Course Outline

Comprehensive Angular

Course Summary

Description

Angular allows developers to easily build dynamic, responsive single-page web applications that dynamically rewrite portions of the current page rather than having to generate a new page in response to every request. The training teaches attendees how to build applications using ES6, TypeScript, and modern front-end tools, including npm and Webpack. Students also gain an understanding of application architecture and design best practices in Angular, as well as learn how to authenticate, unit test, and manage application state in an Angular application.

Note: This course is taught in the current version of Angular at the time of teaching.

Objectives

At the end of this course, students will be able to:

- Understand how single-page web application architectures are different than traditional web application architectures
- Use new JavaScript (ES6) language features including Classes, Modules, and Arrow Functions
- Use new TypeScript language features including Types, Decorators, Interfaces, and Generics
- Learn Angular coding and architecture best practices including project layout and using container and presentation components
- Understand and use Angular model-driven forms, observables, dependency injection, and routing
- Communicate with a backend server using Angular's HttpClient to load and save data
- Configure the router and navigate between components
- Unit test all parts of an application including Components, Services, and Pipes
- Implement Authentication and Authorization in an Angular Application
- Optimize Angular Performance by changing Change Detection Strategies
- Setup new projects from scratch using the Angular CLI
- Scaffold modules, components, services, models, routes, and unit tests in accordance with best practices using the Angular CLI
- Write End-to-End Tests (optional; taught only if this applies to your group)
- Upgrade an existing application from AngularJS to the current Angular version (optional; taught only if this applies to your group)

Topics

- Introduction
- TypeScript and ECMAScript 6 (ES6) Fundamentals
- Angular Overview
- Components
- Angular Modules (NgModule)
- Project Set-Up (Using the Angular CLI)
- Data Binding
- Directives
- Pipes
- Advanced Components
- Services & Dependency Injection
- Dependency Injection
- Model-driven Forms (Reactive Forms)
- Communicating with the Server using the HttpClient Service
- Router
- Deploying an Angular Application to Production
- Upgrading to the latest version of Angular from earlier versions
- RxJS and Observables
- Unit Testing
- Security
- Change Detection
- Advanced Angular CLI
- Advanced Routing
- Advanced Dependency Injection
- Pipes

Course Summary (cont'd)

Audience

This course is designed for those wanting to learn how to build an application from scratch using Angular.

Prerequisites

Students must have object-oriented programming experience. Some experience with JavaScript is helpful; the new language features of JavaScript and TypeScript are covered/reviewed in class.

Duration

Five days

Course Outline

Comprehensive Angular

Course Outline

I. *Introduction*

II. *TypeScript and ECMAScript 6 (ES6) Fundamentals*
- A. TypeScript Installation, Configuration & Compilation
- B. Type Annotations
- C. Classes
- D. Scoping using let, var, and const Keywords
- E. Arrow Functions
- F. ES Modules
- G. Decorators
- H. Template Literals
- I. Spread Syntax and Rest Parameters
- J. Destructuring

III. *Angular Overview*
- A. Benefits of Building using Angular
- B. Understanding Angular Versions
- C. Single-page Web Application Architectures vs. Traditional Server-side Web Application Architectures
- D. Angular Style Guide
- E. Angular Architecture
- F. Angular Compared to Other JavaScript Libraries and Frameworks (React, VueJS, etc…)
- G. Your First Angular Application

IV. *Components*
- A. Understanding Components
- B. Component Properties & Methods
- C. Templates: Inline, Multi-line, and External with Component-relative Paths

V. *Angular Modules (NgModule)*
- A. Angular Modules vs. ES Modules
- B. Organizing your code into Feature Modules

VI. *Project Set-Up (Using the Angular CLI)*
- A. Angular CLI Features
- B. Creating a New Project (with new CLI Prompts)
- C. Generating Code
- D. Customizing the Angular CLI

VII. *Data Binding*
- A. Interpolation
- B. Property binding
- C. Event binding
- D. Two-way data binding

VIII. *Directives*
- A. Structural: ngFor, ngIf, ngSwitch
- B. Attribute: ngClass, ngStyle

IX. *Pipes*
- A. Built-in Pipes: Using, Passing Parameters, Chaining

X. *Advanced Components*
- A. Component Communication using @Input, @Output
- B. Component Architecture
- C. Component Styles
- D. Component Lifecycle Hooks
- E. Evaluating UI Component Frameworks & Libraries

XI. *Services & Dependency Injection*
- A. Using a service to access data
- B. Using a service to encapsulate business logic
- C. Understanding the scope of services

XII. *Dependency Injection*
- A. Understanding Dependency Injection
- B. Angular's Dependency Injection System
- C. Registering
- D. Injecting

