Building Node.js App With TypeScript

Duration: 5 Days; Instructor-led

WHAT YOU WILL LEARN

TypeScript is an open source language for building enterprise JavaScript applications. It is a strict superset of JavaScript that compiles to plain JavaScript and brings static typing and object-oriented development to the language. Server-side JavaScript with Node.js and Express course teaches experienced JavaScript developers how to create server-side applications with JavaScript and Node.js, culminating with an MVC application built on the Express framework that queries databases and calls back-end web services. Node.js is a powerful tool for controlling servers, building web applications, and creating event-driven programs. Node.js takes JavaScript a language familiar to all web developers out of browser.

AUDIENCE

None

PREREQUISITES

REQUIRED PREREQUISITES

Before attending this course, students should have general programming experience and knowledge of HTML, CSS and JavaScript.

METHODOLOGY

This program will be conducted with interactive lectures, PowerPoint presentation, discussion and practical exercise.

COURSE OBJECTIVES

All participants will be able to:

- Understand why server-side JavaScript is useful
- Explain how Node.js is architected to allow high scalability with asynchronous code
- Automate tasks with Gulp
- Build an HTTP server using the core modules in Node.js
- Use stream I/O to efficiently serve the web pages
- Test the reliability of the application with unit tests
- Convert the application to an MVC framework using Express

COURSE OUTLINES

Module 1 - Introduction to "Modern (ES5)" JavaScript

- Basic language features
- Creating variables with var
- Functions and functional programming
- Nesting functions
- Nesting with function reference variables
- Creating Objects with JavaScript literal syntax and constructor functions
- Indexed property access
- Typeof and objects
- Special comparisons for objects
- Scope and Context
- JavaScript visibility rules
- Global variables
- Object functions and execution
- The this keyword in JavaScript
- By reference – vs – by value
- JavaScript types: Number, String, Boolean, Date
- JavaScript == and false conditions
- Meet Triple Equals ===
- Primitives and wrapper objects in JavaScript
- Arrays – Ordered sequence of items
- Key JavaScript built-in objects
- Using Object.create
- Prototype – key points
- Metadata – built-in functions and objects
- Closure

Module 2 - Getting Started with TypeScript

- Course Introduction
- Introduction
- Why use TypeScript?
- TypeScript Features
- Tooling and Framework Options - Sublime Text
- Tooling and Framework Options - TypeScript Compiler
- Tooling and Framework Options - NodeJS

Module 3 - Typing, Variables and Functions

- Overview
- Type 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
- Summary

Module 4 - Classes and Interfaces

- Introduction
- Defining Classes and Properties
- Casting and Type Definition Files
- Extending Types
- Using Interfaces
- Extending an Interface

Module 5 - 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

Module 6 - Intro to Node.js

- RAM vs. I/O latency
- Blocking vs. Non-Blocking
- Event-driven Programming
- Event Loop
- Blocking The Event Loop
- Node.js Philosophy

Module 7 - Node.js Platform Setup

- Download and Install
- Node REPL
- Using Node.js to execute scripts
- First Hello World

Module 8 - Modules and npm

- Anatomy of a module
- The Node Package Manager
- Private code
- Accessing and using modules
- npm commands
- Creating a project
- package.json – Configuration file
- Global vs local package installation
- Automating tasks with Gulp

Module 9 - The Callback Pattern

- What are callbacks
- Callback-last
- Error-first

Module 10 - Events

- When to use Event Emitters
- Binding Functions to Events
- Event Requests
- Event Listening

Module 11 - Error Handling

- Callbacks: Error-first
- Errors in Event Emitters
- Uncaught Exceptions
- Using Domains

Module 12 - File System

- Synchronous vs Asynchronous I/O
- Path and directory operations
- dirname and ___filename
- Asynchronous file reads and writes

Module 13 - Buffers

- Why Buffers exist
- Creating Buffers
- Reading and Writing Buffers
- Manipulating Buffers
- Using buffers for binary data
- Flowing vs. non-flowing streams
- Streaming I/O from files and other sources
- Processing streams asynchronously
- Configuring event handler

Module 14 - Streams

- What are streams
- Read and Write Stream API
- Flow Control
- Piping
- Duplex Stream
- Transform Stream
- Configuring event handler

Module 15 – HTTP

- The HTTP protocol
- Building an HTTP server
- Rendering a response
- Processing query strings
- Using Representational State Transfer
- Configuring TLS

Module 16 - Express.js

- Intro and Installing Express.js
- The model-view-controller pattern
- Building a front-end controller
- Building a Hello Express application
- Creating routes
- Creating actions
- Rendering Layouts
- Using templates
- Adding partials
- Using locals and conditional templates
- Modularizing routes using REST
- Reading POST data
- Adding middleware

Module 17 - Socket.io

- The Basics
- Server side
- Client side
- Sending and Receiving Objects
- Streaming Data

Module 18 - Connecting to Databases

- How Node.js connects to databases
- RDBMS databases and NoSQL Document Stores
- Connecting to RDBMS and NoSQL databases
- Configuration and platform setup
- Performing CRUD Operations
- Building client requests to web services

Module 19 - Modules and Unit Testing

- Modularization
- The CommonJS and RequireJS specifications
- Defining modules with exports
- Modules are singletons
- Creating a package
- Module scope and construction
- Unit testing frameworks
- What to test and how to test it
- Building unit tests with Mocha