DESARROLLO DE SOFTWARE FRONTEND II

ESPINOSA MARTINEZ CARLOS LEONEL

Aarón Hernández García

2.1 Investigación: Elementos Básicos de Angular

30/01/2025

Introducción a Angular

Angular es un framework de desarrollo web basado en TypeScript desarrollado por Google. Se utiliza para construir aplicaciones web de una sola página (SPA, Single Page Applications) que ofrecen una experiencia de usuario rápida y fluida. Su arquitectura modular permite organizar el código de manera eficiente y escalable, facilitando el mantenimiento y la reutilización de componentes.

1. ¿Qué es Angular y para qué se utiliza?

Angular es un framework que permite la creación de aplicaciones web interactivas con una estructura modular basada en componentes. Se utiliza para desarrollar aplicaciones de gran escala con facilidad de mantenimiento y optimización del rendimiento.

Principales características:

- **Basado en componentes**: La aplicación se divide en pequeños bloques reutilizables.
- **Inyección de dependencias**: Facilita la gestión de dependencias en toda la aplicación.
- **Directivas**: Permite modificar el comportamiento y la estructura del DOM de manera declarativa.
- **Enlace de datos (Data Binding)**: Sincronización automática entre el modelo y la vista.
- **Enrutamiento**: Facilita la navegación entre diferentes vistas de la aplicación.
- **Modularidad**: Organiza la aplicación en módulos para mejorar la escalabilidad.

2. Componentes en Angular

¿Qué son los componentes?

Los componentes son bloques fundamentales de una aplicación Angular. Cada componente define una parte de la interfaz de usuario y su comportamiento asociado.

Creación y estructura de un componente

Un componente se define mediante una clase TypeScript decorada con @Component. Contiene:

- **Selector**: Define el nombre del elemento HTML asociado.
- **Template (HTML)**: La estructura de la interfaz de usuario.
- **Estilos (CSS/SCSS)**: Para la apariencia visual.
- **Lógica (TypeScript)**: Define el comportamiento del componente.

```typescript
import {Component} from '@angular/core';
import {FormsModule} from '@angular/forms';
import {bootstrapApplication} from '@angular/platform-browser';

@Component({
  selector: 'app-root',
  template: `
    <hr />
    <h1>Hola me llamo Aaron {{ name }}!</h1>
  `,
  imports: [FormsModule],
})
export class DemoComponent {
  name = '';
}
```

3. Módulos en Angular

¿Qué son los módulos?

Un módulo en Angular agrupa componentes, directivas, servicios y otros elementos relacionados.

Propósito del AppModule

El AppModule es el módulo principal donde se registran los componentes y dependencias de la aplicación.

Ejemplo de un AppModule:

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component';

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

Diferencias entre NgModule y Lazy-Loaded Modules

- **NgModule:** Define y carga los componentes de manera estática en el AppModule.
- **Lazy-Loaded Modules:** Se cargan bajo demanda para mejorar el rendimiento.

4. Servicios en Angular

¿Qué son los servicios?

Son clases que contienen lógica reutilizable y permiten compartir datos entre componentes.

Creación de un servicio

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

@Injectable({ providedIn: 'root' })
export class EjemploService {
  obtenerMensaje() {
    return 'Este es un servicio en Angular';
  }
}
```

Rol de @Injectable e inyección de dependencias

El decorador @Injectable indica que la clase puede ser inyectada como dependencia en otros componentes o servicios.

5. Directivas en Angular

¿Qué son las directivas?

Son instrucciones que modifican la estructura o apariencia de los elementos del DOM.

Diferencias entre directivas estructurales y de atributo

- **Estructurales:** Modifican la estructura del DOM (*ngIf, *ngFor).
- **De atributo:** Alteran la apariencia o comportamiento de un elemento ([ngStyle], [ngClass]).

Ejemplo de directivas:

```
<p *ngIf="mostrarTexto">Este texto se muestra condicionalmente.<
<ul>
  <li *ngFor="let item of lista">{{ item }}</li>
</ul>
<p [ngStyle]="{color: 'red'}">Texto en rojo</p>
<p [ngClass]="'clase-destacada'">Texto con clase CSS</p>
```

Conclusión

Angular es un framework poderoso y versátil que permite desarrollar aplicaciones web estructuradas y escalables. Gracias a su arquitectura basada en componentes, módulos, servicios y directivas, los desarrolladores pueden construir aplicaciones reutilizables y de fácil mantenimiento.

La combinación de estos elementos permite separar la lógica de negocio de la interfaz de usuario, lo que facilita la colaboración entre equipos de desarrollo y mejora la eficiencia en el desarrollo de software.

Además, la inyección de dependencias y el uso de módulos cargados de forma diferida optimizan el rendimiento de la aplicación, asegurando que solo se carguen los recursos necesarios en el momento adecuado.

En conclusión, comprender estos fundamentos de Angular no solo ayuda a desarrollar aplicaciones más robustas y eficientes, sino que también permite aprovechar al máximo las herramientas y capacidades que ofrece el framework, mejorando la experiencia del usuario final y la productividad del equipo de desarrollo.

Referencias

1. Angular - Documentación Oficial: https://angular.io/docs
2. TypeScript - Documentación Oficial: https://www.typescriptlang.org/docs