS rutas.module.ts
  # app.component.css
  <> app.component.html
  TS app.component.ts
  TS app.module.ts
  
```

Utiliza «*ngIf» y «*ngFor» en las plantillas.

Usa constructores para la inyección de dependencia y no emplees señales.