

Angular typescript example

Angular is built with TypeScript, so all Angular examples inherently use TypeScript. A fundamental example involves defining a component with a TypeScript class, an HTML template, and a CSS selector.

Basic Angular Component Example

This example demonstrates a simple `HelloComponent` that uses a string property in its template via interpolation.

`hello.component.ts` (TypeScript class with metadata):

typescript

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

@Component({
  selector: 'app-hello', // The tag used in HTML, e.g., <app-hello></app-hello>
  template: `<h1>{{ title }}</h1>
             <p>Welcome to the Angular application!</p>`, // Inline HTML template
  standalone: true // Marks the component as standalone (modern approach)
})
export class HelloComponent {
  title = 'Hello world!'; // A TypeScript property available to the template
}
```

`main.ts` (Bootstrapping the application):

typescript

```
import { bootstrapApplication } from '@angular/platform-browser';
import { HelloComponent } from './hello.component';

bootstrapApplication(HelloComponent)
  .catch(err => console.error(err));
```

TypeScript Fundamentals in Angular

Angular makes extensive use of core TypeScript features.

Interfaces

Interfaces are used for static typing, ensuring objects have the correct structure.

typescript

```
interface Product {
  id: number;
  name: string;
  price: number;
}

const productList: Product[] = [
  { id: 1, name: 'Laptop', price: 999 },
  { id: 2, name: 'Phone', price: 499 }
];
```

Classes and Decorators

Components and services are defined as TypeScript classes. **Decorators** (like `@Component` or `@Injectable`) attach metadata to these classes, telling Angular how to process them.

typescript

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

@Injectable({
  providedIn: 'root'
})
export class ProductService {
  getProducts(): Product[] {
    // Logic to fetch data (e.g., from an API using HttpClient)
    return productList;
  }
}
```

Data Binding

Angular uses TypeScript properties and methods with data binding syntax in the HTML templates.

typescript

```
// In a component class (.ts file)
username: string = 'JaneDoe';

deleteItem(index: number): void {
  // Logic to handle an event
  console.log(`Deleting item at index ${index}`);
}
```

html

```
<!-- In the corresponding template (.html file) -->
<input [(ngModel)]="username"> <!-- Two-way binding -->
<button (click)="deleteItem(i)">Delete</button> <!-- Event binding -->
<p>{{ username }}</p> <!-- Interpolation -->
```

For more in-depth examples and tutorials, the official Angular documentation and the **TypeScript handbook** are excellent resources

Revision #2

Created 29 October 2025 02:43:40 by AI API

Updated 11 December 2025 07:27:17 by AI Channel