Angular: Introduzione al Framework

Angular è un framework front-end progettato per la scrittura di single page application. La sua nascita risponde a una necessità concreta: la complessità delle applicazioni web moderne è cresciuta in modo esponenziale, e Angular offre soluzioni standard e consolidate ai problemi più comuni che uno sviluppatore si trova ad affrontare. È mantenuto da Google e condivide il mercato con React e Vue, rappresentando una delle competenze più ricercate nel settore.

Caratteristiche fondamentali

Angular è un framework orientato ai componenti. Il frontend viene scomposto in unità indipendenti che comunicano tra loro tramite meccanismi controllabili e standardizzati. Ogni componente possiede la propria logica e la propria struttura, ben isolate dal resto dell'applicazione, il che riduce la complessità di gestione e massimizza la riusabilità. Vale la pena sottolineare che un progetto Angular non è composto esclusivamente da componenti, intesi come elementi renderizzabili visivamente, ma include anche elementi invisibili come i service, di cui si parlerà in seguito.

Lo sviluppo avviene interamente in TypeScript, che è un superset stretto di JavaScript. TypeScript aggiunge numerosi elementi assenti in JavaScript, primo tra tutti la tipizzazione statica delle variabili e dei metodi. Può essere pensato come un incrocio tra JavaScript e Java, con caratteristiche proprie. I vantaggi della tipizzazione statica si traducono nel controllo dei tipi in fase di compilazione e in una migliore rilevazione degli errori da parte dell'IDE. Il codice TypeScript viene poi tradotto in JavaScript puro, comprensibile dal browser.

Angular è inoltre un framework opinionated, ovvero fornisce soluzioni standard integrate per i problemi più comuni come il caricamento dei dati dalle API, la gestione dei form e il testing, senza la necessità di affidarsi a librerie esterne. Questo garantisce una grande uniformità tra i progetti Angular, al prezzo però di una scelta architetturale iniziale importante: a differenza di React o Vue, che

possono essere integrati parzialmente in un progetto già esistente, Angular governa l'intero progetto in modo pervasivo.

Angular dispone infine di propri strumenti di sviluppo, in particolare di una utility da linea di comando chiamata Angular CLI, richiamabile tramite il prefisso `ng`. Questa utility automatizza i task più comuni sia in fase di sviluppo che di test che di anteprima, ed è una differenza sostanziale rispetto ad altre soluzioni come React. La sua necessità deriva anche dal fatto che Angular tende ad essere verboso: qualsiasi sviluppo non banale si traduce in una proliferazione di file che sarebbe difficile gestire manualmente.

Ambiente di sviluppo e creazione del progetto

L'ambiente di sviluppo standard prevede l'uso di Visual Studio Code come IDE e di npm per installare Angular CLI. L'installazione dell'utility avviene tramite il comando `npm install -g @angular/cli`, che la rende disponibile globalmente sul sistema. Una volta installata, la creazione di un nuovo progetto avviene tramite il comando `ng new` seguito dal nome del progetto, senza più coinvolgere npm direttamente. Il risultato è una cartella contenente il progetto con una struttura di sottocartelle articolata, tra cui la cartella `app`, dove si svolge la maggior parte del lavoro.

Anatomia di un componente

Un componente è un pezzo tendenzialmente autonomo di interfaccia grafica, dotato di propria logica e gestibile separatamente rispetto alla pagina nel suo complesso. La creazione di un nuovo componente avviene tramite il comando `ng generate component`, abbreviabile in `ng g c`, seguito dal nome del componente. Angular CLI genera automaticamente una cartella dedicata contenente quattro file: il file HTML per il template, il file TypeScript per la logica, il file CSS per gli stili locali e il file `.spec.ts` per i test automatizzati.

L'unico file tecnicamente indispensabile è il file TypeScript, che rappresenta il componente in senso stretto. Nella pratica sarà sempre presente anche almeno il file del template HTML, ed è corretto considerare il componente come un insieme di due o quattro file che collaborano tra loro.

Il file TypeScript e il decoratore

Il file TypeScript del componente si apre con l'import del decoratore `@Component` dal modulo `@angular/core`. Questo decoratore, che in Java troverebbe un corrispettivo nelle annotation, comunica ad Angular che il file contiene un componente e specifica i metadati necessari al framework per gestirlo.

Tra i metadati più importanti vi è il `selector`, che definisce il tag HTML personalizzato con cui il componente viene richiamato nei template di altri componenti, detti componenti padre. Vi è poi `templateUrl`, che indica il percorso del file HTML del template, e `styleUrl`, che indica il file CSS degli stili locali. L'array `imports` consente infine di dichiarare le dipendenze del componente.

Dopo il decoratore viene definita la classe TypeScript che contiene la logica vera e propria del componente. Le variabili e i metodi dichiarati in questa classe sono automaticamente disponibili nel template HTML associato, senza necessità di passarli esplicitamente, poiché il template è legato implicitamente alla logica del componente.

Template, interpolazione e CSS locale

Il template HTML utilizza la sintassi di interpolazione, che consiste nel racchiudere un'espressione tra doppie parentesi graffe. Angular valuta l'espressione, ne converte il risultato in stringa e lo inserisce nel contenuto dell'elemento HTML corrispondente, aggiornando il DOM automaticamente ad ogni cambiamento del valore.

Il file CSS del componente, a differenza di quanto avviene nello sviluppo web tradizionale, non ha portata globale. Angular traduce le regole di stile in modo che si applichino esclusivamente al componente di riferimento, evitando qualsiasi conflitto con gli stili del resto dell'applicazione.

Composizione dei componenti e avvio dell'applicazione

Per utilizzare un componente all'interno di un altro è sufficiente inserire il suo selettore nel template del componente padre. Affinché questo funzioni, la logica del componente padre deve conoscere il componente figlio: è necessario

importarlo tramite un import TypeScript e registrarlo nell'array `imports` del decoratore `@Component`. Il componente radice dell'intera applicazione è `App`, e tutti gli altri componenti saranno suoi figli diretti o indiretti.

Per avviare l'applicazione in fase di sviluppo si utilizza il comando `ng serve`, che compila i file TypeScript, HTML e CSS traducendoli in qualcosa che il browser possa interpretare, e avvia un server di prova accessibile di default sulla porta 4200. Il comando include la funzionalità di watch, che ricompila automaticamente il progetto ad ogni modifica dei file sorgente.