# ANGULA 11.0 with ES6 and typescript COURSE CONTENT

## Duration:

5 days

## AngularJS Training Objectives

Angular 11 is one of the fastest, most popular open source web app frameworks today, and knowing how to use it is essential for developers. You'll learn how to create components and user interfaces, data-binding, retrieving data using HTTP, and more.

This training will provide depth knowledge on Angular 8 with more hands on. Introduces to latest web standards to implement Rich UI with Responsive, performance and with better design.

After this training participants will be able to design and implement UI with Angular 11

## PRERQUISITES

Developing Web applications using HTML, CSS

Hands-on experience in JavaScript

## LAB SET UP

* Any OS
* IDE: Visual Studio Code ([**Download**](https://code.visualstudio.com/download))
* Chrome browser (with latest update)
* NodeJS (latest) (Download: [**Mac**](https://nodejs.org/download/release/v7.4.0/node-v7.4.0.pkg), [**Windows**](https://nodejs.org/download/release/v7.4.0/node-v7.4.0-x64.msi))

## Day-1

### Object-oriented JavaScript

* [Introductory](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript)
* [Introduction to Object Oriented JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript)
* [Object-oriented programming](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Object-oriented_programming)
* [Prototype-based programming](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Prototype-based_programming)
* [JavaScript object oriented programming](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#JavaScript_object_oriented_programming)
* [The class](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#The_class)
* [The object (class instance)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#The_object_(class_instance))
* [The constructor](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#The_constructor)
* [The property (object attribute)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#The_property_(object_attribute))
* [The methods](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#The_methods)
* [Inheritance and the prototype chain](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Inheritance_and_the_prototype_chain)
* [Encapsulation](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Encapsulation)
* [Abstraction](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Abstraction)
* [Polymorphism](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Polymorphism)
* [Closures](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Closures)
* [Strict mode](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode)

### New Features in ES6 (ES 2015), ES7 (ES 2016), ES8 (ES2017) and es9 (2018)

ES6 and 7:

* Arrow Operator
* Classes
* Enhanced object literals
* Template strings
* Destructuring
* Default, rest and spread
* Let and const
* Iterators and for..of
* Generators
* Modules
* Module loaders
* Map + set + weakmap + weakset
* Proxies
* Symbols
* Promises
* Math + number + string + array + object APIs

ES8:

* Async functions
* Trailing commas
* String padding

ES9:

* Rest/Spread Properties
* Async iterations
* Regular expression improvements

## Day-2

### Introduction to typescript

* + Why use TypeScript?
  + TypeScript Features
  + TypeScript Syntax, Keywords, and Code Hierarchy
  + Tooling and Framework Options
  + Tooling and Framework Options - TypeScript Playground
  + Tooling and Framework Options - Visual Studio
  + Tooling and Framework Options - Web Essentials
  + Tooling and Framework Options - Sublime Text
  + Tooling and Framework Options - TypeScript Compiler
  + Tooling and Framework Options - NodeJS
  + Hello World Example
  + Hello World Example - Creating a Class

### Typing, Variables, and Functions

* + Overview
  + Grammar, Declarations, and Annotations
  + Type Inference
  + Grammar
  + Static and Dynamic Typing
  + Compile Time or Run Time
  + Ambient Declarations and Type Definition Files
  + The Any Type and Primatives
  + Applying Types
  + Objects
  + Functions
  + Arrow Functions and Debugging
  + Functions and Interfaces
  + Static Typing Recap
  + Optional and Default Parameters
  + Rest Parameters
  + Demo: Defining Parameters
  + Function Overloads
  + Demo: Using Function Overloads

### Classes and Interfaces

* + Introduction
  + Defining Classes
  + Demo: Defining Classes
  + Demo: Property Limitations
  + Casting and Type Definition Files
  + Demo: Casting and Type Definition Files
  + Extending Types
  + Demo: Extending Types
  + Using Interfaces
  + Demo: Using Interfaces
  + Extending an Interface
  + Demo: Extending an Interface

