Complete Angular Tutorial For Beginners

← Best Angular Books Introduction to Angular →

This **Angular Tutorial** helps you to learn the concepts of Angular. Out step by step guides walk you through building Angular Applications, and adding Components, Directives, Pipes, etc. Learn how to organize Applications using Modules, Navigate Using Routers, etc. The tutorial also covers advanced topics like component communications, Services, Forms, routers, HTTP communications, observable, SEO, etc. This Tutorial takes a simple, step-by-step approach and includes many examples and codes.

This Tutorial covers all versions of Angular, Starting from Angular 2 to the latest editions, i.e., Angular 17.

Table of Contents

What is Angular

Angular is a Javascript Framework-based development platform for building a single-page application for mobile and desktop. It uses Typescript & HTML to build Apps. The Angular itself is written using Typescript. It now comes with every feature you need to develop a complex and sophisticated web or mobile application. Angular is loaded

with features like components, Directives, Forms, Pipes, HTTP Services, Dependency Injection, etc.

Angular Versions

The early version of Angular was named Angular 2. Then later, it was renamed to just Angular. You can learn about Angular Versions

Prerequisites

This tutorial requires a working knowledge of Javascript, TypeScript, HTML & CSS. It also requires the concept of OOP.

We are going to use **Typescript** as our language. You will find it very easy if you know C# or Java. You can learn Typescript from the Typescript Tutorial.

Angular Tutorial

Introduction to Angular

This Introduction to Angular gives you a glimpse of Angular. Angular is a Javascript framework for building mobile and desktop web applications. It is built using Javascript. You can build amazing client-side applications using HTML, CSS, and Typescript using Angular. Knowing how the Angular framework works is essential before you start using it. The following tutorials introduce you to Angular, and you will learn Angular's architecture.

1. Introduction to Angular
2. Angular Architecture Overview & Concepts
3. Install Angular

Getting Started With Angular

Angular has undergone many changes since the version of Angular 2. From the Angular 7 version, installing and creating a new Angular project has become very simple. You only need to install and Visual Studio code, NPM Package Manager & Angular CLI. Once you install the required dependencies, creating a new project is as easy as running a simple command ng new. Angular CLI takes care of the Configuration & initialization of various libraries.

You can read the following step-by-step tutorial to learn how to create an Angular application and get started.

1. How to Create a new project in Angular
2. Bootstrapping in Angular

Components

The Component is the main building block of an Angular Application. A Component contains the definition of the View and the data that defines how the View looks and behaves. The Angular Components are plain javascript classes and defined using. @component Decorator. This Decorator provides the component with the View to display & Metadata about the class.

The Component passes the data to the view using a process called Data Binding. This is done by Binding the DOM Elements to component properties. Binding can display component class property values to the user, change element styles, respond to a user event, etc.

1. Introduction to Angular Components
2. Data Binding
3. Interpolation in Angular
4. Property Binding
5. Event Binding
6. Two-way Binding & ngModel
7. ngModelChange & Change Event
8. Adding Child Component
9. Standalone Components

Directives

The Angular directive helps us to manipulate the DOM. You can change a DOM element's appearance, behavior, or layout using the directives. They help you to extend HTML. The Angular directives are classified into three categories based on how they behave. They are Component, Structural, and Attribute Directives

The ngFor is an Angular structural directive that repeats a portion of the HTML template once per each item from an iterable list (Collection). The ngSwitch allows us to Add/Remove DOM elements. It is similar to the switch statement of Javascript. The ngIf allows us to Add/Remove DOM elements.

The ngClass Directive is an Angular Attribute Directive, which allows us to add or remove CSS classes to an HTML element. The ngStyle directive allows you to modify the style of an HTML element using the expression. Using the ngStyle, you can dynamically change the style of your HTML element.

1. Angular Directives
2. ngFor
3. ngSwitch
4. ngIf
5. ngClass
6. ngStyle
7. ngFor Trackby
8. Custom Directive

Pipes

The Angular pipes are used to Transform the Data. For Example, the Date pipe formats the date according to locale rules. We can pass arguments to pipe and chain pipes. Angular also allows us to create a Custom Pipe.

