

Table of Contents

- **Chapter 1. Introducing Angular**
 - What is Angular?
 - Central Features of the Angular Framework
 - Appropriate Use Cases
 - Building Blocks of an Angular Application
 - Basic Architecture of an Angular Application
 - Installing and Using Angular
 - Anatomy of an Angular Application
 - Running the Application

- **Chapter 2. Introduction to TypeScript**
 - TypeScript Syntax
 - Programming Editors
 - The Type System – Defining Variables
 - The Type System – Defining Arrays
 - Type in Functions
 - Type Inference
 - Defining Classes
 - Class Methods
 - Class Constructors
 - Class Constructors – Alternate Form
 - Interfaces
 - Working with ES6 Modules
 - Visibility Control
 - var vs let
 - Arrow Functions
 - Arrow Function Compact Syntax
 - Arrow Function and Caller Context
 - Template Strings
 - Generics in Class
 - Generics in Function

- **Chapter 3. Components**
 - What is a Component?
 - An Example Component
 - Creating a Component Using Angular CLI
 - The Component Class
 - The @Component Decorator
 - Registering a Component to Its Module
 - Component Template
 - Using a Component

- Run the Application
- Component Hierarchy
- The Application Root Component
- The Bootstrap File

- **Chapter 4. Component Templates**
 - Templates
 - Template Location
 - The Mustache {{ }} Syntax
 - Setting DOM Element Properties
 - Event Binding
 - Expression Event Handler
 - Prevent Default Handling
 - Attribute Directives
 - Apply Styles by Changing CSS Classes
 - Example: ngClass
 - Applying Styles Directly
 - Structural Directives
 - Conditionally Execute Template
 - Example: ngIf
 - Looping Using ngFor
 - ngFor Local Variables
 - Manipulating the Collection
 - Example - Deleting an Item
 - Item Tracking with ngFor
 - Swapping Elements with ngSwitch
 - Template Reference Variable

- **Chapter 5. Inter Component Communication**
 - Communication Basics
 - The Data Flow Architecture
 - Preparing the Child to Receive Data
 - Send Data from Parent
 - More About Setting Properties
 - Firing Event from a Component
 - @Output () Example - Child Component
 - @Output () Example - Parent Component
 - Full Two-Way Binding
 - Setting up Two Way Data Binding in Parent

- **Chapter 6. Template Driven Forms**
 - Template Driven Forms
 - Importing Forms Module
 - Basic Approach
 - Setting Up a Form
 - Getting User Input
 - Omitting ngForm Attribute
 - Initialize the Form

- Two Way Data Binding
- Form Validation
- Angular Validators
- Displaying Validation State Using Classes
- Additional Input Types
- Checkboxes
- Select (Drop Down) Fields
- Rendering Options for Select (Drop Down)
- Date fields
- Radio Buttons

- **Chapter 7. Reactive Forms**
 - Reactive Forms Overview
 - The Building Blocks
 - Import ReactiveFormsModule
 - Construct a Form
 - Design the Template
 - FormControl Constructor
 - Getting Form Values
 - Setting Form Values
 - The Synchronous Nature
 - Subscribing to Input Changes
 - Validation
 - Built-In Validators
 - Showing Validation Error
 - Custom Validator
 - Using a Custom Validator

- **Chapter 8. Services and Dependency Injection**
 - What is a Service?
 - Creating a Basic Service
 - The Service Class
 - What is Dependency Injection?
 - Injecting a Service Instance
 - Injectors

- **Chapter 9. Pipes and Data Formatting**
 - What are Pipes?
 - Built-In Pipes
 - Using Pipes in HTML Template
 - Chaining Pipes
 - The number Pipe
 - Currency Pipe
 - Create a Custom Pipe
 - Custom Pipe Example
 - Using Custom Pipes
 - Using a Pipe with ngFor
 - A Filter Pipe

- **Chapter 10. HTTP Client**
 - The Angular HTTP Client
 - Using The HTTP Client - Overview
 - Importing HttpClientModule
 - Simple Example
 - Service Using HttpClient
 - ES6 Import Statements
 - Making a GET Request
 - What does an Observable Object do?
 - Using the Service in a Component
 - Error Handling
 - Customizing Error Object with . catch ()
 - Making a POST Request
 - Making a PUT Request
 - Making a DELETE Request

- **Chapter 11. The Angular Component Router**
 - The Component Router
 - View Navigation
 - The Angular Router API
 - Creating a Router Enabled Application
 - Hosting the Routed Components
 - Navigation Using Links and Buttons
 - Programmatic Navigation
 - Passing Route Parameters
 - Navigating with Route Parameters
 - Obtaining the Route Parameter Values
 - Retrieving a Route Parameter
 - Routing Enabled Feature Module
 - Using the Feature Module
 - Lazy Loading the Feature Module
 - Creating Links for the Feature Module
 - Components
 - More About Lazy Loading
 - Preloading Modules
 - routerLinkActive binding
 - Default Route
 - Wildcard Route Path
 - redirectTo
 - Child Routes
 - Defining Child Routes
 - for Child Routes
 - Links for Child Routes
 - Navigation Guards
 - Creating Guard Implementations
 - Using Guards in a Route

- **Chapter 12. Advanced HTTP Client**
 - Request Options
 - Returning an HttpResponse Object
 - Setting Request Headers
 - Creating New Observables
 - Creating a Simple Observable
 - The Observable.create() Method
 - Observable Operators
 - More About map
 - Piping Operators
 - The flatMap() Operator
 - The tap() Operator
 - The zip() Operator
 - Caching HTTP Response
 - Making Sequential HTTP Calls
 - Making Parallel Calls
 - Customizing Error Object with catchError ()

- **Chapter 16. Unit Testing Angular Applications**
 - Unit Testing Angular Artifacts
 - Testing Tools
 - Typical Testing Steps
 - Test Results
 - Jasmine Test Suites
 - Jasmine Specs (Unit Tests)
 - Expectations (Assertions)
 - Matchers
 - Examples of Using Matchers
 - Using the not Property
 - Setup and Teardown in Unit Test Suites
 - Example of before Each and after Each Functions
 - Angular Test Module
 - Example Angular Test Module
 - Testing a Service
 - Injecting a Service Instance
 - Test a Synchronous Method
 - Test an Asynchronous Method
 - Using Mock HTTP Client
 - Supplying Canned Response
 - Testing a Component
 - Component Test Module
 - Creating a Component Instance
 - The Component Fixture Class
 - Basic Component Tests
 - The Debug Element Class