ANGULAR 4 TRAINING OBJECTIVES

DURATION: - 5 DAYS

All students will learn to:

- Understand how Angular is different than traditional web development frameworks
- Code using new ES6 and TypeScript language features
- Develop an application from scratch using Angular 4
- Explore Angular coding and architecture best practices
- Understand and use Angular Forms, Observables, Dependency Injection, and Routing
- Retrieve, update, and delete data using Angular's Http service
- Unit test all the parts of an Angular application including Modules, Components, Services, and Pipes
- Upgrade an existing application from AngularJS to Angular 4 over time by running both frameworks in the same project
- Create, build, and deploy an Angular (Angular 4) application using the Angular CLI
- Develop dynamic Model-driven forms that are easier to unit test

ANGULAR 4 TRAINING OUTLINE

Introduction

Why Angular 4?

User Experience similar to a Desktop Application

Productivity and Tooling

Performance

Community

Full-featured Framework

Platform for Targeting Native Mobile not just Web Browsers

Understanding Angular Versions

AngularJS (Angular 1.x)

Angular

Understanding Angular 4

Drop-in replacement for Angular 2

Angular 4 Features

View Engine generates smaller code

Enhanced *ngIf syntax

Animation code now in own packages

TypeScript 2.1

Improved compiler speed

Angular Universal

Flat ESModules

TypeScript and ECMAScript 6 (ES6) Fundamentals

Classes

ES Modules

Arrow Functions

Template Literals

Scoping using Let and Const Keywords

Spread Syntax and Rest Parameters

Destructuring

Decorators (JavaScript Aspect-Oriented Programming)

Angular 4 Basics

Components

Templates

Modules

Models

Template Syntax

HTML in templates

Interpolation

Binding syntax

Property binding

Event binding

Two-way data binding

Attribute, class, and style bindings

Built-in Directives

Template Input Variables

The NgSwitch Directives

Template Reference Variables

Input and output properties

Template Expression Operators

Pipe (|)

Safe Navigation Operator (?.)

Components

Implementing the Component Lifecycle Hook OnInit

Component Communication

Services

Using a services to access data

Using a service to encapsulate business logic

Using a service to configure your application

Using a service for logging

Dependency Injection

Understanding Dependency Injection

Angular’s Dependency Injection System

Registering

Injecting

Hierarchical Injection

Template-driven Forms

NgSubmit Directive

FormsModule

NgForm, NgModel, and NgModelGroup Directives

Validation Directives

Displaying validation messages

Styling validation messages

Communicating with the Server using the Http Service

Deciding between Promises or Observables (RxJS)

Making Http GET Requests

Making Http POST and PUT Requests

Issuing a Http DELETE Request

WebSockets

Router

Importing the RouterModule and Routes

Configuring Routes

Displaying Components using a RouterOutlet

Navigating with RouterLink and RouterLinkActive Directives or the Router

Accessing parameters using ActivedRoute

Organizing your code into Modules

Testing

Tools: Jasmine, Karma

Jasmine Syntax: describe, it, beforeEach, afterEach, matchers

Setup and your First Test

Testing Terminology: Mock, Stub, Spy, Fakse

Angular Testing Terminology: TestBed, ComponentFixture, debugElement, async, fakeAsync, tick, inject

Simple Component Test

Detecting Component Changes

Testing a Component with properties (inputs) and events (outputs)

Testing a Component that uses the Router

Testing a Component that depends on a Service using a Spy

Testing a Component that depends on a Service using a Fake

Testing a Service and Mocking its Http requests

Testing a Pipe

Security

How to Prevent Cross-site Scripting (XSS)

Trusting values with the DOMSanitizer

HTTP Attacks

Security Audits of Angular Applications

Advanced Components

Component Styles using MetaData properties: Styles and StyleUrls

Change Detection Strategies

Component Lifecycle Hooks

Advanced Routing

Lazy-loading Angular Modules

Location Strategies

Nested or Child Routes

Route Guards

Advanced Dependency Injection

Providers

Using the @Optional and @Host Decorators

Model-driven Forms (Reactive Forms)

ReactiveFormsModule

AbstractControl, FormControl, FormGroup, and FormArray

FormBuilder

Validators

Attribute Directives

Creating a custom Attribute Directive using ElementRef, Render

Pipes

Built-in Pipes: Using, Passing Parameters, Chaining

Creating a custom Pipe using PipeTransform

Understanding Pure and Impure Pipes

Creating, Building, and Deploying an Angular Application

Manually

Using the Angular CLI with Ahead-Of-Time (AOT) Compilation and Tree-Shaking (removing unused library code)

Upgrade Strategies from AngularJS

Preparing your AngularJS Project

Angular 4 and AngularJS together

Redux

Redux Basics

Debugging and Time Traveling with Redux DevTools