1. Angular Pipes
2. Angular Custom Pipes
3. Date Pipe
4. Async Pipe
5. KeyValue Pipe
6. Using Pipes in Components & Services

Component Communication

1. Angular Component Communication
2. Passing data from Parent to child component
3. Passing Data from Child to Parent Component

Component Life Cycle Hook

The life cycle hooks are the methods that angular invokes on directives and components as it creates, changes, and destroys them. Using life-cycle hooks, we can fine-tune the behavior of our components during creation, update, and destruction.

1. Component Life Cycle
2. OnInit & OnDestroy
3. Onchanges
4. DoCheck

Angular Forms

The data entry forms can be very simple to very complex. The Forms contain a large no of input fields, a variety of fields like Text boxes, Dates, Numbers, Emails, Passwords, Check Boxes, Option boxes, etc. These fields can Span multiple tabs or multiple pages. Forms may also contain complex validation logic interdependent on multiple fields.

The Angular forms modules are designed to handle the above and more. Angular Forms now supports the Reactive forms approach to Forms development. The older way of Template-based approach is also supported.

1. Angular Forms Tutorial: Fundamental & Concepts
2. Template-Driven Forms in Angular
3. Set Value in Template Driven forms in Angular
4. Reactive Forms in Angular
5. FormBuilder in Reactive Forms
6. SetValue & PatchValue in Angular
7. StatusChanges in Angular Forms
8. ValueChanges in Angular Forms
9. FormControl
10. FormGroup
11. FormArray Example
12. Build Dynamic or Nested Forms using FormArray
13. SetValue & PatchValue in FormArray
14. Select Options Dropdown
15. Typed Forms in Angular

16. FormRecord in Angular

Services & Dependency Injection

Services allow us to create reusable code and use it for every component that needs it. The Services can be injected into components and other services using the dependency injection system. The dependencies are declared in the Module using the Provider's metadata. The Angular creates a tree of injectors & Providers that resembles the Component Tree. This is called the hierarchical pattern.

1. Services
2. Dependency injection
3. Injector, @Injectable & @Inject
4. Providers
5. Injection Token
6. Hierarchical Dependency Injection
7. Angular Singleton Service
8. ProvidedIn root, any & platform
9. @Self, @SkipSelf & @Optional Decorators
10. @Host Decorator in Angular

11. ViewProviders

Angular Forms Validation

One of the common tasks that is performed while building a form is Validation. The Forms Validation is built into the Angular Forms Module. Angular provides several Built-in validators out of the box. If those validators do not fit your needs, you can create your own custom validator.

1. Validations in Reactive Forms in Angular
2. Custom Validator in Reactive Forms
3. Passing Parameter to Custom Validator in Reactive Forms
4. Inject Service into Custom Validator
5. Validation in Template-Driven Forms
6. Custom Validator in Template-Driven Forms
7. Angular Async Validator
8. Cross Field Validation
9. Adding Validators Using SetValidators

HTTP

The newly designed HttpClient Module allows us to query the Remote API source to get data into our Application. It requires us to Subscribe to the returned response using RxJs observables.

1. Angular HTTP Client Tutorial
2. HTTP GET Example
3. HTTP POST Example

4. Passing URL Parameters (Query strings)
5. HTTP Headers Example
6. HTTP Interceptor

Angular Router

The Router module handles the navigation & Routing in Angular. The Routing allows you to move from one part of the application to another part or one View to another View.

1. Angular Router
2. Angular Router with standalone components
3. Location Strategies in Angular Router
4. Passing Parameters to Route
5. Child Routes / Nested Routes
6. Query Parameters
7. Navigation between Routes
8. Angular Pass data to route
9. RouterLinkActive
10. Router Events
11. ActivatedRoute

Angular Route Guards

1. Angular Route Guards
2. CanActivate Guard
3. CanActivateChild Guard

4. CanDeactivate Guard
5. Angular Resolve Guard

Angular Module

The Angular Modules help us to organize our code into manageable parts or blocks. Each block implements a specific feature. The Components, Templates, Directives, Pipes, and Services that implement that feature become part of the module. The following tutorial explains how best you can create an Angular Module, The folder structure you can use, etc. We can also load the Modules lazily or Preload them, thus improving the application's performance.

