UI Training

Duration: 3Days

Prerequisite: Programming Knowledge in any language

Day 1

Module 1: **Core JavaScript**

Overview of JavaScript

JavaScript Core Language Reference

JavaScript Introduction

Popups in JavaScript

Variables, Functions, and Scopes

Data Types in JavaScript

Math, Number, and Boolean Objects

Functions

Parameters

return with Functions

Default Values

arguments Object

Control Structures

Exception Handling

Conditionals and Loops

Expressions and Statements

JavaScript Operators

Literals in JavaScript

The Array Literal

The Object Literal

The String Object

The Date Object

Timings Methods

Document Object Reference

Document Object Model Essentials

Generic HTML Element Objects

Location & History Objects

The Document & Body Objects

Image Object

Form Object

Button Object

Event Object

Style Sheet & Style Objects

Debugging

Module 2: Advanced JavaScript

Advanced Functions in JavaScript

Arrow Function

Callback Functions

Bind()

Currying

Chaining

Modules of ES6

Typed Array

Set and Map

Cookies and Storage

Important Features

Spread Syntax

Rest Parameters

Getter and Setter

Ajax

JSON

Object-oriented JavaScript (OOJS)

Prototypal Inheritance

Built-in Prototypes

Functional Inheritance

Closures

Class Abstractions

Name-spacing

Polymorphism

Encapsulation

Asynchronous Execution and Timers

Promises

Await

Sync

JavaScript Design Patterns

OO Design Patterns

JavaScript-specific Coding Patterns

Singleton Pattern

Module Pattern

Revealing Module Pattern

Factory Pattern

Anti-patterns

Object Creation Patterns

Code Reuse Patterns

JavaScript Code Compression

JavaScript Loading Strategies

Dependency Management of JavaScript

JavaScript Security

Assignments

**Typescript**

1. Introduction to TypeScript

• History and Importance of JavaScript

• JavaScript Strengths, Weaknesses

• Limitations of JavaScript

• Advantages of TypeScript

• TypeScript Design Goals

• Installing TypeScript

• Getting Started

2. TypeScript Language Basics

• Declaring Variables

• Basic Types

• Type Annotations

• Non-Nullable Types

• Basic Data Structures

• Operators

• Flow Control Statements

3. TypeScript Type System

• Type Compatibility

• Type Inference

• Access Modifiers

• Union Types

• Type Guards

• Intersection Types

• Type Aliases

• String Literal Types

• Polymorphic “this” Types

3. Functional Programming

• Functional Programming

• Named Functions

• Arrow Functions

• Function Types

• Optional and Default Parameters

• Rest Parameters

• Function Overloading

Day 2

## **Node.js Training**

## Prerequisites

Node.js training attendees should have a thorough knowledge of JavaScript. They should be familiar with web server application design concepts (such as accessing databases and SOA concepts), as well as basic HTML and CSS.

## Hands-on/Lecture Ratio

This Node.js training class is 65% hands-on, 35% lecture, with the longest lecture segments lasting for 20 minutes.

## Node.js Training Objectives

* Learn why server-side JavaScript is useful
* Install Node.js
* Learn how Node.js is architected to allow high scalability with asynchronous code
* Create basic web applications with Node.js
* 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
* Create modules to organize the server
* Test the reliability of the application with unit tests
* Convert the application to an MVC framework using Express
* Interface to a MongoDB database and a web service

## Node.js Training Outline

* Introduction
* Foundation
  + The Node.js framework
  + Installing Node.js
  + Using Node.js to execute scripts
* Node Projects
  + The Node Package Manager
  + Creating a project
  + The package.json configuration file
  + Global vs. local package installation
  + Automating tasks with Gulp.
* HTTP
  + The HTTP protocol
  + Building an HTTP server
  + Rendering a response
  + Processing query strings
  + Using Representational State Transfer
  + Configuring TLS
* File System
  + Synchronous vs. asynchronous I/O
  + Path and directory operations
  + \_\_dirname and \_\_filename
  + Asynchronous file reads and writes
* Buffers, Streams, and Events
  + Using buffers for binary data
  + Flowing vs. non-flowing streams
  + Streaming I/O from files and other sources
  + Processing streams asynchronously
  + Configuring event handlers
* 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
* Express
  + The model-view-controller pattern
  + Defining Jade and Handlebars templates
  + Building a front-end controller
  + Defining routes
  + Creating actions
  + Configuring Express to use Handlebars
  + Using REST
  + Reading POST data
  + Building Handlebars helpers
  + Adding middleware
* Data Sources
  + How Node.js connects to databases
  + RDBMS databases and NoSQL databases
  + Connecting to RDBMS and NoSQL databases
  + Performing CRUD operations
  + Building client requests to web services
* Conclusion

Day 3

**MobX**

Introduction

* MobX vs Redux vs Relay

Understanding State Management

* How React handles local state
* How Redux manages state
* How MobX manages state

Getting Started

* Preparing the Development Environment
* Installing and Configuring MobX

Building an Application

* Overview of the component architecture
* Creating a React Application (Components, Element, JSX, etc.)

Working with React Component State

* Initializing and updating state
* Working with forms

Implementing a MobX Data Store

* Setting up the data store
* Passing data to components
* Listing out contents of data store

Managing Changes to Data

* Adding data to store
* Notifying React of the change in data state

Updating the View

* Declaring observable data
* Updating the view

Improving DataFlow

* Modifying state data through actions
* Calling actions (avoiding unsafe data manipulation)

Working with Computed Values

* Perforing calculations on core data

Using Advanced Development Tools and Techniques

* Creating components with ES2017
* Using Babel for ES2017 and JSX
* Using WebPack for React and MobX
* Performing asynchronous actions with MobX

Testing the Application

* Creating unit tests for React components
* Choosing tools and frameworks (Jest, Enzyme, TestUtils, etc.)
* Creating unit tests for MobX stores
* Debugging the Application

Deploying the Application

* Tooling and automation
* Implementing Continuous Integration (CI)

Troubleshooting