**Chapter 1: Introduction to Angular**

**Angular** is a powerful, full-featured, open-source framework developed by **Google** for building dynamic and scalable web applications. Unlike lightweight libraries such as **React** or **Vue**, Angular offers a **complete solution** — including routing, forms, HTTP services, dependency injection, and more — out of the box.

Angular is built on **TypeScript**, a superset of JavaScript that adds static typing and class-based object-oriented programming, which makes large-scale applications more robust and easier to maintain.

Angular is especially suitable when:

* You're building **enterprise-grade, feature-rich applications**
* You want **strong typing and tooling support** (TypeScript)
* You need **built-in solutions** for forms, routing, HTTP, etc.
* You want a **scalable architecture** for long-term projects

**Key Features:**

* **Standalone components (v14+)** — no more boilerplate NgModules
* **Signals (v16+)** — fine-grained reactivity built into the framework
* **Functional route guards (v15+)** — easier, stateless security logic
* **Zoneless apps (v17+)** — remove NgZone for leaner performance
* **Modern control flow syntax (v17+)** — @if, @for, @switch

**Angular vs AngularJS**

Angular (v2+) is a **complete rewrite** of AngularJS (v1.x). They share the name but differ in:

|  |  |  |
| --- | --- | --- |
| Feature | AngularJS (v1.x) | Angular (v2+) |
| Language | JavaScript | TypeScript |
| Architecture | MVC-style | Component-based |
| Mobile Support | No | Yes (via Angular + Ionic) |
| Performance | Slow for large apps | Fast with Ahead-of-Time (AOT) comp |
| Tooling | Basic | Advanced (CLI, IDE support, etc.) |

**Angular Version History**

Here’s a table summarizing the evolutio­­­n of Angular:

|  |  |  |
| --- | --- | --- |
| Version | Release Year | Key Features |
| v2 | 2016 | First complete rewrite, components, TypeScript |
| v4 | 2017 | Smaller bundles, \*ngIf with else |
| v5 | 2017 | HttpClient, build optimizer |
| v6 | 2018 | CLI Workspaces, ng update |
| v7 | 2018 | Virtual scrolling, DragDrop |
| v8 | 2019 | Differential loading, Ivy preview |
| v9 | 2020 | Ivy renderer by default |
| v10 | 2020 | TypeScript 3.9, stricter defaults |
| v11 | 2020 | Improved HMR, faster builds |
| v12 | 2021 | Nullish coalescing, standalone style APIs |
| v13 | 2021 | No View Engine, ESBuild support |
| v14 | 2022 | 🧩 Standalone Components introduced |
| v15 | 2022 | Functional guards, improved APIs |
| v16 | 2023 | 🚦 Signals API, hydration, esbuild builds |
| v17 | 2023 | ⚡ Zoneless Angular, control flow @if, @for |
| v18 | 2024 | Stable signals, router refactor, default standalone |
| v19 | 2025 | Signal-first, full module opt-out, better SSR support |

**Setting Up Angular CLI**

**Prerequisites**

* [Node.js (LTS version)](https://nodejs.org/)
* npm (comes with Node.js)

**✅ Install Angular CLI**

npm install -g @angular/cli

Verify version:

ng version

**Creating Your First Angular Project**

**Command:**

ng new my-app --standalone

**Options Explained:**

* --standalone is built in Angular 19, No need to explicitly add it from version 19.
* my-app: Name of your project folder
* --standalone: Enables Angular’s **standalone API** — no need for NgModules

**Project Structure Overview**

Here's what Angular CLI generates (important files highlighted):

my-app/

│

├── src/

│ ├── app/

│ │ ├── app.component.ts // Root component

│ │ └── app.config.ts // App configuration (routing, providers)

│ ├── main.ts // Bootstraps the app

│ └── index.html // Host HTML file

│

├── angular.json // CLI build & project config

├── package.json // npm dependencies

└── tsconfig.json // TypeScript configuration

**Angular CLI Commands**

The Angular CLI is your best friend during development. Here are must-know commands:

|  |  |
| --- | --- |
| Command | Description |
| ng serve | Runs development server at localhost:4200 |
| ng build | Compiles and bundles your app |
| ng generate component my-comp | Generates a new component |
| ng generate service my-service | Generates a service |
| ng test | Runs unit tests |
| ng add @angular/material | Installs libraries with CLI integration |

## Angular Workspace

An **Angular workspace** is the foundational structure that lets you manage **multiple Angular projects**—including applications and libraries—under a **single configuration**.

Purpose of a Workspace

Think of an Angular workspace as a **monorepo** for Angular. It exists to:

#### ✅ ****1. Manage Multiple Applications****

You can build:

* Separate apps (e.g., admin-panel, user-portal)
* Micro frontends
* Angular Universal (SSR) projects

All **under one folder**, without needing separate repositories.

#### ✅ ****2. Share Common Code****

You can create **libraries** for:

* UI components
* Utility functions
* Services or APIs

These libraries can be reused **across multiple apps**, avoiding code duplication.

#### ✅ ****3. Centralized Tooling and Configuration****

All projects share:

* Common build configuration (angular.json)
* Shared TypeScript settings (tsconfig.base.json)
* Unified dependency management (package.json)

#### ✅ ****4. Scalable for Large Teams****

Different teams can work on different apps or libraries within the same repo while still sharing a common structure.

## Step-by-Step: Create a Workspace with Multiple Projects

### 1. ****Create a New Workspace Without an App****

ng new my-workspace –no-create-application

Creates a workspace that’s ready for multiple apps and libraries.

### 2. ****Add a Standalone Application****

ng generate application admin-app --standalone

Adds a standalone app with no NgModule. You can also add:

ng generate application user-portal --standalone

### 3. ****Add a Shared Library****

ng generate library shared-utils

Creates a reusable library in projects/shared-utils.

## 📁 Project File Structure Overview

my-workspace/

├── angular.json // Workspace and project build configs

├── package.json // Shared npm dependencies

├── tsconfig.base.json // Base TypeScript config for all projects

├── projects/

│ ├── admin-app/ // Standalone application

│ ├── user-portal/ // Another app

│ └── shared-utils/ // Reusable library

## Generate Angular Artifacts in a Specific Project

|  |  |
| --- | --- |
| Task | Command Example |
| Generate Component | ng g c name --project=admin-app |
| Generate Service | ng g s path/name --project=user-portal |
| Generate Directive | ng g d name --project=shared-utils |
| Generate Pipe | ng g p name --project=shared-utils |

## Angular CLI Commands for Workspace Projects

|  |  |
| --- | --- |
| Command | Description |
| ng serve admin-app | Serves the admin-app |
| ng build user-portal | Builds the user-portal |
| ng test shared-utils | Runs tests for the shared-utils library |
| ng generate component xyz --project=admin-app | Generates a component in a specific project |