XIII. *Model-driven Forms (Reactive Forms)*
- A. Importing the ReactiveFormsModule
- B. FormControl, FormGroup, and AbstractControl
- C. Binding DOM Elements to FormGroups and FormControls
- D. Validation Rules, Messages, and Styles
- E. Refactoring Reactive Forms for Reuse
- F. Custom Validators

Course Outline (cont'd)

XIV. Communicating with the Server using the HttpClient Service
- A. Deciding between Promises or Observables (RxJS)
- B. Making an HTTP GET Request
- C. Sending data to the server using Http POST and PUT Requests
- D. Issuing an HTTP DELETE Request
- E. Intercepting Requests and Responses

XV. Router
- A. Importing the RouterModule
- B. Configuring Routes
- C. Displaying Components using a RouterOutlet
- D. Navigating declaratively with RouterLink
- E. Navigating with code using the Router
- F. Accessing parameters using ActivatedRoute

XVI. Deploying an Angular Application to Production
- A. Building an application using the Angular CLI
- B. Differential loading: creating a modern build (ES2015) and a legacy build (ES5)
- C. Deploying to a web server

XVII. Upgrading to the latest version of Angular from earlier versions
- A. 2.x and above
- B. Update Guide
- C. Deprecation Guide
- D. AngularJS to Angular upgrades is covered in a separate course

XVIII. RxJS and Observables
- A. What is an Observable?
- B. Creating Observables
- C. What is an Observer?
- D. Observer Example
- E. Operators: map, switchMap, debounceTime, distinctUntilChanged
- F. Practical Application of using RxJS
- G. Subscriptions
- H. Unsubscribing from Observables in Angular (unsubscribe, Async Pipe, takeUntil)
- I. Subject
- J. Subject Example
- K. Subject Variants (AsyncSubject,BehaviorSubject, ReplaySubject)
- L. EventEmitter or Observable
- M. RxJS Operators and HTTP

XIX. Unit Testing
- A. Tools: Jasmine, Karma
- B. Jasmine Syntax: describe, it, beforeEach, afterEach, matchers
- C. Setup and your First Test
- D. Testing Terminology: Mock, Stub, Spy, Fakes
- E. Angular Testing Terminology: TestBed, ComponentFixture, debugElement, async, fakeAsync, tick, inject
- F. Simple Component Test
- G. Detecting Component Changes
- H. Testing a Component with properties (inputs) and events (outputs)
- I. Testing a Component that uses the Router
- J. Testing a Component that depends on a Service
- K. Testing a Service and Mocking its HTTP requests
- L. Testing a Pipe

XX. Security
- A. Best Practices
- B. Preventing Cross-site Scripting (XSS)
- C. Trusting values with the DOMSanitizer
- D. HTTP Attacks (CSRF and CSSI)
- E. Authentication using JSON Web Tokens (JWT)
- F. Authorization: Router Guards

XXI. Change Detection
- A. Understanding Zone.js and Change Detection
- B. Change Detection Strategies Default and OnPush

Course Outline (cont'd)

XXII. Advanced Angular CLI
- A. Customizing a build using Builder APIs in the CLI
- B. Generating web workers

XXIII. Advanced Routing
- A. Lazy-loading Angular Modules (using Dynamic Imports)
- B. Nested or Child Routes

XXIV. Advanced Dependency Injection
- A. Providers
- B. Hierarchical Injection
- C. providedIn options: root, module, platform, any

XXV. Pipes
- A. Creating a custom Pipe using PipeTransform
- B. Understanding Pure and Impure Pipes

XXVI. Conclusion

XXVII. Choose any two additional topics. If desired, the course can be customized to include more than two of these topics if other topics are scaled back or removed.
- A. npm QuickStart
  1. Installing Dependencies
  2. Understanding package.json and package-lock.json
  3. Using npm as a Build Tool
- B. Managing Shared Application State using ngrx and Redux
  1. Benefits Overview
  2. Three Principles of Redux: Single Source of Truth, State is Read-Only, Pure Functions
  3. Examples of Pure Functions
  4. Reducers
  5. Simple ngrx Example
  6. Time-traveling with Redux Devtools
  7. Full ngrx Example Application
  8. Upgrade Strategies from AngularJS
  9. High-level Approaches
  10. Concept Mapping AngularJS to Angular
- C. UpgradeAdapter
  1. What can be Upgraded or Downgraded
  2. What cannot be Upgraded or Downgraded
  3. UpgradeAdapter and Dependency Injection
- D. End-to-End Testing
  1. What is Protractor?
  2. Why Protractor?
  3. Using Locators
  4. Page Objects
  5. Debugging E2E Tests