1. Introduction to Angular Modules
2. Routing Between Angular Modules
3. Angular folder structure: Best Practices
4. Lazy Loading in Angular
5. Preloading Strategy
6. CanLoad Guard

Advanced Components

The Components in Angular are very powerful features. The following tutorials take you through some of the important features of the Angular Component.

1. Ng-Content & Content Projection in Angular
2. Angular @input, @output & EventEmitter
3. Template Reference Variable in Angular
4. ng-container in Angular

5. ng-template & TemplateRef in angular
6. ngtemplateoutlet in angular
7. HostBinding & HostListener
8. ViewChild, ViewChildren & QueryList
9. ElementRef
10. Renderer2
11. ContentChild & ContentChildren
12. AfterViewInit, AfterViewChecked, AfterContentInit & AfterContentChecked
13. Angular Decorators

Observable in Angular

Angular uses the Observable Pattern extensively. The following tutorials give you an introduction to observable and how to use it in an Angular Application.

1. Angular Observable Tutorial
2. Create an Observable from a string, array, object, collection
3. Observable from events using fromEvent
4. Observable pipe
5. Map Operator
6. Filter Operator
7. Tap Operator
8. SwitchMap
9. MergeMap
10. ConcatMap
11. ExhaustMap

12. take, takeUntil, takeWhile, takeLast
13. First, last & Single
14. Skip, SkipWhile, SkipUntil & SkipLast
15. Scan & Reduce
16. DebounceTime & Debounce
17. Delay & DelayWhen
18. ThrowError
19. CatchError
20. Retry & ReTryWhen
21. Unsubscribe from an observable
22. Subjects in Angular
23. ReplaySubject, BehaviorSubject & AsyncSubject
24. Angular Subject Example

Styling the Application

Angular uses several different ways to style the Application. You can style the app globally and then override it locally in the component very easily. The component styles have local scope, which is achieved using the various View Encapsulation strategies. Learn all these in the section

1. Angular Global Styles
2. View Encapsulation
3. Style binding in Angular
4. Class Binding in Angular
5. Component Styles
6. How to Install & Use Angular FontAwesome
7. How to Add Bootstrap to Angular

Others

1. Location Service

Configuration

The apps usually need some Run-time configuration information like URL endpoint etc., which it needs to load at startup. Also, different environments like development, production & testing require different environments, etc.

1. How to use APP_INITIALIZER
2. Run time configuration
3. Environment Variables

Handling Errors

In the following Angular tutorials, we look at how Angular handles errors. We handle errors by setting up a Global Error handler or custom error handler. Also,.whenever the error occurs in an HTTP operation, Angular wraps it in an `httpErrorResponse` Object. Learn how to handle HTTP Errors also.

1. Error Handling in Angular
2. HTTP Error Handling

Angular CLI

Learn how to use Angular CLI to speed up the development of Angular Application

1. Angular CLI Tutorial
2. ng new
3. Upgrading Angular App to the latest version
4. Migrating to Standalone Components
5. Multiple Apps in One Angular Project

SEO & Angular

Your Website is useless if the Search Engines do not find it. There are many things you must consider to make your App SEO-friendly. You need to set up Title & Meta Tags for each page. Ensure that the search engines can crawl and read your page. Set the correct Canonical URL for each page etc. Also, ensure that the app loads quickly, etc. The Following Angular Tutorials guide you through some of the important SEO features.

1. Title Service Example
2. Dynamic Title based on Route
3. Meta Service
4. Dynamic Meta Tags
5. Canonical URL
6. Lazy Load Images in Angular

Angular Universal

The following Angular Universal Tutorial explains how to achieve Server Side Rendering using Angular Universal. The App's rendering on the server side makes it load quickly and ensures that the search engines can crawl the content.

1. Server-Side Rendering Angular Universal Tutorial

Building & Hosting

1. The Requested URL is not found on this server

Angular Resources

1. Angular Examples & Projects
2. The Angular Learning Resources

Angular how-to guides

1. How to get the current Route
2. ExpressionChangedAfterItHasBeenCheckedError in Angular
3. Angular CLI Check Version
4. Property 'value' does not exist on type 'EventTarget' Error in Angular

Module Loaders

The Angular application can use either SystemJs or Webpack module loader. We will demonstrate how to use both loaders by building a small application.

1. Create Angular Application using SystemJS