GENERAL

angular 4 desde cero la gu a pr ctica y completa para convertirte en desarrollador angular full stack

Published: September 19, 2026 | Index ID: #-

Introduction

Angular has become one of the most popular frameworks for building dynamic, single page applications. Whether you're a beginner looking to dive into web development or an experienced developer wanting to master the full stack, learning **Angular 4 from scratch**provides a solid foundation for modern, scalable applications. This guide is designed to walk you through every step of the learning journey—from setting up your environment to deploying a production ready, full stack Angular project.

Understanding Angular 4

What is Angular?

Angular is a component based framework that allows developers to build complex, client side applications using TypeScript, HTML, and CSS. It follows the Model View Controller (MVC) architecture, providing a clear separation of concerns and making code easier to maintain.

Why Angular 4?

Released in 2017, Angular 4 introduced several performance improvements, reduced bundle size, and enhanced tooling. Although newer versions exist, Angular 4 remains a popular choice for educational purposes due to its stable API and comprehensive documentation. By mastering Angular 4, you'll acquire skills that translate directly to later releases.

Evolution and Relevance

Angular's evolution from AngularJS to Angular 4 and beyond reflects a shift from JavaScript to TypeScript, modular architecture, and a focus on performance. Understanding Angular 4 gives you a historical perspective that helps you adapt to future updates.

Prerequisites

Node.js & npm

Node.js is essential for running the Angular CLI and managing dependencies. Download the LTS version from nodejs.org and verify the installation with:

1. node -v
2. npm -v

IDE / Editor

Choose an editor that supports TypeScript and Angular extensions. Visual Studio Code is a popular choice because it offers built in IntelliSense, debugging, and a rich ecosystem of extensions.

TypeScript Fundamentals

Angular is built on TypeScript. A basic understanding of classes, interfaces, generics, and decorators will help you

navigate Angular’s codebase more efficiently. Resources such as the official TypeScript handbook can be a good starting point.

Setting Up the Environment

Installing Node and npm

After downloading Node.js, run the installer. Verify the installation:

node -v\nnpm -v\n\n\nInstalling Angular CLI

The Angular CLI simplifies project creation, development, and testing. Install it globally:

npm install -g @angular/cli@1.4.0\n\nNote: The @1.4.0 tag ensures compatibility with Angular 4.

Creating a New Angular 4 Project

Generate a new project with a custom Angular 4 version:

ng new angular4-tutorial --style=scss --routing\ncd angular4-tutorial\nng serve\n\nOpen http://localhost:4200 to see the default Angular application.

Project Structure Overview

Typical Angular 4 project layout:

- src/app – Application code (components, services, modules)
- src/assets – Static assets (images, fonts)
- src/environments – Environment-specific configurations
- src/index.html – Root HTML file
- src/styles.scss – Global styles

Core Concepts of Angular 4

Modules, Components, Templates

Angular applications are modular. Each feature is encapsulated in an **NgModule**that declares components, directives, and pipes. Components combine a TypeScript class with an HTML template and CSS styles.

Services and Dependency Injection

Services provide shared data and logic. Angular’s dependency injection system manages service instances, promoting reusability and testability.

Routing and Navigation

The Angular Router enables navigation between views without page reloads. Configure routes in app-routing.module.ts:

const routes: Routes = [\n { path: '', component: HomeComponent },\n { path: 'about', component: AboutComponent }\n];\n\n\nForms

- Template driven forms – Simpler, ideal for small forms.
- Reactive forms – Offer more control, especially for complex validation.

HTTP Client & Observables

Angular’s HttpClient service communicates with REST APIs. It returns Observable streams, enabling reactive

programming patterns.

Building a Sample Application

Project Idea: A Simple Blog

Let’s build a basic blog application with CRUD operations for posts. This will cover components, services, routing, and HTTP communication.

Creating Components

Generate components using the CLI:

ng generate component components/post-list\nng generate component components/post-detail\nng generate component components/post-form\n\n\nCreating Services

Generate a service to handle HTTP requests:

ng generate service services/post\n\nIn post.service.ts, implement methods like getPosts(), getPost(id), createPost(post), updatePost(id, post), and deletePost(id).

Implementing Routing

Define routes in app-routing.module.ts:

const routes: Routes = [\n { path: '', component: PostListComponent },\n { path: 'post/:id', component: PostDetailComponent },\n { path: 'new', component: PostFormComponent }\n];\n\n\nData Binding and Directives

Use {{}} interpolation for data display, ngFor for lists, and ngIf for conditional rendering. Example:

\n <li *ngFor="let post of posts">\n <a [routerLink]="['/post', post.id]">{{ post.title }}\n \n
\n\n\nUsing Pipes

Pipes transform displayed data. Angular ships with many built in pipes (e.g., date, currency). Create custom pipes for specialized formatting.

Consuming REST API

Configure HttpClientModule in app.module.ts and inject PostService into components. Use subscribe() to handle asynchronous data.

State Management

Using RxJS

RxJS operators like map, filter, and switchMap help transform streams. Combine multiple API calls with forkJoin or combineLatest.

NgRx Basics

For larger applications, NgRx provides Redux style state management. Set up actions, reducers, and selectors to keep state predictable.

Testing

Unit Tests with Jasmine/Karma

Angular CLI scaffolds component tests. Use TestBed to configure testing modules and it() blocks for assertions.

End to End Tests with Protractor

Protractor simulates user interactions. Write tests to verify navigation, form submissions, and API responses.

Deployment

Building for Production

Generate an optimized bundle:

ng build --prod\n\nThis command minifies code, extracts CSS, and performs Ahead of Time (AOT) compilation.

Hosting Options

- Static Hosting – GitHub Pages, Netlify, Vercel.
- Cloud Platforms – AWS S3 + CloudFront, Azure Static Web Apps.
- Server Side Rendering – Angular Universal for SEO benefits.

Full Stack Integration

Backend Options

- Node.js + Express – RESTful API with JSON Web Tokens.
- .NET Core – Strongly typed APIs, integration with SQL Server.
- Java (Spring Boot) – Robust enterprise backend.

Auth with JWT

Implement token based authentication. Store JWTs in localStorage and attach them to HttpHeaders. Protect routes with AuthGuard.

WebSockets and Real Time

Baca Juga (Artikel Terkait)

• **advanced engineering electromagnetics solution manual edition hardcover by balanis constantine a published by wiley**

URL: http://dtrainderailed.com/advanced-engineering-electromagnetics-solution-manual-edition-hardcover-by-balanis-constantine-a-published-by-wiley.pdf

• **advanced engineering electromagnetics solutions manual**

URL: http://dtrainderailed.com/advanced-engineering-electromagnetics-solutions-manual.pdf

• **advanced engineering electromagnetics wiley 1989 grading**

URL: http://dtrainderailed.com/advanced-engineering-electromagnetics-wiley-1989-grading.pdf

• **advanced engineering fluid mechanics by biswas**

URL: http://dtrainderailed.com/advanced-engineering-fluid-mechanics-by-biswas.pdf