### Modules

* + Overview
  + Identifying a Module
  + Creating an Internal Module
  + Internal Module Accessibility and IIFE
  + Named Modules
  + Extending Modules and Importing Shortcuts
  + Organizing Internal Modules
  + Separating Internal Modules
  + External Modules and Dependency Resolution
  + Module Dependencies
  + Importing External Modules Using AMD
  + Importing 3rd Party Libraries Using AMD

### Generics

* + Introduction and Overview
  + What are Generics and Type Parameters?
  + Using Array <T>
  + Generic Functions
  + Demo: Creating and Using Generic Functions
  + Generic Interfaces and Classes
  + Demo: Creating and Using a Generic Class
  + Generic Constraints
  + Demo: Applying Constraints to Generics

### Compiler options and project configuration

* + Introduction and Overview
  + Specifying Compiler Options
  + Demo: Using Compiler Options
  + Role and Structure of tsconfig.json
  + Demo: Managing a Project with tsconfig.json

### type definitions

* + Introduction and Overview
  + What Are Type Definition Files?
  + Ambient Modules
  + DefinitelyTyped
  + Demo: Using Definitions from DefinitelyTyped
  + Managing Type Definitions with tsd
  + Demo: Installing Type Definitions with tsd
  + Managing Type Definitions with typings
  + Demo: Installing Type Definitions with typings

#### other artifacts (other concepts that are required before createion of angualr application)

* + Babel compiler
  + Module loader (ts-loader etc.)
  + Light weight servers (lite-server, webpack-dev-server etc.)
  + Bundlers (webpack etc.)

## Day-3

### Introduction to Angular 11

* + Introduction
  + Anatomy of an Angular 11 Application
  + High-Level architectural Overview of Angular 11
    - Code Simplification
    - Performance Improvements
  + Sample Application

### Setup Environment and Package Requirement, Installtion Etc

* + Introduction
  + Selecting a Language
  + Selecting an Editor
  + Setting up Our Environment
  + Setting up an Angular 11 Application
  + Running an Angular 11 Application
  + About Modules
  + Loading Modules and Hosting our Application

### Introdduction to Component

* + Introduction
  + What Is a Component?
  + Creating the Component Class
  + Defining the Metadata with a Decorator
  + Importing What We Need
  + Demo: Creating the App Component
  + Bootstrapping the App Component
  + Demo: Bootstrapping the App Component

### Templates, Interpolation and Directives

* + Introduction
  + Building a Template
  + Building the Component
  + Using a Component as a Directive
  + Binding with Interpolation
  + Adding Logic with Directives: ngIf
  + Adding Logic with Directives: ngFor

### Data Binding and Pipes

* + Introduction
  + Property Binding
  + Handling Events with Event Binding
  + Handling Input with Two-way Binding
  + Transforming Data with Pipes

## Day-3

### More on Components

* + Introduction
  + Defining Interfaces
  + Encapsulating Component Styles
  + Using Lifecycle Hooks
  + Building Custom Pipes

### Building Nested Components

* + Introduction
  + Building a Nested Component
  + Using a Nested Component
  + Passing Data to a Nested Component Using @Input
  + Passing Data from a Component Using @Output
  + Using @ViewChild()

### Services and Dependency Injection

* + Introduction
  + How Does It Work?
  + Building a Service
  + Registering the Service
  + Injecting the Service

### Retrieving Data using Http

* + Introduction
  + Observables and Reactive Extensions
  + Setting Up
  + Sending an Http Request using HttpClient in angular 11
  + Subscribing to an Observable

## Day-5

### Navigation and Routing

* + Introduction
  + Setting Up
  + Configuring Routes
  + Tying Routes to Actions
  + Placing the Views
  + Passing Parameters to a Route
  + Activating a Route with Code

#### Working With Forms

* + Angular Forms API
    - Form Controls, Control groups
    - FormBuilder
    - Form Directives
  + Forms Validation
    - Programmatic Approach
    - Custom Validators
    - Custom Validation Directives
  + Template-Driven Forms
  + Model-Driven Forms

#### Testing Angular applications

* + Unit Testing – Jasmine & Karma
  + Test a Class
  + Test a Component
  + Test a Pipe